10th European Summer School in Logic, Language and Information

organized by:
Universität des Saarlandes Saarbrücken
in cooperation with DFKI Saarbrücken
under the auspices of the European Association
r Logic, Language and Information (FoLLI)

"Recent Advances in Corpus Annotation"

Brigitte Krenn, Thorsten Brants,
Wojciech Skut, Hans Uszkoreit

Workshop

Language and Computation

Construction and Annotation of Test-Items in DiET

Judith Klein^, Sabine Lehmann*, Klaus Netter^, and Tillmann Wegst^

DFKI^ GmbH
Stuhlsatzenhausweg 3
D-66123 Saarbrücken
(Germany)
Firstname.Lastname@dfki.de

ISSCO*, University of Geneva
54, route des Acacias
CH-1227 Geneva
(Switzerland)
Sabine.Lehmann@issco.unige.ch

1 Motivation

As industrial use of language technology is flourishing and the market for competing software systems is expanding, there is an increasing need for the assessment of NLP components on the level of adequacy evaluation and quality assurance. However, effective and efficient assessment is often restricted by the lack of suitable test material and tools. One approach traditionally used for testing and evaluating NLP systems comprises *test-suites,* i.e. systematic collections of constructed language data associated with annotations describing their properties.

2 Project Aims

The objective of the DiET[1] - Diagnostic and Evaluation Tools for NLP Applications - project is to develop a tool-box which will allow the user to construct test-suites, and to adapt and employ these resources for the evaluation of NLP components. The project builds on the results of the tsnlp (Test Suites for Natural Language Processing) project (Lehmann *et al.* 1996) where a test-suite database tsdb[2] was built which contains substantial English, French and German test-suites covering a wide range of syntactic phenomena.
DiET aims at extending and developing the tsnlp test-suites with annotated test-items for syntax, morphology and discours phenomena, but DiET is not a mere continuation of tsnlp . It goes beyond the tsnlp approach and tries to overcome the data-related and technological shortcomings of tsdb (as summarized e.g. in (Klein 1996)). The main focus of DiET is no longer the building of a large amount of test material but the development of more sophisticated diagnostic and evaluation tools - efficiently working under/behind* a user-friendly graphical interface - for the creation, retrieval and employment of such data. The DiET system provides a flexible annotation schema which comprises a number of already existing annotation-types and allows the user to easily modify these types or define a completely different set of annotation-types according to his/her specific needs. In contrast to tsnlp where the adaptation of the (conceputally) flexible annotation schema is possible - as for example, (Flickinger and Oepen 1997) show - but only for people familiar with relational database systems[3] in DiET these extensions and modifications can easily be one by means of graphical interface tools.
While the first set of tools is hence devoted to the construction of annotated test-items, the second set concerns the customisation of test and reference data to specific applications and domains, i.e. the user is given the means to construct, modify and extend data as well as the list of annotations for his/her specific requirements. Customisation efforts focus on three different types of applications, namely Grammar and Controlled Language Checkers, Parsers as components of Machine Translation Systems and Translation Memories. Under the heading of customisation a whole range of different processes and operations are subsumed: (i) functions to perform adaptation to new domains on the basis of lexical replacement, (ii) the development of document profiles, establishing a relation between test-suites and domain specific corpora and (iii) the support for the process of adapting and customising the test material for concrete evaluation procedures.
This paper focuses on the annotations, i.e. customisation processes are only addressed inasmuch as they are reflected in the annotation schema. Also, only the tools immediately relevant for the annotation task are discussed.

3 Annotation Schema

As already mentioned, within the scope of DiET a set of annotation-types will be specified. For this

[1]The project started in April 1997 and lasts until April 1999. More information on DiET can be obtained through the world-wide web under http://dylan.ucd.ie/DiET/.
The DiET project is supported by the European Commission and the Swiss government under the Telematics Application Programme (Language Engineering LE 4204).
[2]The test-suite database tsdb can be directly accessed via http://tsnlp.dfki.uni-sb.de/tsnlp/tsdb/tsdb.cgi

[3]The modification of the annotation schema, realised as conceptual database schema, requires basic knowledge on relational database design in order to re-arrange the internal database relations.

purpose the annotation-types defined in tsnlp were revised. Most of the attributes relevant for syntactic analyses (i.e. *structural and functional information, wellformedness),* phenomenon descriptions (e.g. *phenomenon supertypes)* and formal properties (e.g. *origin)* will be kept for DiET, changing only the representation format. Some of the salient characteristics of the syntactic phenomena (the so-called *parameters)* remain unchanged (e.g. *agreement and verbal subcategorisation),* others need to be consistently defined and labelled for the three languages (e.g. *coordination attributes).* A few annotation-types (which were already problematic in tsnlp , e.g. *phenomenon interaction)* remain to be defined. In general, some new annotation-types describing formal and linguistic properties need to be specified. Furthermore, the whole complex of application-specific annotations will be worked out for the three applications chosen for DiET since tsnlp made only a general proposal on *user & application profile* annotations. Finally, DiET will provide two completely new sets of annotations: one comprises annotation-types describing corpus-related information, the other one consists of annotation-types describing evaluation-specific aspects.[4]

Since it is impossible to foresee all information needed by a user, the annotation schema is open to customisation and extension. It is therefore designed to be flexible and can be tailored to the user requirements: the system will not only provide the means to annotate data on the basis of a given set of different annotation-types, but will allow the user to define his/her own specific types. Many of the annotations consist of simple features attributing certain values to the test-items, but there are also structural annotations, for example to describe the internal phrasal and relational structure of the items.

The objects that an annotation may be attached to, are (i) test-items (strings), (ii) (ordered) groups of test-items, and (iii) segments of test-items. The test data are assigned a broad range of different annotations describing (i) linguistic properties, (ii) application-specific features, (iii) corpus-related information based on text profiling, and (iv) evaluation-specific attributes.

3.1 Formal and Linguistic Annotations

In principle the annotations interpret and classify a test-item as a whole, including language, format variations, information about well-formedness and the test-item level (i.e. phrasal, sentence or text level).

The linguistic annotations of the test-items comprise morphological, syntactic and discourse analysis. The morphological annotations provide information on lexical category and attaches the lexical items to the respective ambiguity class. At the syntax level - following the example of the NEGRA annotation tool (Skut *et al.* 1997) - graphical tree and dependency representations will provide information on the structural analysis of the test-items where the non-terminal nodes are assigned phrasal categories and the arcs some grammatical functions (subject, object, modifier, etc.). It is possible to assign a syntactic well-formedness to the overall structure. The discourse analysis provides information on direction (e.g. *antecedent)* and type (e.g. *co-reference)* of semantic relations between test-item-segments.

All test-items can be classified according to the linguistic phenomenon illustrated. In order to characterise test-items for the *salient properties* of a specific phenomenon - which are language-specific - additional annotation-types will be defined. In German, for example, the annotation-types *case, number* and *gender* need to be specified for the syntactic phenomenon *NP agreement.*

3.2 Application-specific Annotations

The annotation schema also provides information about application-specific properties of the test-items such that the users can formulate specific search statements retrieving suitable test material. But these annotations also serve as reference material which can be used within the comparison phase of the evaluation procedure. For grammar and controlled language checker, for example, annotation-types can be defined, which specify the error type of ill-formed test-items and connect the ungrammtical test-items to their grammatical pendants. The test material for parsers can be annotated with the number of alternative readings for ambiguous test-items. For translation memories the test-items of the source language can be connected to adequate translation units from the translation system.

In addition to these pre-defined application-specific annotation-types, the users can easily extend the annotation schema for their specific needs.

3.3 Corpus-related Annotations

Systematically constructed data are useful for diagnosis but should not be deemed sufficient for adequacy evaluation. What is needed are test-suites related to test corpora to provide weighted test data, since for evaluation purposes it is very important how representative a test-item is of a certain (text or application) domain, whether it occurs very frequently, whether it is of crucial relevance, etc. Establishing a relation between the isolated, artificially constructed test-items and real-life corpora presupposes the identification of the typical and salient characteristics of the examined text domain. This process is called *text profiling.*

[4]It is important to note, that within the DiET project, these annotation-types will just be specified. The DiET group does not aim at annotating all test-items according to these annotation-types.

DiET will integrate the NEGRA tagging and parsing tools to semi-automatically annotate corpus texts for part-of-speech and possibly also structural and dependency information. Analysis tools will be employed to establish a text profile containing information on frequency and distribution of text elements. The information in the profile will include non-linguistic information such as use of punctuation marks as well as a description of linguistic characteristics such as sequences of lexical categories, patterns of specific lexical items and lexical categories or even structural descriptions. Manual inspection and interpretation of the resulting profile will reveal which linguistic constructions are salient in the examined text and hence relevant for the envisaged application domain.

Once the corpus-relevant linguistic characteristics are identified they will be used to (i) elaborate a relevance measure, i.e. decide to what extent certain criteria are significant to establish the relevance values and to (ii) select test-items and classify them for *relevance* (extending the annotation schema for this new annotation-type). If, for example, the text profile records a high percentage of co-ordinating conjunctions between two nominal phrases, the user will select test-items exemplifying simple NP co-ordination and give them a high relevance value. This is a simplified example since very often, the relevance value will be some kind of a composition of the values given to several annotation-types, e.g. *sentence-length*, *number-of-conjunctions, number-of-coordinated-elements, NP-type, number-of-NP-elements,* etc.

Currently, the responsible DiET group works on the conceptual design to realise text profiling and allow for the linkage between test-items and text corpora, i.e. the concrete mechanisms for this customisation process can not be provided yet.

3.4 Evaluation-specific Annotations

The annotation schema will comprise annotation-types describing the evaluation scenario, including the user-type (developer, user, customer), type and name of the system under evaluation, goal of the evaluation (diagnosis, progress, adequacy), conditions (black box, glass box), evaluated objects (e.g. parse tree, error correction, etc.), evaluation criteria (e.g. correct syntactic analysis, correct error flag, etc.), evaluation measure (the analysis of the system results, for example in terms of recall and precision), and the quality predicates (e.g. the interpretation of the results as *excellent, good, bad,* etc.).

Since the DiET database is also meant to accommodate evaluation results, the annotation schema will comprise annotation-types to annotate the actual output of the examined NLP system to individual items. This information may allow for the (at least manual) comparison of the actual to the expected output (the reference annotation specified within the application-specific annotation-types).

Clearly, not all annotations can be gathered automatically but the DiET system will support the users to organise evaluation runs and keep track of the results.

3.5 Meta-information on the Annotations

The database also provides information about the test-suite as a whole, such as listings of the vocabulary occurring in the test-items, part-of-speech tag sets or the terminology used for the specification of annotation-types. Additionally, (declarative) information will be assigned to the linguistic phenomena, comprising the (traditional) phenomenon name, its relation to other phenomena displayed in a hierarchical phenomenon classification and a (informal) description of the linguistic constructions belonging to the phenomenon in question.

4 Annotation Tool

The objective of DiET is to offer a flexible tool package which is open enough to cover the requirements of different types of users who wish to employ the system for a range of applications. A tool package accessible from a graphical user interface portably implemented in Java is being built. DiET has a client/server architecture, with a central database system, lexical replacement, tagging and syntactic analysis working as servers and a client integrating construction, display and editing facilities. The central construction and annotation tool serves to enter new data and to annotate these items on different levels of linguistic abstraction, such as phrasal constituency or anaphoric relations and for various application- and corpus-specific information. The process of annotation is supported among others by part-of-speech tagging and parsers with bootstrapping facilities learning from existing annotations.

Figure 1 gives an impression on how the main window of the tool will look like. In principle, the interface aims at simplicity in design and offers clearly arranged operation fields to provide an easy to use annotation tool. The left window contains the test-items. The right one is split up into two parts: the upper window shows the hierarchically arranged annotation-types together with the values attributed to the selected test-item, the lower part presents more information on the value(s) of the annotation-type marked in the window above.

How to annotate a test-item? First, the user selects a test-item. From the pool of annotation-types (in the upper right window), s/he choses an annotation-type, e.g. *syntactic analysis, NP.coordination, etc.* In the lower right window, fields, appropriate for the given annotation-type, will appear allowing the

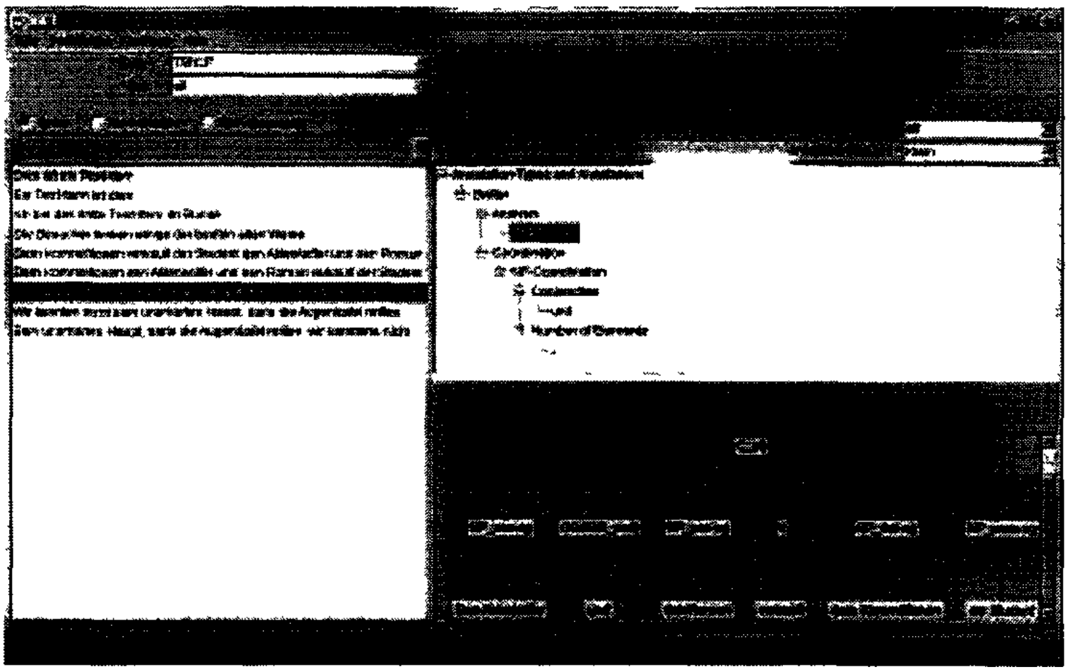


Figure 1: DIET GUI

value(s) to be enters. In the case of syntactic analysis for example, this will be a tree window.

It is often the case that the construction of test-items can be based on already existing test-items since only minor changes in the test-items or in the annotations need to be done. DiET gives therefore the possibility to duplicate and adapt a similar entry rather than producing the test-item string and its annotations from scratch.

The user can also easily specify new annotation-types. For the definition of annotation-types a dialog-window will open (see figure 2). To define a new annotation-type, the user choses a name for the new type, attributes it to the respective data type, (if necessary) defines the range of allowed values, and positions it within the hierarchically ordered list of annotation-types. Annotation types can be applied to different types of linguistic objects (i.e., test-items, groups of test-items, etc.) and their values (if any) can be configured to be entered manually or are alternatively provided through some server, i.e. the user selects a service (e.g. a tagger) which will provide the values. Instances of such annotation types could be, for example, phrasal or relational structures over strings building on the data type tree, anaphoric relations making use of a simple data type arc, well-formedness judgements with a boolean value, etc.

While most of the described functions of declaration, selection, and data entry will be carried out in the central client module, there will also be a number of specialised and potentially decentralised servers supporting the tasks of data construction and annotation. (Semi-)automatic annotation of data by servers is forseen for standard annotation types such as part of speech tagging. These will be available for the three languages, as will be a morphology component to assign standardised morpho-syntactic classifications.

5 Concluding remarks

DiET supports NLP evaluation by the development of tools which allow to construct and customise annotated test material which can consist of a systematic collection of constructed language data or be real life corpora. In contrast to the Penn Treebank (Marcus *et al.* 1994) and the NEGRA annotation tool (Skut *et al.* 1997) DiET provides the means to attach richer annotations to the data which go beyond part-of-speech tagging and syntactic analysis, namely the possibility to assign salient phenomenon-specific characteristics, the so-called *parameters* to the test-items. Furthermore, semantic information can be associated with test-item-segments above sentence level.

The project focuses on the flexibility of the annotation schema which - with the given tool-box - can easily be tailored according to the requirements of the user: the test-items can be annotated with respect to formal and linguistic properties, application-specific features, corpus-related information based on document profiling and evaluation-

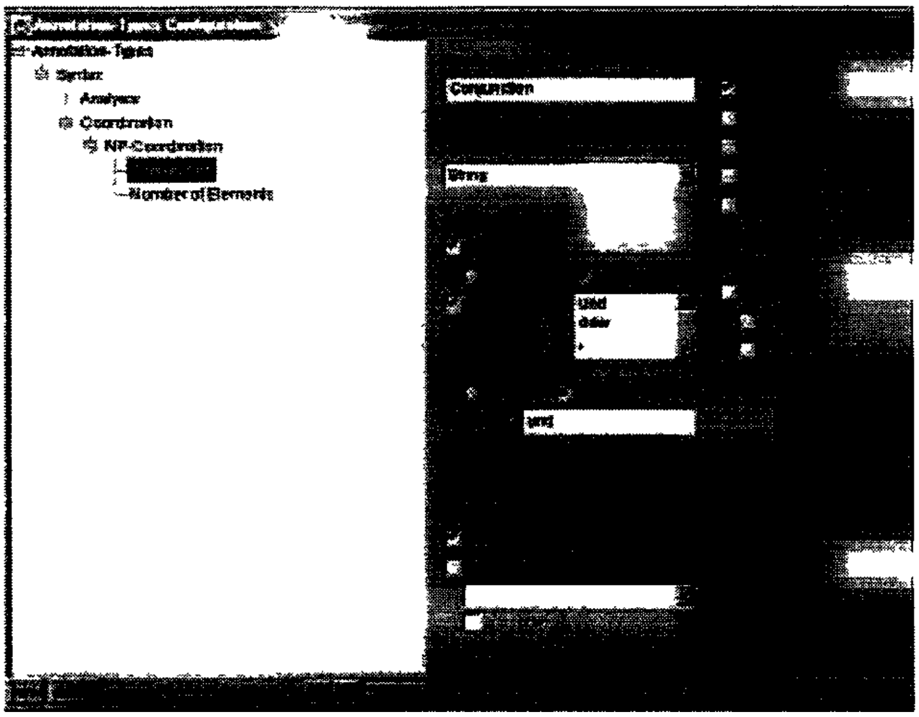


Figure 2: Configuration of annotation-types

specific attributes.

The project also develops a set of tools which support the customisation of test-suites by adapting them to new domains on the basis of lexical replacement, the development of text profiles and the support for evaluation purposes.

Acknowledgement

We thank all our DIET colleges, Susan Armstrong (ISSCO), Jochen Bedersdorfer (DFKI), Tibor Kiss (IBM Germany), Bernice McDonagh (LRC), David Milward (SRI Cambridge), Dominique Petitpierre (ISSCO), Steve Pulman (SRI Cambridge), Sylvie Regnier-Prost (Aerospatiale), Reinhard Schaler (LRC), and Hans Uszkoreit (DFKI) for their contributions.

References

Dan Flickinger and Stephan Oepen: *Towards Systematic HPSG Grammar Profiling. Test Suite Technology Ten Years After* In: Proceedings of DGfS-CL, Heidelberg 1997

Judith Klein: *TSDB - Em Informationssystem zur Unterstützung der syntaktischen Evaluierung natürlichsprachhcher Systeme.* Master's Thesis, Saarbrücken 1996

Sabine Lehmann, Stephan Oepen, Sylvie Regnier-Prost, Klaus Netter, Veronika Lux, Judith Klein, Kirsten Falkedal, Frederik Fouvry, Dominique Estiva!, Eva Dauphin, Herve Compagnion, Judith Baur, Lorna Balkan, and Doug Arnold: tsnlp — *Test Suites for Natural Language Processing* In: Proceedings of COLING, Kopenhagen 1996 pp.711-716

Marcus, M. P. et al.: *The Penn Treebank: Annotating Predicate Argument Structure.* ARPA Human Language Technology Workshop 1994 (http://www.ldc.upenn.edu/doc/treebank2/arpa94.html)

Klaus Netter, Susan Armstrong, Tibor Kiss, Judith Klein, Sabine Lehmann, David Milward, Sylvie Regnier-Prost, Reinhard Schaler, Tillmann Wegst: *DiET - Diagnostic and Evaluation Tools for Natural Language Processing Applications* In: Proceedings of the first international Conference on Language Resources and Evaluation, Granada 1998 pp.573-579

Stephan Oepen, Klaus Netter and Judith Klein: tsnlp — *Test Suites for Natural Language Processing.* In: Nerbonne, J. (Ed.): Linguistic Databases. CSLI Lecture Notes. Standford 1997 pp.13-37

Wojciech Skut, Brigitte Krenn, Thorsten Brants, and Hans Uszkoreit *An Annotation Scheme for Free Word Order Languages* In: Proceedings of ANLP 1997, Washington 1997 pp.88-96

GLOSS: A Visual Interactive Tool for Discourse Annotation

Dan Cristea **Ovidiu Craciun** **Cristian Ursu**

University "AI.Cuza" Iasi
Faculty of Computer Science
16, Berthelot St.
6600 - Iasi, Romania
{dcristea, noel, cursu}@thor.infoiasi.ro

Abstract

We present an annotation tool, called GLOSS, that manifests the following features: accepts as inputs SGML source documents and/or their database images and produces as output source SGML documents as well as the associated database images; allows for simultaneous opening of more documents; can collapse independent annotation views of the same original document, which also allows for a layer-by-layer annotation process in different annotation sessions and by different annotators, including automatic; offers an attractive interface to the user; permits discourse structure annotation by offering a pair of building operations (adjoining and substitution) and remaking operations (undo, delete parent-child link and tree dismember). Finally we display an example that shows how GLOSS is employed to validate, using a corpus, a theory on global discourse.

1. Introduction

It is generally accepted that annotation increases the value of a corpus. The added value resides in the human expertise that is so contributed. A fully automatic annotation, although perhaps not foreseeable, would make the very idea of annotating a corpus completely useless, as, in that hypothetical moment of the future, all human expertise would be totally reproducible. If such a picture is not realistic, still there will always exist the need for low level automatic annotation tasks, that could speed up tedious phases of the annotation work done by humans. There is also a tremendous need for advanced tools able to help this process, acting in an interactive manner.

On the other hand, recent progress on corpus driven methods for studying natural language phenomena puts more and more accent on the reusability of linguistic resources. It becomes natural to think that annotated corpora created for a certain goal could further be used on a different purpose by another research team that finds in the existing annotation a partial fulfilment of its needs and would like to add the missing parts by its own efforts. In order to speed up an annotation task it is also plausible that the workload be distributed to independent annotators, each responsible for just one set of markups. In such a case it is desirable that the annotators work in parallel, on different copies of the same base document, and an integrating device exists able to combine the contributing markups in a single document.

Such a need is described in Cristea, Ide and Romary (1998a), where a marking procedure is proposed in order to deal with the goal of annotating corpora from two different perspectives: discourse structure and reference. This scheme yielded by the necessity to validate a theory on discourse structure in correlation with anaphora (Cristea, Ide and Romary, 1998b).

This paper presents an annotation tool, called GLOSS, that implements the ideas presented in (Cristea, Ide and Romary, 1998a) regarding multiple annotation views on the same original document, while also providing a powerful visual interactive interface for annotation of discourse structure. The result of the annotation process is a pair of files: an ASCII SGML document, and its database mirror, which can be SQL-accessed.

2. The annotator's features

SGML compatibility

The annotator accepts in input original un-annotated texts as well as SGML documents which must be paired with their DTD files[1]. At any moment of the annotation process the

[1] The current implementation allows for a simplified DTD syntax.

document under annotation can be saved in the SGML format.

Database image copy

During the annotation process, an internal representation of the markups is kept in an associated database. The database records markers to text as pairs of beginning and end indices in the bare text document. When an annotation session is finished, the associated database can be saved for interrogation purposes. GLOSS supports simple queries to be addressed to the database, in the SQL language, interactively during an annotation session.

Once a database image of a document exists, it can act as input for a subsequent annotation session with GLOSS. This allows for enriching certain types of tags by an automatic procedure, as sketched in figure 1. In this way manual annotation can be combined with automatic annotation in an easy way.

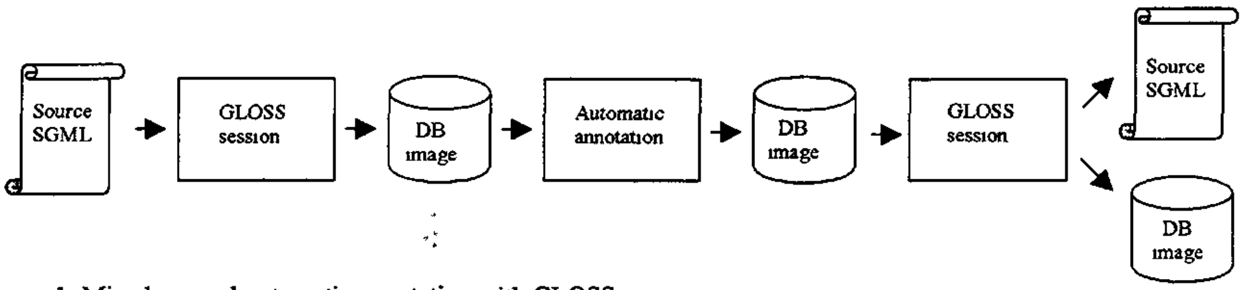


Figure 1: Mixed manual-automatic annotation with GLOSS

Multiple parentage / multiple views feature

The annotator allows for simultaneous opening of more than one document, the same way an editor behaves. Within the same session the user can commute from one document to another, each having an associated workspace window (as shown below)

The multi-document feature is justified by the multiple-views philosophy that it implements (Cristea, Ide and Romary, 1998a). According to this philosophy, a document can inherit from multiple documents, considered as its parents in a hierarchical structure. The overall architecture is that of a directed acyclic graph (DAG) with a base document as root. All documents in the hierarchy represent annotations made from different perspectives of the same original hub document. If a document is defined to inherit from one or more parents, then any or all of the contributing markups can be displayed simultaneously on the screen.

The multiple views behaviour is obtained by a process of intelligent unifying of the database representations of the parent documents. So, when a document is declared to inherit from two or more parent documents, another database is generated that copies once from the parents the common parts and than adds the markups that are specific to each of them. Ulterior to the moment the declaration of parentage is made, which must occur when a new view is created, the current document looses any connection with the parent documents, such that any modification can be made to this version without affecting in any way the original markups.

Dynamic versus static inheritance in the views architecture

Following current approaches on inheritance systems (Daelemans, De Smedt & Gazdar, 1992) for instance), two styles of recording data can be used in a hierarchy of views. In a dynamic manner only new information is recorded, while old information is referred to by pointers in one of the hierarchically upper views (in fact - their associated database images). This mimics a kind of monotonic inheritance. This decision is one that yields maximum of parsimony in recording data. On the other hand still, it obliges that all hierarchically upper views be simultaneously open and active during an annotation or interrogation session. Also it blocks performing any modification in the original text in any of the documents in the hierarchy. This is because alignments would be lost down the hierarchy, as text indexes could be changed.

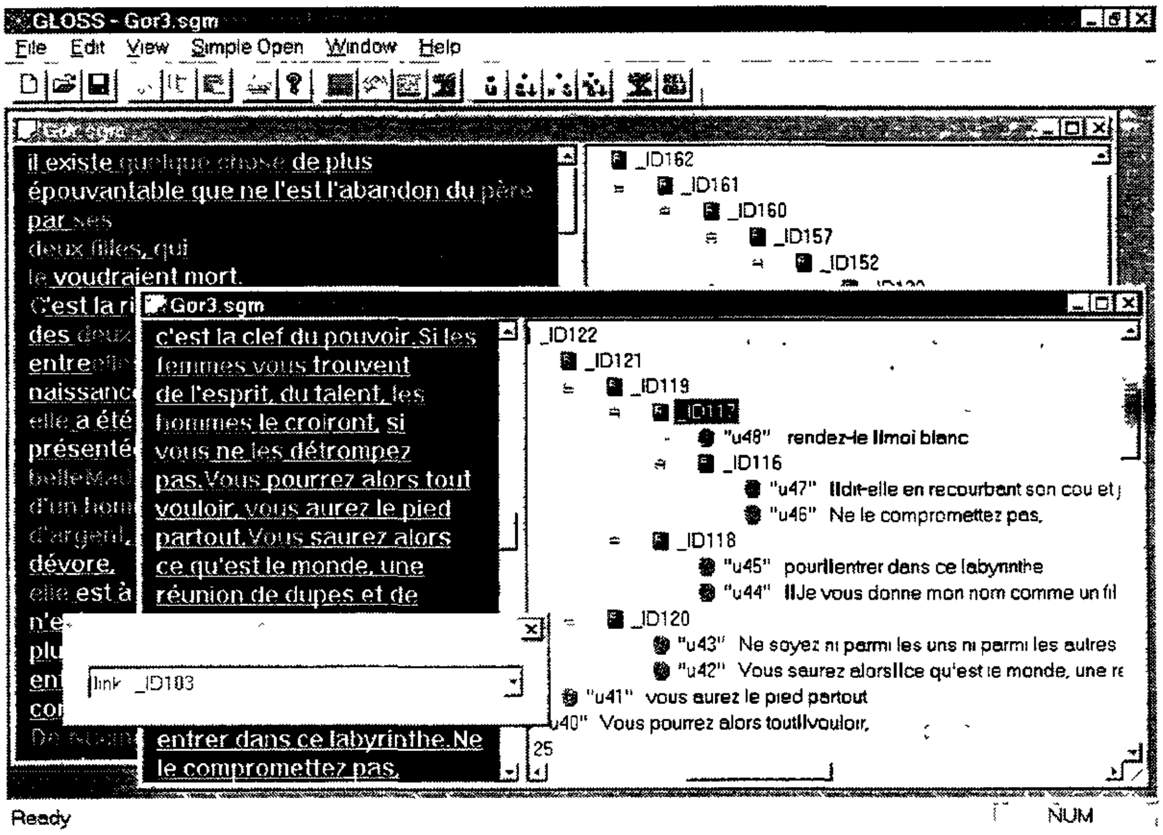
On the contrary, in the static manner a new copy of the database is created for each new view. Although recording anew the inherited data, this style permits independence from the original parents during annotation and interrogation. As such, different views on the hierarchy are isolated and can diverge significantly. The only restriction is that all parents of a child view be based on the same text document.

GLOSS implements the static inheritance philosophy.

The main interactive workspace

Each opened document under annotation has a main window where the original text is displayed (see figure 2). The SGML tags do not appear on this screen. Instead, the tagged segments of text are highlighted in colours. The user can assign a different colour to each type of tag. Highlights are sensible to the mouse click and can open for visualisation and/or modification the corresponding attribute-value structures.

Text-empty tags, as for instance co-reference links (Brunesseaux and Romary, 1997) or, in a certain representation, discourse relations, can also be displayed. Since there is no text that they could be anchored to by highlighting, only their IDs are displayed. This happens in a child window where the text-empty tags are grouped by their types. As in the case of tags that surround text, the text-empty tags can be opened for inspection and/or modification with the mouse click.



il existe quelque chose de plus
épouvantable que ne l'est l'abandon du père
par ses
deux filles, qui
le voudraient mort.
C'est la ri
des deux
entre elle
naissanc
elle a été
présenté
belle Ma
d'un hom
d'argent,
dévore,
elle est à
n'e
plu
en
co
De

c'est la clef du pouvoir. Si les
femmes vous trouvent
de l'esprit, du talent, les
hommes le croiront, si
vous ne les détrompez
pas. Vous pourrez alors tout
vouloir, vous aurez le pied
partout. Vous saurez alors
ce qu'est le monde, une
réunion de dupes et de
entrer dans ce labyrinthe. Ne
le compromettez pas.

link _ID103

_ID162
_ID161
_ID160
_ID157
_ID152

_ID122
_ID121
_ID119
_ID117
"u48" rendez-le llmoi blanc
_ID116
"u47" lldit-elle en recourbant son cou et
"u46" Ne le compromettez pas.
_ID118
"u45" pourllentrer dans ce labyrinthe
"u44" llJe vous donne mon nom comme un fil
_ID120
"u43" Ne soyez ni parmi les uns ni parmi les autres
"u42" Vous saurez alorsllce qu'est le monde, une re
"u41" vous aurez le pied partout
u40" Vous pourrez alors toutllvouloir,
25

Figure 2: A GLOSS session with two documents opened

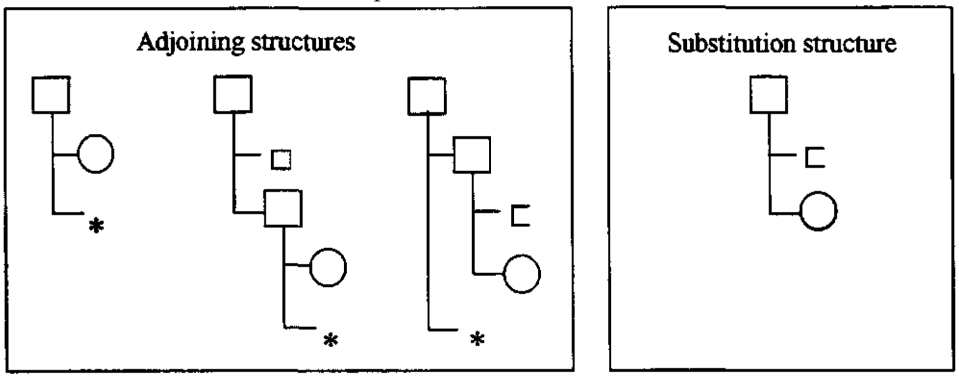


Figure 3: Floating window with adjoining and substitution structures

The discourse structure annotation workspace

Annotating the discourse structure in GLOSS is an interactive visual process that aims at creating a binary tree (Marcu, 1996) (Cristea & Webber, 1997), where intermediate nodes are relations and terminal nodes are units. The discourse structure window appears near the main document window, as shown in figure 2.

An incremental, unit by unit elaboration, that would precisely mimic an automatic expectation-driven incremental tree development, as described in Cristea & Webber (1997), is not compulsory during a manual process. Manual annotation resembles more a trial and error, islands driven process. To facilitate the tree structure building, GLOSS allows development of partial trees that could subsequently be integrated into existing structures. Each unit or partial structure is incorporated within one of the already existing trees either by an adjoining operation (adding to a developing structure an auxiliary tree - one that has a foot node, marked by a *, on the left extreme of the terminal frontier) or a substitution operation (replacement of a substitution node, marked by a , belonging to the developing structure, with a whole tree) (Cristea & Webber, 1997). Adjoining of a partial tree (minimum a unit) into an already existing, developing tree is allowed only on the nodes of the outer or inner right frontier[2], as defined in Cristea & Webber (1997). The approach in Cristea & Webber (1997) that restricts substitution only into the most inner substitution node is also preserved.

Both build operations (adjoining and substitution) require two arguments: the inserted structure and the node of the final structure where the insertion takes place. The adjoining operation is a drag-and-drop one from root of the inserted partial structure to the node of an existing structure where the insertion must take place. In the same manner, a substitution operation is performed between an elementary tree structure and a destination substitution node belonging to the partial developed structure. If instead of dragging, the source node is right clicked, a float appears displaying a family of structures (auxiliary or elementary trees) that could be created using as material the selected node. These structures are shown in figure 3.

After completion of any adjoining or substitution operation the obtained tree obeys the Principle of Sequentiality[3] (Cristea & Webber, 1997).

[2] An *outer right frontier* is the right frontier (the nodes on the path from the root to the most right terminal node) - in a tree without substitution sides. An *inner right frontier* is the right frontier of the tree rooted at the left sister of the most inner substitution site - in a tree with substitution sites.

Remaking the discourse tree

As discourse annotation is an ever-refining process, we have implemented a number of remaking operations that would allow for undoing the work already done when building trees: cuts of father-child links and complete dismembering of parts of the developing tree.

These operations are:

- **deleteUpper:** deletes the upper link of a node. As a result the tree under the node becomes unbound and the corresponding link from the parent remains pending and is marked by a <i because its further behaviour will be that of a substitution node;
- **dismemberTree:** forgets all the relations and the parent-child links of the structure under the selected node.

3. Applying GLOSS for validating the Veins Theory

In (Cristea, Ide, Romary, 1998b), a theory of global discourse called Veins Theory (VT) was proposed. VT identifies hidden structures in the discourse tree, called *veins,* that enable, on one hand, to define referential accessibility domains for each discourse unit and, on the other, to apply Centering Theory (CT) of Grosz, Joshi & Weinstein (1995), globally, to the whole discourse. Veins are defined as sub-sequences of the sequence of units making up the discourse. From the point of view of discourse cohesion, VT claims that referential strings belonging to a unit are allowed to refer only discourse antecedents placed on the same vein as the unit itself, or if this is not the case, then the referents can be inferred from the general or contextual knowledge. The point on coherence defended by VT is that if a different metric than the surface order of the units is used, than the inference load the discourse obliges to is less. The new metric

[3] A left-to-right reading of the terminal frontier of the tree corresponds to the span of text scanned in the same left-to-right order.

being along veins, the comparison criteria is a "smoothness" index computed on the base of CT transitions (continuation is smoother than retaining, which is smoother than smooth shift, which is smoother than abrupt shift, which itself is smoother than no transition at all). Accordingly, values from 4 - continuation, down to 0 - no transition, are summed up for all discourse units and the result is divided by the number of transitions. The research aims at proving that in all cases the VT smoothness score is at least as good as the one computed following CT, if not better.

The specific scheme proposed is a multi-level (hierarchical) parallel annotation architecture described in detail in Cristea, Ide & Romary (1998a). The overall hierarchical architecture of this schema is a DAG of views as in figure 4.

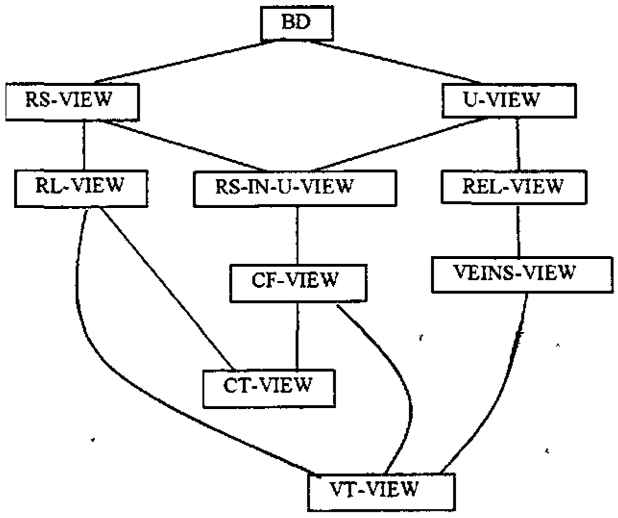


Figure 4: The hierarchy of views for the validation of VT

Following is a short description of each of these views:

- BD: the base document, contains the raw text or, possibly, markup for basic document structure down to the level of paragraph;
- RS-VIEW: includes markup for isolated reference strings. The basic elements are <RS>'S (reference strings);
- RL-VIEW: the reference links view, imposed over the RS-VIEW, includes reference links between an anaphor, or source, and a referee, or target. Links configure co-reference chains, but can also indicate bridge references (Strube & Hahn, 1996; Passoneau, 1994, 1996). They are coded as <LINK> elements with TYPE=CO-REFERENCE or TYPE=BRIDGE;

U-VIEW: marks discourse units (sentences, and possibly clauses). Units are marked as <SEG> elements with TYPE=UNTT;

REL-V1EW: reflects the discourse structure in terms of a tree-like representation. To build the REL-VIEW, full usage of the interactive tree building facilities is made. The parent-child relations are marked as <LINK TYPE=RELATION> with the attributes TARGET1 (left child) and TARGET2 (right child);

VEINS-VIEW: includes markup for vein expressions. VEIN attributes (with values comprising lists) are added to all <SEG TYPE=UNIT> elements; ;„

RS-IN-U-VIEW: inherits <RS> and <SEG TYPE=UNIT> elements from U-VIEW and RS-VIEW. It also includes markup that identifies the discourse unit to which a referring string belongs;

CF-VIEW: inherits all markup from RS-IN-U-VIEW, and adds a list of forward looking centers (the CF attribute) to each unit in the discourse;

CT-VIEW (Centering Theory view): inherits the CF attribute from the CF-VIEW and backward references from the RL-VIEW. Using the markup in this view, first backward-looking centers[4] (the CB-C[5] attribute of the <SEG TYPE=UNIT> elements) and then transitions can be computed following classical CT, therefore between sequential units. A global smoothness score following CT is finally added;

VT-VIEW *(Veins Theory view):* inherits forward-looking lists from the CF-VIEW, back-references from the RL-VIEW, and vein expressions from the VEINS-VIEW. The VT-VIEW also includes markup for backward-looking centers computed along the veins of the discourse structure. As, for each unit these centers depend on the vein the current unit is actually placed, they are recorded as CB-H[6] attribute of the <LINK> elements. Transitions are computed following VT and then, a global VT smoothness score, thus providing support for the validation of the claim on coherence uttered by VT. Also the simultaneous

[4] CT defines a backward-looking center as the first element of the forward-looking list of the previous unit that is realised also in the current unit.

[5] From "classical".

[6] From "hierarchical".

existence of reference links, inherited from RL-VTEW and of vein expressions, inherited from VEINS-VIEW allows for the validation of the claim on cohesion of VT (references are possible only in their domains of accessibility).

Conclusions

We present an annotation tool, called GLOSS, that manifests the following features: accepts as input SGML source documents and/or their database images and produces as output source SGML documents as well as their associated database images; allows for simultaneous opening of more documents; can mix independent annotation views of the same original document, which also allows for a layer-by-layer annotation process in different annotation sessions and by different annotators, including automatic; offers an attractive interface to the user; permits discourse structure annotation by offering a pair of building operations (adjoining and substitution) and remaking operations (delete parent-child link and tree dismember). Finally we display an example that shows how GLOSS is used to validate, by partial manual, partial automatic corpus annotation, a theory of global discourse. The system currently runs on a PC under Windows'95orNT.

Although designed in the idea of assisting annotation tasks in discourse, GLOSS could also be used for other types of annotation. The use of SGML with an associated DTD file allows easy definition of any annotation standard with specific markings and their families of attributes. To assist discourse structure coding, the current version implements binary trees with a range of operations that permit interactive building of trees in an add/modify/delete manner. This module could be extended to allow for interactive annotation of any kind of trees. As such, syntactic structures become possible to be marked in a GLOSS session.

As further extensions to GLOSS, we plan to add an interface that will allow easy definition and incorporation of other structures, different from trees.

References

L. Bruneseaux & L. Romary (1997). Codage des references et conferences dans les dialogues homme-machine. *Proceedings of ACH/ALLC*, Kingston (Ontario).

D. Cristea, N. Ide & L. Romary. (1998a). Marking-up multiple views of a Text: Discourse and Reference. Proceedings of the First International Conference on Language Resources Evaluation, Granada, 28-30 May.

D. Cristea, N. Ide & L. Romary. (1998b). Veins theory: A Model of Global Discourse Cohesion and Coherence. Proceedings of ACL/COLING'98.

D. Cristea & B. L. Webber. (1997). Expectations in Incremental Discourse Processing. *Proceedings of ACL-EACL'97.*

W. Daelemans; K. De Smedt & G. Gazdar (1992). Inheritence in Natural Language Processing. *Computational Linguistics*, vol. 18, no. 2.

N. Ide & G. Priest-Dorman. (1996). Corpus Encoding Specification. http://www.cs.vassar.edu/CES/.

B. J. Grosz, A. K. Joshi & S. Weinstein. (1995). Centering: A framework for modelling the local coherence of discourse. *Computational Linguistics*, 12(2), June.

D. Marcu (1996). Bulling up rhetorical structure trees, *Proceedings of the 13th National Conference on Al (AAAI-96)*, vol.2, Portland, Ore., 1069-1074.

C. M. Sperberg-McQueen & L. Burnard. (1994). Guidelines For Electronic Text Encoding and Interchange. ACH-ACL-ALLC Text Encoding Initiative, Chicago and Oxford.

Linguistic Annotation of Two Prosodic Databases

Maria Wolters
Institut fur Kommunikationsforschung und Phonetik
Poppelsdorfer Allee 47, D-53115 Bonn
wolters@ikp.uni-bonn.de

Abstract

Two prosodic databases were annotated with linguistic information using SGML (Standard General Markup Language), one database of American English and one of Modern Standard German. Only information that might have prosodic correlates was annotated. Phonetic and morphological information was supplied by automatic tools and then hand corrected. Semantic and pragmatic information was inserted by hand. The SGML tagset is essentially the same for both languages. Tags delimit structural units, all other information is supplied by attributes. The databases themselves, which combine linguistic and phonetic information, are stored as SPSS files.

1 Introduction

The prosody of an utterance is closely related to its linguistic form. Therefore, a prosodic database for a language should contain both phonetic/prosodic and linguistic information. This paper describes how linguistic information is encoded in the Bonn prosodic databases of German (Heuft et al., 1995), henceforth G, and American English (Eisner et al., 1998), henceforth AE. The databases were read by 2-3 native speakers under laboratory conditions. Each speaker read the complete text. AE consists of 443 question/answer dialogues, while G contains 110 question/answer dialogues, 3 short texts and 116 single sentences. Most of the material was written specifically for these corpora. The phonetic information in these corpora is rather detailed for a speech corpus of that size, which permits very fine-grained analyses.

The databases are used to study how information such as dialogue act, sentence type or focus can be signalled prosodically in content-to-speech synthesis (CTS). In CTS, the input is enriched with information about semantic units and pragmatic functions. For our analyses, we mainly use the statistics package SPSS. Since each segment and each word is stored as a separate case in the SPSS file, word-level linguistic annotations are difficult to maintain in this format. Therefore, they are stored in a SGML-based markup of the text. SGML (Standard General Markup Language, (Goldfarb, 1990)) was chosen because it is a standard language for corpus annotation. In an SGML annotation, the text is partitioned into elements E, which are organized hierarchically. The boundaries of these elements are marked by start (<E>) and end (</E>) tags. Each tag can have an arbitrary number of attributes (<E attr=value>). The main disadvantage of SGML is the strict hierarchical structure it imposes - annotating units that partly overlap in a single document is impossible.

Combining the two annotations is straightforward. The parsed SGML file with full attribute specifications for each tag is converted into a tab-delimited database of words where each tag and each attribute has its own column. Tag columns have two values, 0 for closed tag and 1 for open tag. Attribute columns have the value -1 if the tag for which the attribute is defined is closed, and the correct value if that tag is open. In a last step, the information for each word is added to its entry in the SPSS file.

In the next sections, we present the hierarchical structure of the SGML tagset and give an outline of its four layers. It is beyond the scope of this paper to list all tags and their attributes; this information can be found in (Wolters, 1998). To conclude, we will discuss extension and evaluation of the coding scheme.

2 Overview of the Tagset

Both corpora share a basic tagset. The attributes for identifying sound files, part-of-speech labels, and phrase-level tags differ. All phonemic, morphological and syntactic information which was inserted automatically was subsequently corrected by hand.

The tags specify units on four different layers, structural layer, word layer, phrase layer and semantic/pragmatic layer. The properties of these units are encoded in attributes. The first layer specifices the overall structure of the database. Information annotated in the second and third layer such as part of speech (POS) and syntactic phrase boundaries is relevant for text-to-speech (TTS) as well as for CTS systems, because it can be provided automatically. In contrast, the annotations on the fourth layer contain the information that is only exploited in CTS systems.

The third and the fourth layer are largely orthogonal: semantic and pragmatic units are both parts of larger syntactic phrases and contain smaller syntactic phrases. If a semantic/pragmatic unit is coextensive with a syntactic phrase, the phrase boundaries are contained within the unit boundaries.

3 The Structural Layer

There are four main elements on this layer. The tag <dialbase> indicates that the text is a prosodic database. Its units are dialogues (tag: <dialog>), stories (tag: <story>), and sentences (tag: <sent>).[1] Each unit has a unique identifier.

Stories and dialogues consist in turn of sentences. Sentences are sequences of words ended by a dot, a question mark or an exclamation mark. Sentences are marked for sentence type (attribute typ) and, in dialogues, for the current speaker (attribute sp). Table 1 lists the six types covered.

4 The Word Layer

The main unit on this layer is the word (<w>). The attribute pos supplies part-of-speech (POS) information, while the tags <phon> and <orth> delimit the canonical phonemic and orthographic transcription of a word. Example:[2]

(1) <w pos=VB> <phon> g II v <orth> give

Since POS tags are comparatively easy to supply in a text-to-speech synthesis system, they provide valuable information for a first set of prosodic rules (Widera et al., 1997). Including canonical transcriptions allows to check for reductions, elisions and insertions which cooccur with prosodic parameters such as stress and speaking rate.

The canonical transcriptions for AE were retrieved from cmudict[3] and converted to SAMPA for American English (Wolters, 1997); G was transcribed by a rule set implemented in the language P-TRA (Stock, 1992). Each vowel symbol is followed by an indication of stress type, primary (1), secondary (2) or none (0). The AE POS tags were provided by the Brill tagger (Brill, 1993), which uses the Penn Treebank tagset (Santorini, 1990). The POS tags for G were derived from the Bonner Wortdatenbank (Brustkern, 1992).

C	command	ST	statement
DEQ	decision question	SEQ	selection question
EQ	echo question	WQ	WH-question

Table 1: Sentence types. The answers to questions are labelled with the label of the question + a for answer. Echo questions are defined as declarative sentences with a question intonation, the definitions of the other types are straightforward.

5 The Phrase Layer

There are two types of units, syntactic phrases and lists (tag: <li>). Lists are frequently used in applications of text-to-speech synthesis. They consist of items (tag: <1>) and coordinators (tag: <cc>). Example:[4]

(2) <sent typ=deq>
Sind <li> <1> Peter </!> <1> Tina </!> <1> Petra </!> <cc> und </cc> <1> Klaus </!> </li> wirklich

[1]Turns are not indicated by separate tags, *as* they would have to be for spontaneous speech, because turn taking is almost impossible to study using read speech.

[2] If not stated otherwise, all examples come from one of the databases.

[3]compiled by Bob Weide, available at ftp://svr-ftp.eng.cam.ac.uk/pub/comp-speech/dictionaries

[4] All examples only show the relevant tags.

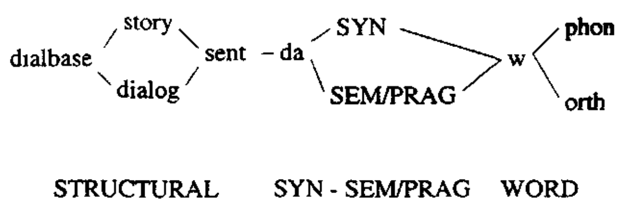


Figure 1- Hierarchy and layering of SGML elements CAPS: name of layer. Only the most important tags are shown.

miteinanderverwandt?
trans Are Peter, Tina, Petra and Klaus really relatives?

Syntactic phrase boundaries were provided by the Apple Pie Parser (Sekine, 1995) for AE and the SVOX (Traber, 1995) parser for G. These tags are used for examining if and how syntactic boundaries are marked prosodically. There are four phrase types, noun phrases (tag: <np>), verb phrases (tag: <vp>), adverb phrases (tag: <advp>), and adjective phrases (tag: <adjp>). An example from the German database:

(3) <sent typ=c>
<advp st=k f=u>
<w pos=QPN> <phon> 1 aUl t 60 <orth> lauter
<w pos=INJ> <phon> b II t ©0 <orth> bitte
</advp>
trans.. "Speak up, please".
Attributes of the adverbial phrase: st=k - comparative, f=u - not inflected

When a parser provides more detailed information about a phrase than just the type of its head, this is stored in attributes. These attributes mostly refer to features of the phrase head. Their default value is "unspecified", in case the parser does not yield this information.

6 Semantic/Pragmatic Level

This level differs from the others in that most of the concepts to be annotated are difficult both to define clearly and and to detect automatically. Therefore, all annotations have to be inserted by hand.

Date, time, numbers. These units are used for current research in the VERBMOBIL project. Dates (<d>) must consist at least of day and month, and times (<t>) at least of hours. Numbers (<nr>) are sequences of digits which refer to e.g. a telephone number. All three units consist of noun phrases at the syntactic level. Some examples:

(4) <sent typ=deq> War seine Telefonnummer nicht <nr> eins vier drei drei zwo </nr>
trans.: Wasn't his phone number one four three three two?

(5) <sent typ=deq> <d> Am siebenundzwanzigsten Mai </d> <t> um siebzehn Uhr fünfunddreißig </t> , kann ich das eintragen?
trans.: The 27th of May at 17.35, can I note that down?

Coreference. It is well known that old information is less likely to be accented than new information. But how can this dimension of familiarity be quantized for annotation? To avoid semantic complications, we only code the familiarity of discourse referents. First, all referring expressions are marked with the tag <coref> following (MUC, 1996). Referring expressions can consist of syntactic noun phrases and of other referring expressions. Each expression is assigned an integer number id unique within a dialogue or story. The attribute ref specifies the entity an expression refers to. A referent is identified by the cardinal number of the referring expression which introduces it. The attribute ground specifies how the referent is linked to the preceding discourse. The links, summarized in Tab. 2, are a subset of those proposed in (Passonneau, 1997). A constructed example:

(6) Does <coref id=l ref=l ground=no> Mimi </coref> like <coref id=2 ref=2 ground=no> detective stories </coref>

no	no link, first extensional mention in the discourse
id	referential identity
isa	member of a previously mentioned set
subs	subset of a previously mentioned set
part	part of a previously mentioned whole

Table 2: Types of links between referring expressions.

> Yes, <coref id=3 ref=l ground=yes> she </coref> especially likes <coref id=4 *Tef—2* ground=subs> old-fashioned whodunits </coref> .

Focus. Five types of foci are annotated, adverbial, answer, contrast, correction, and information foci. The first four types are common, rather descriptive categories, while the last type relies on the focus theory of (Vallduvi, 1992). It was included in order to account for foci which are not covered by the other categories. Although they could have been annotated in terms of Vallduvi's theory, more theory-independent definitions were used in order to facilitate a re-analysis in an arbitrary framework.

The definitions of all five types is purely semantic. What our tags identify is *potential* focus positions, which allows us to examine in a second step which of these foci were realized prosodically. Thereby, we avoid circularity. The first four focus types are tagged using f, with the attribute typ for focus type. Focus "anchors" are specified by the tag (<fa>). An anchor is a constituent in the preceding discourse which the focus refers to. For example, the position of the answer focus in wh-questions depends on the information which is requested, and the position of the adverbial focus depends on the focus adverb, more precisely on the scope of the focus adverb. Although anchors are not likely to be accented, it might be interesting to examine in which conditions they do receive an accent. Furthermore, specifying the anchor also facilitates constructing a semantic representation for the focussed consituent. The attribute ref links foci and their anchors as well as the elements of contrast and correction foci. Tab. 3 gives short definitions of these focus types and their anchors. Some constructed examples:

(7) Does Anne prefer <fa typ=a ref=l> tea </fa> or <fa typ=a ref=l> coffee </fa> ? She prefers <f typ=a ref=l> tea. </f> (answer focus)

(8) <f typ=cn ref=2> Anne </f> likes <f typ=cn ref=3> tea </f>, <f typ=cn ref=2> Ben </f> likes <f typ=cn ref=3> coffee </f> . (contrast focus)

(9) Ben doesn't like <fa typ=cr ref=4> tea </fa>, he prefers <f typ=cr ref=4> coffee </f>. (correction focus)

(10) But sometimes, <fa typ=v ref=5> even </fa> <f typ=v ref=5> Ben </f> drinks tea. (adverb focus)

The fifth type of focus is intended to cover foci which do not belong to one of the first four types. We assume with (Heim, 1983) that there is a file card for each discourse referent which serves as a repository of information about that referent. The constituents of an utterance which are *in focus* specify information which is to be added to a referent's file card. This concept of focus is based on (Vallduvi, 1992). To distinguish this general type of focus from the other, more specific ones, which it can be extended to include, it is labelled *information focus,* tag: **<if>.** There are no anchors for information foci, since they are defined in relation to abstract file cards. The attribute update specifies how the file card to be updated is retrieved. To avoid a proliferation of tags, information focus is only annotated when none of the other focus types is present.

(11) Ben doesn't like chocolate.
He <if update=p> doesn't drink milk, </if> either, (information focus, update via pronominal reference).

Dialogue Act. Dialogue acts are difficult to elicit in read speech, because they depend on the aim a speaker has in mind when producing a certain utterance. When labelling dialogue, this aim has to be reconstructed from the context. For example, the utterance "that won't work" can be assigned three different dialogue acts:

(12) A: Turn the screw now. B: That won't work, *negative feedback*

(13) A: Can we meet on Saturday. B: That won't work, *rejection of suggestion*

L

Type	Focus	Anchor
adverb	scope of adverb	adverb
correction	correction	consitutent to be corrected
contrast	constituents to be contrasted	not applicable
answer	*wh-question:*	
	constituent corresponding to wh-phrase	wh-constituent
	selection question:	
	member of alternative set	alternative set from question

Table 3. Focus types and their anchors.

feedback	positive	positive answer to question; previous utterance not accepted/understood, FP
	accept	accept suggestion, ACC
	negative	negative answer to question, previous utterance not accepted/understood, FN
	reject	reject suggestion, REJ
inform		give information (default), INF
	give reason	give reason for sth. recently mentioned, GIR
request	information	ask for information, REQI
	action	request action, REQA
	suggest	request suggestion, REQS
suggest		suggest

Table 4: Types of dialogue acts, **bold type:** main type, underlined: subtype, *italics:* subsubtype. CAPS: value of attribute typ

editors, *Meaning, use, and interpretation of language,* pages 164-189. de Gruyter, Berlin.

B. Heuft, T. Portele, J. Kramer, H. Meyer, M. Rauth, and G. Sonntag. 1995. Parametric description of FO contours in a prosodic database. In *Proc. Int. Congress of Phon. Sci. Stockholm,* pages 378-381.

MUC. 1996. Conference task definition V. 3.0. Message Understanding Conference.

R. Passonneau. 1997. Summary of the coreference group. In J. Carletta, N. Dahlback, N. Reithinger, and M.A. Walter, editors, *Standards for Dialoge Coding in Natural Language Processing,* pages 12-21. http://www.dag.uni-sb.de/ENG.

B. Santorini. 1990. Part-of-speech tagging guidelines for the Penn Treebank Project. Technical report, University of Pennsylvania.

S. Sekine. 1995. Apple Pie Parser. Technical report, Department of Computer Science, New York University, http://cs.nyu.edu/cs/projects/proteus/app/. .

D. Stock. 1992. P-TRA - eine Programmiersprache zur phonetischen Transkription. In W. Hess and W.F. Sendlmeier, editors, *Beiträge zur angewandten und experimentellen Phonetik,* pages 222-231. Steiner, Stuttgart.

C. Traber. 1995. *SVOX: The Implementation of a Text-to-Speech Sytem for German.* Ph.D. thesis, Institut für Technische Informatik und Kommunikationsnetze, ETH Zurich.

E. Vallduvi'. 1992. *The Informational Component.* Garland, New York.

C. Widera, T. Portele, and M. Wolters. 1997. Prediction of word prominence. In *Proc. Eurospeech,* pages 999-1002. Rhodes.

M. Wolters. 1997. A multiphone unit inventory for the concatenative synthesis of American English. VERBMOBIL Memo 120.

M. Wolters. 1998. Konventionen für linguistisches Tagging. Technical report, Institut für Kommunikationsforschung und Phonetik, Universitat Bonn, ftp://asll.ikp.uni-bonn.de/pub/mwo/dbaseanno.ps.gz.

A Linguistically Interpreted Corpus of German Newspaper Text

Wojciech Skut, Thorsten Brants, Brigitte Krenn, Hans Uszkoreit
Universität des Saarlandes, Computational Linguistics
D-66041 Saarbrücken, Germany
{skut,brants,krenn,Uszkoreit}@coli.uni-sb.de

Abstract

In this paper, we report on the development of an annotation scheme an annotation tools for unrestricted German text. Our representation format is based on argument structure, but also permits the extraction of other kinds of representations. We discuss several methodological issues and the analysis of some phenomena. Additional focus is on the tools developed in our project and their applications.

1 Introduction

Parts of a German newspaper corpus, the Frankfurter Rundschau, have been annotated with syntactic structure. The raw text has been taken from the multilingual CD-ROM which has been produced by the European Coding Initiative ECI, and is distributed by the Linguistic Data Consortium LDC.

The aim is to create a linguistically interpreted text corpus, thus setting up a basis for corpus linguistic research and statistics-based approaches for German. We developed tools to facilitate annotations. These tools are easily adaptable to other annotation schemes.

2 Corpora for Data-Driven NLP

An important pardigm shift is currently taking place in linguistics and language technology. Purely introspective research focussing on a limited number of isolated phenomena is being replaced by a more data-driven view of language. The growing importance of stochastic methods opens new avenues for dealing with the wealth of phenomena found in real texts, especially phenomena requiring a model of preferences or degrees of grammaticality.

This new research paradigm requires very large corpora annotated with different kinds of linguistic information. Since the main objective here is rich, transparent and consistent annotation rather than putting forward hypotheses or explanatory claims, the following requirements are often stressed:

descriptivity: phenomena should be described rather than explained as explanatory mechanisms can be derived (induced) from the data.

data-drivenuess: the formalism should provide means for representing all types of grammatical constructions occurring in the corpus[1].

theory-neutrality: the annotation format should not be influenced by theory-internal considerations. However, annotations should contain enough information to permit the extraction of theory-specific representations.

In addition, the architecture of the annotation scheme should make it easy to refine the information encoded, both in width (adding new description levels) and depth (refining existing representations). Thus a structured, multi-stratal organisation of the representation formalism is desirable.

The representations themselves have to be easy to determine on the basis of simple empirical tests, which is crucial for the consistency and a reasonable speed of annotation.

3 Why Tectogrammatical Structure?

In the data-driven approach, the choice of a particular representation formalism is an engineering problem rather than a matter of 'adequacy'. More important is the theory-independence and reusability of linguistic knowledge, i.e., the recoverability of theory/application specific representations, which in the area of NL syntax fall into two classes:

Phenogrammatical structure: the structure reflecting surface order, e.g. *constituent structure* or topological models of surface syntax, cf. (Ahrenberg, 1990), (Reape, 1994).

Tectogrammatical representations: predicate-argument structures reflecting lexical argument structure and providing a guide for assembling

'This is a revised version of the paper (Skut et al., 1997a).
tThe work has been carried out in the project NEGRA of the Sonderforschungsbereich 378 'Kognitive ressourcenadaptive Prozesse' (resource adaptive cognitive processes) funded by the Deutsche Forschungsgemeinschaft.

[1]This is what distinguishes corpora used for grammar induction from other collections of language data. For instance, so-called *test suites* (cf. (Lehmann et al., 1996)) consist of typical instances of selected phenomena and thus focus on a subset of real-world language.

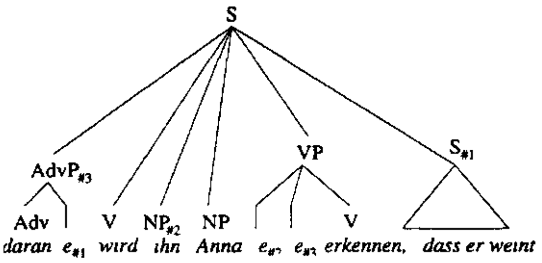
meanings. This level is present in almost every theory: D-structure (GB), f-structure (LFG) or argument structure (HPSG). A theory based mainly on tectogrammatical notions is dependency grammar, cf. (Tesniere, 1959).

As annotating both structures separately presents substantial effort, it is better to recover constituent structure automatically from an argument structure treebank, or vice versa. Both alternatives are discussed in the following sections.

3.1 Annotating Constituent Structure

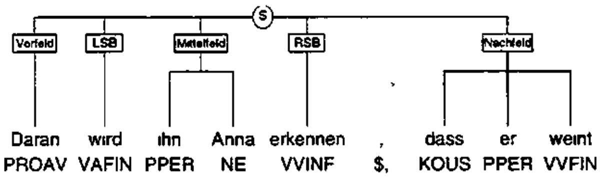
Phenogrammatical annotations require an additional mechanism encoding tectogrammatical structure, e.g., trace-filler dependencies representing discontinuous constituents in a context-free constituent structure (cf. (Marcus, Santorini, and Marcinkiewicz, 1994), (Sampson, 1995)). A major drawback for annotation is that such a hybrid formalism renders the structure less transparent, as is the phrase-structure representation of sentence (1):

(1) daran wird ihn Anna erkennen, dass er weint
at-it will him Anna recognise that he cries
'Anna will recognise him at his cry'



Furthermore, the descriptivity requirement could be difficult to meet since constituency has been used as an explanatory device for several phenomena (binding, quantifier scope, focus projection).

The above remarks carry over to other models of *phenogrammatical structure,* e.g. *topological fields,* cf. (Bech, 1955). A sample structure is given below[2]


S
Vorfeld
LSB
Mittelfeld
RSB
Nachfeld
Daran wird ihn Anna erkennen , dass er weint
PROAV VAFIN PPER NE VVINF $, KOUS PPER VVFIN

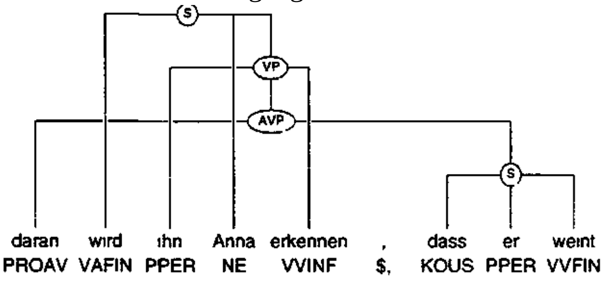
Here, as well, topological information is insufficient to express the underlying tectogrammatical structure (e.g., the attachment of the extraposed that-clause)[3]. Thus the *field model* can be viewed as a non-standard phrase-structure grammar which needs additional tectogrammatical annotations.

[2]LSB, RSB stand for *left* and *right sentence bracket.*

[3]Even annotating grammatical functions is not enough as long as we do not explicitly encode their tectogrammatical attachment of such functions.

3.2 Argument Structure Annotations

An alternative to annotating surface structure is to directly specify the tectogrammatical structure, as shown in the following figure:


S
VP
AVP
S
daran wird ihn Anna erkennen , dass er weint
PROAV VAFIN PPER NE VVINF $, KOUS PPER VVFIN

This encoding has several advantages. Local and non-local dependencies are represented in a uniform way. Discontinuity does not influence the hierarchical structure, so the latter can be determined on the basis of lexical subcategorisation requirements, agreement and some semantic information.

An important advantage of tectogrammatical structure is its proximity to semantics. This kind of representations is also more theory-neutral since most differences between syntactic theories occur at the phenogrammatical level, the tectogrammatical structures being fairly similar.

Furthermore, a constituent tree can be recovered from a tectogrammatical structure. Thus tectogrammatical representations provide a uniform encoding of information for which otherwise both constituent trees *and* trace-filler annotations are needed.

Apart from the work reported in this paper, tectogrammatical annotations have been successfully used in the TSNLP project to construct a language competence database, cf. (Lehmann et al., 1996).

3.3 Suitability for German

Further advantages of tectogrammatical annotations have to do with the fairly weak constraints on German word order, resulting in a good deal of discontinuous constituency. This feature makes it difficult to come up with a precise notion of constituent structure. In the effect, different kinds of structures are proposed for German, the criteria being often theory-internal[4].

In addition, phrase-structure annotations augmented with the many trace-filler co-references would lack the transparency desirable for ensuring the consistency of annotation.

[4]Flat or binary right-recursive structures, not to mention the status of the head in verb-initial, verb-second and verb-final clauses, cf. (Netter, 1992), (Kasper, 1994), (Nerbonne, 1994), (Pollard, 1996).

4 Methodology

The standard methodology of determining constituent structure (e.g., the *Vorfeld* test) does not carry over to tectogrammatical representations, at least not in all its aspects. The following sections are thus concerned with methodological issues.

4.1 Structures vs. Labels

The first question to be answered here is how much information has to be encoded structurally. Rich structures usually introduce high spurious ambiguity potential, while flat representations (e.g., category or function labels) are significantly easier to manipulate (alteration, refinement, etc.).

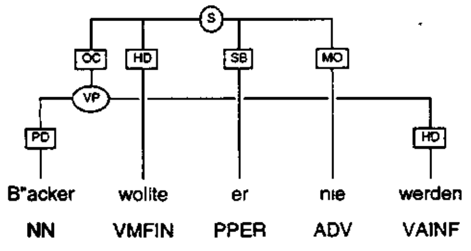
Thus it is a good strategy to use rather simple structures and express more information by labels.

4.2 Structural Representations

As already mentioned, tectogrammatical structures are often thought of in terms of *dependency grammar* (DG, cf. (Hudson, 1984), (Hellwig, 1988)), which might suggest using conventional *dependency trees* (stemmas) as our representation format. However, this would impose a number of restrictions that follow from the theoretical assumptions of DG. It is mainly the DG notion of heads that creates problems for a flexible and maximally theory-neutral approach. In a conventional dependency tree, heads have to be unique, present and of lexical status, requirements other theories might not agree with.

That is why we prefer a representation format in which heads are distinguished outside the structural component, as shown in the figure below, sentence (2)[5]:

(2) Bäcker wollte er nie werden
baker wanted he never become
'he never wanted to become a baker'



The tree encodes three kinds of information:

tectogrammatical structure: trees with possibly crossing branches (no non-tangling condition);

syntactic category: node labels and part-of-speech tags (Stuttgart-Tubingen Tagset, cf. (Thielen and Schiller, 1995)).

functional annotations: edge labels.

4.3 Classification of Labels

Compared to the fairly simple structures employed by our annotation scheme, the functional annotations encode a great deal of linguistic information. We have already stressed that the notion *head* is distinguished at this level. Accordingly, it seems to be the appropriate stratum to encode the differences between different classes of dependencies.

For instance, most linguistic theories distinguish between complements and adjuncts. Unfortunately, the theories do not agree on the criteria for drawing the line between the two classes of dependents. To this date there is no single combination of criteria such as category, morphological marking, optionality, uniqueness of role filling, thematic role or semantic properties that can be turned into a transparent operational distinction linguists of different schools would subscribe to.

In our scheme, we try to stay away from a theoretical commitment concerning borderline decisions. The distinction between functional labels such as SB and DA - standing for traditional grammatical functions - on the one hand and phrases labelled MO on the other should not be interpreted as a classification into complements and adjuncts. For the time being, functional labels different from MO are assigned only if the grammatical function of the phrase can easily be detected on the basis of the linguistic data. MO is used, e.g., to label adjuncts as well as prepositional objects. Likewise the label OC is used for easily recognisable clausal complements. Other embedded sentences depending on the verb are labelled as MO[6]. This is consistent with our philosophy of stepwise refinement. We are in the process of designing a more fine-grained classification of functional labels together with testable criteria for assigning them. This classification will not contain a distinction between complements and adjuncts. Thus the locative phrase *m Berlin* in the sentence *Peter wohnt in Berlin* (Peter lives in Berlin) will just be marked as a locative MO with the category PP. As linguistic theories disagree on the question, we will not ask the annotators to decide whether this phrase is a complement of the verb.

This strategy differs from the one pursued by the creators of the Penn Treebank. There the difference between complements and adjuncts is encoded in the

[5]Edge labels- HD head, SB subject, OC clausal complement, PD predicative, MO modifier. Note that crossing edges indicate discontinuous constituency.

[6]MO is inspired by the usage of the term 'modifier' in traditional structuralist linguistics where some authors (Bloomfield, 1933) use it for adjuncts and others also for complements (Trubetzkoy, 1939).

hierarchical structure. Verbal complements are encoded as siblings of the verb whereas adjuncts are adjoined at a higher level. In a case of doubt, the annotators are asked to select adjunction. We consider this structural encoding less suitable for refinement than a hierarchy of functional labels in which MO can be further specified by sublabels.

5 Annotation Tools

The development of linguistically interpreted corpora presents a laborious and time-consuming task. In order to make the annotation process more efficient, extra effort has been put into the development of the annotation software.

5.1 Structural Annotation

The annotation tools are an integrated software package that communicates with the user via a comfortable graphical interface (Plaehn, 1998). Both keyboard and mouse input are supported, the structure being annotated is shown on the screen as a tree. The tools can be employed for the annotation of different kinds of structures, ranging from our rudimentary predicate-argument trees to standard phrase structure annotations with trace-filler dependecies, cf. (Marcus, Santorini, and Marcinkiewicz, 1994). A screen dump of the annotation tool is shown in figure 1.

The kernel part of the annotation tool supports purely manual annotation. Further modules permit interaction with an external stochastic or symbolic parser. Thus, the tools are not dependent on a particular automation method. Also the degree of automation can vary from part-of-speech tagging and recognition of grammatical functions to full parsing.

In our project, we rely on an interactive annotation mode in which the annotator specifies rather small annotation increments that are then processed by a stochastic parser. The output of the parser is immediately displayed and the annotator edits it if necessary. Currently, the annotator's task is to specify substructures containing up to 20 — 30 words; their internal structure as well as the labels for grammatical functions and categories are assigned by the parser. The precision of the parser is about 96% for the assignment of labels and 90% for partial structures (Brants and Skut, 1998; Skut and Brants, 1998a; Skut and Brants, 1998b).

Another part of our software package is the corpus search tool. It is very helpful for both linguistic investigations and detecting annotation errors. As for this latter application, we have also developed programs that compare annotations. Each sentence is annotated independently by two annotators. During the comparison, inconsistencies are highlighted, and the annotators have to correct errors and/or agree on one reading.

In addition to the treebank project, the tools are currently used in the Verbmobil project to annotate transliterated spoken dialogues in English and German (Stegmann and Hinrichs, 1998), in the FLAG project to annotate spelling errors in German newsgroup texts, and it is planned to employ them in the DIET project to build a linguistic competence database (Netter et al., 1998).

5.2 Automation

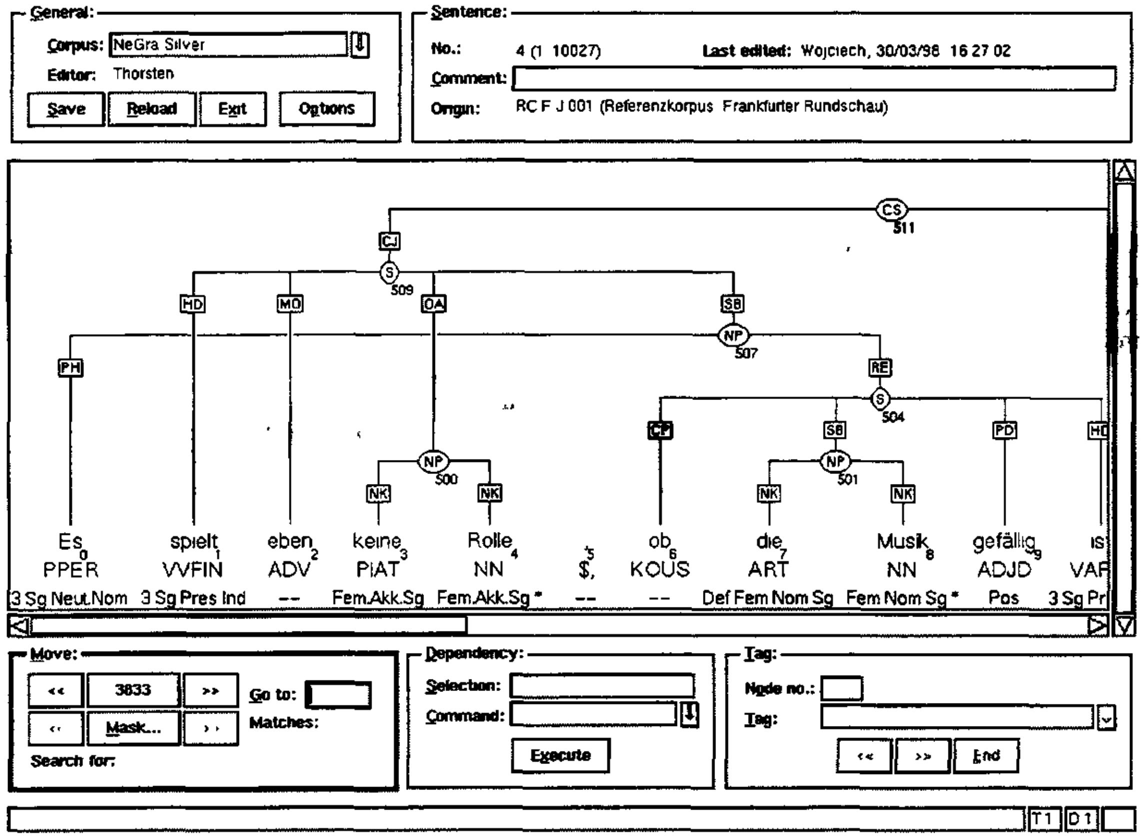
The graphical surface communicates with several separate programs to perform the task of semi-automatic annotation. Currently, these separate programs are a part-of-speech tagger, a tagger for grammatical functions and phrasal categories and an NP/PP chunker.

The part-of-speech tagger is a trigram part-of-speech tagger that is trainable for a wide variety of languages and tagsets (Brants, 1996). We trained it on all previously annotated material in our corpus, using the Stuttgart-Tbingen tagset, and it currently achieves an accuracy of 96% on new, unseen text.

In our project, annotation is an interactive task. After the annotator has specified a partial structure, the tool automatically inserts all the labels into the structure, i.e. the grammatical functions (edge labels) and phrasal categories (node labels). This task is performed by a tagger for grammatical functions and phrasal categories (Brants, Skut, and Krenn, 1997). The underlying mechanism is very similar to part of speech tagging. There, states of a Markov model represent tags, and outputs represent words. For tagging grammatical functions, states represent grammatical functions, and outputs represent terminal and non-terminal tags. Thus, tagging is applied to the next higher level.

Grammatical functions have a different distribution within each type of phrase, so each type of phrase is modeled by a different Markov model. If the type of phrase is known, the corresponding model is used to assign grammatical functions. If the type is not known, all models run in parallel and the model assigning the highest probability is used. This determines at the same time the phrasal category. The tagger is also trained on all previous material of the corpus and achieves 97% accuracy for assigning phrasal categories, and 96% accuracy for assigning grammatical functions.

When tagging for part-of-speech, grammatical functions, and phrasal categories, we additionally calculate the second best assignment and its probability. This is used to estimate the reliability of the first assignment. If the probability of the alternative is close to that of the best assignment, the first choice is regarded as unreliable, wheres it is reliable if the alternative has a much lower probability. Reliable and unreliable are distinguished by a threshold on the distance of the best and second best assing-


General:
Corpus: NeGra Silver
Editor: Thorsten

Sentence:
No.: 4 (1 10027) Last edited: Wojciech, 30/03/98 16 27 02
Comment:
Origin: RC F J 001 (Referenzkorpus Frankfurter Rundschau)

```
Es          spielt        eben  keine       Rolle        ,   ob    die           Musik          gefällig  ist
0           1             2     3           4            5   6     7             8              9
PPER        VVFIN         ADV   PIAT        NN           $,  KOUS  ART           NN             ADJD      VAFIN
3 Sg Neut.Nom 3 Sg Pres Ind --  Fem.Akk.Sg  Fem.Akk.Sg * --  --    Def Fem Nom Sg Fem Nom Sg *  Pos       3 Sg Pr
```

Move: 3833 Go to: Matches: Search for:
Dependency: Selection: Command:
Tag: Node no.: Tag:

Figure 1: Screen dump of the annotation tool

ment. The annotation tool simply inserts all reliable labels and asks the human annotator for confirmation in the unreliable cases.

The next level of automation is concerned with the structure of NPs and PPs which can be fairly complex in German (see figure 2). As shown in (Brants and Skut, 1998), recognition of complete NP/PP structures can also efficiently performed with Markov models, encoding *relative structures*, i.e. stating that a word is attached lower, higher or at the same level as its predecessor. The annotator no longer has to build the structure level by level, but marks the boundaries of NPs and PPs, and the internal structures is generated automatically. This approach has an accuracy of 85 - 90%, depending on the exact task.

6 Applications of the Corpus

The corpus provides training and test material for stochastic approaches to natural language processing. It is also a valuable source of data for theoretical linguistic investigations, especially into the relation of competence grammar and language usage.

6.1 Statistical NLP

As described in section 5, statistical annotation methods have been developed and implemented. In our bootstrapping approach, the accuracy of the models is improved and functionality increases as the annotated corpus grows, thus leading to completely automatic NLP methods. For instance, the *chunk tagger* initially designed to support the annotator is used for the recognition of major phrases in unrestricted text pre-tagged with part-of-speech information (Skut and Brants, 1998a; Skut and Brants, 1998b).

Apart from these applications, the corpus is already used in other projects to train rule-based and statistical taggers and parsers.

6.2 Corpus Linguistic Investigations

The treebank has been successfully used for corpus-linguistic investigations. In this regard, two major classes of applications have arisen so far. Firstly,

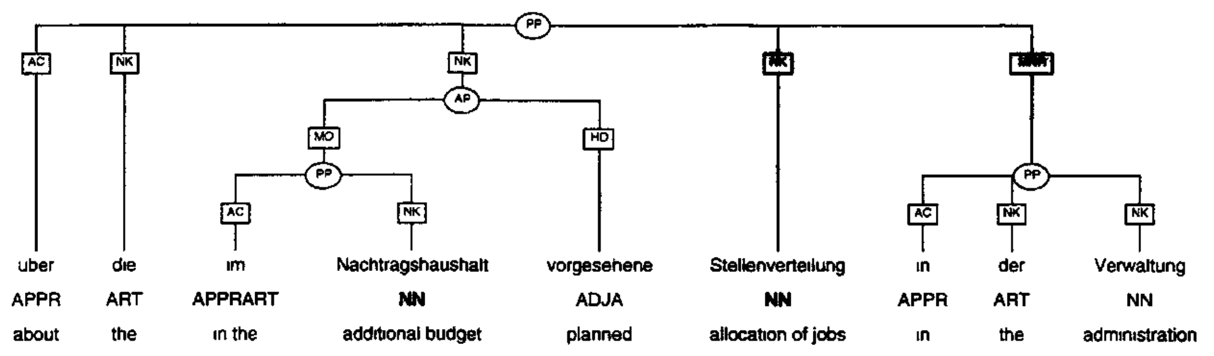


Figure 2: Example of a complex NP.

a search program enables the user to find examples of interesting linguistic constructions, which is especially useful for testing predictions made by linguistic theories. It has also proved to be a great help in teaching linguistics.

The second, more ambitious class of applications consists in statistical evaluation of the corpus data. In a study on relative clause extrapostion in German (Uszkoreit et al., 98), we were able to verify the predictions made by the performance theory of language formulated by Hawkins (1994). The corpus data made it possible to measure the influence of the factors *heaviness* and *distance* on the extraposition of relative clauses. The results of these investigations are also supported by psycholinguistic experiments.

For investigations on statistics-based collocation extraction, various portions of the Frankfurter Rundschau Corpus have been automatically annotated with parts-of-speech and phrase chunks like NP, PP, AP. The part-of-speech tagger (Brants, 1996) and the chunker (Skut and Brants, 1998b) have been trained on the annotated and hand-corrected corpus. Although error rates of 10 to 15 % occur at the stage of chunking, collocation extraction benefits from structurally annotated corpora because of the accessibility of syntactic information (1) accuracy of frequency counts increases, i.e. more syntactically plausible collocation candidates are found, and (2) grammatical restrictions on collocations can mostly be automatically derived from the corpus, cf. (Krenn, 1998b).

Syntactically preprocessed corpora are also a valuable source for insights into actual realisations of collocations. This is particularly important in the case of partially flexible collocations. In order to provide material for investigations into collocations as on the one hand grammatically flexible and on the other hand lexically fixed constructions, collocation examples found in syntactically annotated corpora are stored in a database together with competence-based analyses, cf. (Krenn, 1998a).

7 Conclusions

The increasing importance of data-oriented NLP requires the development of a specific methodology, partly different from the generative paradigm which has dominated linguistics for nearly 40 years. The importance of consistent and efficient encoding of linguistic knowledge has absolute priority in this new approach, and thus we have argued for easing the burden of explanatory claims, which has proved to be a severe constraint on linguistic formalism.

We have presented a number of linguistic analyses used in our treebank and examples of the interaction of different syntactic phenomena. We also have shown how the particular representation chosen enables the derivation of other, theory specific representations. Finally we have given examples for applications of the corpus in statistics-based NLP and corpus linguistics. Our claims are backed by an annotated corpus of currently about 12,000 sentences, all of which have been annotated twice in order to ensure consistency.

References

Ahrenberg, L. 1990. A grammar combining phrase structure and field structure. In *Proceedings of COLING '90.*

Bech, G. 1955. *Studien über das deutsche Verbum infinitum. Max* Niemeyer Verlag, Tubingen.

Bloomfield, L. 1933. *Language.* New York.

Brants, Thorsten. 1996. Tnt - a statistical part-of-speech tagger. Technical report, Universität des Saarlandes, Computational Linguistics.

Brants, Thorsten and Wojciech Skut. 1998. Automation of treebank annotation. In *Proceedings of NeMLaP-3,* Sydney, Australia.

Brants, Thorsten, Wojciech Skut, and Brigitte Krenn. 1997. Tagging grammatical functions. In *Proceedings of EMNLP-97,* Providence, HI, USA.

Hawkins, John A. 1994. *A performance theory of order and constituency.* Cambridge Univ. Press, Cambridge Studies in Linguistics 73.

Hellwig, P. 1988. Chart parsing according to the slot and filler principle. In *COLING 88,* pages 242-244.

Hudson, Richard. 1984. *Word Grammar.* Basil Blackwell Ltd.

Kasper, R. 1994. Adjuncts in the mittelfeld. In J. Nerbonne, K. Netter, and C. Pollard, editors, *German in HPSG.* CSLI, Stanford.

Krenn, Brigitte. 1998a. A Representation Scheme and Database for German Support-Verb Constructions. In *Proceedings of KONVENS '98,* Bonn, Germany.

Krenn, Brigitte. 1998b. Acquisition of Phraseological Units from Linguistically Interpreted Corpora. A Case Study on German PP-Verb Collocations. In *Proceedings of the 3rd International Symposium on Phraseology,* Stuttgart, Germany.

Lehmann, Sabine, Stephan Oepen, Sylvie Regnier-Prost, Klaus Netter, Veronika Lux, Judith Klein, Kirsten Falkedal, Frederik Fouvry, Dominique Estival, Eva Dauphin, Herve Compagnion, Judith Baur, Lorna Balkan, and Doug Arnold. 1996. TSNLP — Test Suites for Natural Language Processing. In *Proceedings of COLING 1996,* Kopenhagen.

Marcus, Mitchell, Beatrice Santorini, and Mary Ann Marcinkiewicz. 1994. Building a large annotated corpus of English: the Penn Treebank. In Susan Armstrong, editor, *Using Large Corpora.* MIT Press.

Nerbonne, John. 1994. Partial verb phrases and spurious ambiguities. In John Nerbonne, Klaus Netter, and Carl Pollard, editors, *German in Head-Driven Phrase Structure Grammar,* number 46 in Lecture Notes. CSLI Publications, Stanford University, pages 109-150.

Netter, Klaus. 1992. On Non-Head Non-Movement. In Giinter Gorz, editor, *KONVENS '92,* Reihe Informatik aktuell. Springer-Verlag, Berlin, pages 218-227.

Netter et al., Klaus. 1998. DiET - diagnostic and evaluation tools for natural language processi ng applications. In *Proceedings of LERC-98,* Granada, Spain.

Plaehn, Oliver. 1998. Annotate — benutzerhandbuch. Technical report, Universitat des Saarlandes, Computerlinguistik, Saarbriicken.

Pollard, Carl. 1996. On head nonmovement. In Arthur Horck and Wietske Sijtsma, editors, *Discontinuous Constituency.* Mouton de Gruyter.

Reape, Mike. 1994. Domain union and word order variation in German. In John Nerbonne, Klaus Netter, and Carl Pollard, editors, *German in Head-Driven Phrase Structure Grammar,* number 46 in Lecture Notes. CSLI Publications, Stanford University, pages 151-197.

Sampson, Geoffrey. 1995. *English for the Computer.* Oxford University Press, Oxford.

Skut, Wojciech and Thorsten Brants. 1998a. A Maximum Entropy Partial Parser for Unrestricted Text. In *6th Workshop on Very Large Corpora,* Montreal, Canada, August.

Skut, Wojciech and Thorsten Brants. 1998b. Chunk tagger, stochastic recognition of noun phrases. In *ESSLI Workshop on Automated Acquisition of Syntax and Parsing,* Saarbriicken, Germany, August.

Skut, Wojciech, Thorsten Brants, Brigitte Krenn, and Hans Uszkoreit. 1997a. Annotating unrestricted german text. In *DGFS-97.*

Skut, Wojciech, Brigitte Krenn, Thorsten Brants, and Hans Uszkoreit. 1997b. An annotation scheme for free word order languages. In *Proceedings of ANLP-97,* Washington, DC.

Stegmann, Rosymary and Erhard Hinrichs. 1998. Stylebook for the german treebank in verbmobil. Verbmobil report, Seminar für Sprachwissenschaft, Universität Tubingen, Germany.

Tesniere, L. 1959. *Elements de Syntaxe Structurale.* Klincksieck, Paris.

Thielen, Christine and Anne Schiller. 1995. Ein kleines und erweitertes Tagset furs Deutsche. In *Tagungsberichte des Arbeitstreffens Lexikon + Text 17./18. Februar 1994, Schlofl Hohentubingen. Lexicographica Series Maior,* Tubingen. Niemeyer.

Trubetzkoy, N. 1939. Le rapport entre le determine, le determinant et ledefmi. In *Melanges de linguistique, offers a Charles Bally.* Geneva.

Uszkoreit, Hans, Thorsten Brants, Denys Duchier, Brigitte Krenn, Lars Konieczny, Stephan Oepen, and Wojciech Skut. 98. Studien zur performanzorientierten Linguisitk. Aspekte der Relativsa tzextraposition im Deutschen. *Kognitionswissenschaft.*

Large Scale Dialogue Annotation in VERBMOBIL

Norbert Reithinger and Michael Kipp (or vice versa)
DFKI GmbH, Stuhlsatzenhausweg 3, D-66123 Saarbrücken
e-mail: {reithinger, kipp}@dfki.de

Abstract

In this paper we present an overview of the approach to dialogue related annotations in VERBMOBIL. We introduce the information annotated in dialogues of the VERBMOBIL corpus and present the rationale behind the use of the open partitur format for annotations. A tool to facilitate the task of the annotators was developed that supports two dialogue related annotation levels. Finally, we show our approach to measure the inter-coder reliability.

1 Introduction to the Environment and Tasks

Large scale language processing systems like VERBMOBIL, a speech-to-speech translation system in the domain of time-scheduling (Bub et al., 1997), heavily rely on corpus data that can be used for the training and test of knowledge sources and algorithms. For VERBMOBIL, currently about 20 CDROMs with German, English, and Japanese dialogues are available[1], containing both speech signals and transliterations. The transliterations of the signal are more than mere orthographic transcriptions. They also cover phenomena like dialectal peculiarities, a symbolic transliteration of all sorts of noises contained in the audio signal, and word breaks.

One of the most important units of processing for the dialogue module are so-called *dialogue acts* like ACCEPT, REJECT, or SUGGEST that characterise the core intention of an utterance.[2] Currently, we use 30 acts that are structured hierarchically. Early during the development of the dialogue module of VERBMOBIL (Alexandersson et al., 1997b), we noticed that the determination and processing of acts has to be based on real data. Therefore, we started to tag the transliterated dialogues of the corpus with dialogue acts.

In this paper, we first describe the format of the dialogues and the annotation level. We then present the tool we developed for annotation and finally show how we take care of the coder's reliability. We close with a look into the future.

2 Annotations and the Partitur Format

When we started annotating the dialogues, we edited the original transliteration files. We added dialogue act tags directly after the utterance using an ad-hoc annotation markup, supported by some EMACS macros. The transliterations looked like this:

```
ja , guten <!1 gu'n> Tag , Herr
Metze .  <Ger"ausch> <#Klopfen>
®(GREET AB) <"ahm> <#> <Ger"ausch>
wir sollen hier- einen <!1 ein'>
Terrain vereinbaren .  <Ger"ausch>
<A>8UNITAB)
```

where annotation consisted of a special character, the dialogue act and the speaker direction in brackets. This format was OK as long as there was just one level of markup (i.e.

[1]http://www.phonetik.uni-muenchen.de/Bas/BasKorporadeu.html

[2]We currently use the third version of the dialogue acts as presented in (Alexandersson et al., 1997a).

dialogue acts) and one annotator involved. However, as soon as we added another level of markup[3], a pandora's box of problems was opened. E.g. version problems appeared between different files annotated by different people. Version control tools alone cannot cope with this problems easily, because both the transliterations and the annotations can be changed by different persons and at different institutions and the tools available tell you where there are changes, but not how to integrate these changes.

As a remedy, we first considered the use of available markup-tools like ALEMBIC (Day, 1996) which provides for a structured markup and supports annotation. But it couldn't solve one of the major problems: the data collection agency, Bavarian Archive for Speech Signals (BAS), updates the transliterations in the master files regularly to correct errors, and these changes couldn't be merged easily with our annotated files since we changed the original text with the markup text.

The solution was to move to an extensible format BAS provides, the so-called *Partitur format*. In this format all levels of description are independent but time aligned like the single parts of a score.

It is an open format that contains independent descriptions of as many different levels of the speech signal as necessary, for instance orthography, canonical transcript, phonology, phonetics, prosody, dialog acts, or POS-tags. Symbolic links between the independent levels allow logical assignments related to the linear flow of language. These links are based on the word units of the utterance and are realized as numbers.[4]

A part of the partitur for the above mentioned sentence is show in figure 1. At the beginning it contains some bookkeeping information, followed by the transliteration, the orthographic transcription, the canonic phoneme representation, and finally by the dialogue act.

```
LHD: Partitur 1.2.3
REP: Muenchen
SNB: 2
SAM: 16000
SBF: 01
SSB: 16
NCH: 1
SPN: AAJ
LED:
TR2: 0 ja ,
TR2: 1 guten <! 1 gu'n>
TR2: 2 Tag ,
TR2: 3 Herr
TR2: 4 'Metze . <Ger"ausch> <#Klopfen>
TR2: 5 <"ahm> <#> <Ger"ausch>
TR2: 6 wir
TR2: 7 sollen
TR2: 8 hier
TR2: 9 einen <!1 ein'>
TR2: 10 Terrain
TR2: 11 vereinbaren . <Ger"ausch> <A>
ORT: 0 ja
ORT: 1 guten
ORT: 2 Tag
ORT: 3 Herr
ORT: 4 Metze
ORT: 5 <"ahm>
ORT: 6 wir
ORT: 7 sollen
ORT: 8 hier
ORT: 9 einen
ORT: 10 Termin
ORT: 11 vereinbaren
KAN: 0 j'a:
KAN: 1 g'u:tSn
KAN: 2 t'a:k
KAN: 3 h'E6
KAN: 4 m"EtsQ
KAN: 5 QE:m
KAN: 6 vi:6+
KAN: 7 z01Qn+
KAN: 8 h'i:6
KAN: 9 QaInQn+
KAN: 10 tE6m'i:n
KAN: 11 f6Q'aInba:r@n
DAS: 0,1,2,3,4 @(GREET AB)
DAS: 5,6,7,8,9,10,11 @(INIT AB)
```

Figure 1: The partitur-file for the example

As can be seen, each level of description is marked by a key, e.g. TR2 for transliteration, ORT for orthographic transcription or DAS for dialogue acts, followed by the information for this level. The DAS level shows

[3]We needed to mark whole turns with a special turn-related set of tags

[4]See http://www.phonetik.uni-muenchen.de/Bas/BasFormatseng.html#Partitur for a detailed description.

the link between the dialogue act with the basic unit indices, that are the canonic KAN units.

Since the KAN track is guaranteed to always stay the same and our own annotations do refer to the KAN layer, changes in the TR2 or ORT layer (by BAS) do not affect our annotation. On the other hand, our annotation never in any way modifies other layers (like TR2 or ORT) so that partitur files annotated at different institutes with other layers of information can be compared and merged with other files easily and consistently. Furthermore, new levels needed, e.g. for turn information, could be added swiftly due to the open specification.

In contrast to the human readable textual transliterations, the partitur is not intended to be worked on with text editors. Since the annotators are only moderately computer literate, we developed a tool to ease the pain of these poor guys.

3 Colourful Tools for Annotation

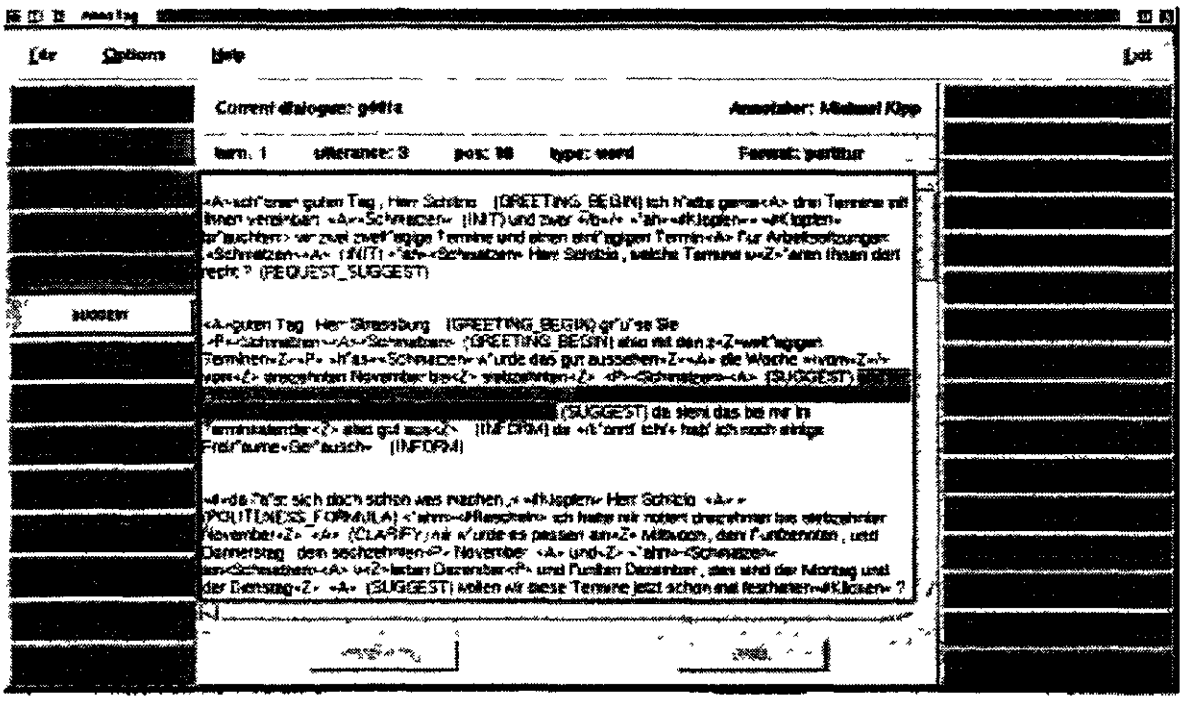
Human annotation of the training data is a tedious task. Since human annotators have to read and tag thousands of utterances, mistakes of syntactical and semantic nature are commonplace. Syntactic errors occur even when using tools, since annotators tend to sidestep the tools sometimes, e.g. they use other means to just quickly correct or add a tag with their favourite editor, even if they are told not to do so. Semantic mistakes stem from documentation of the current dialogue act definitions that cover most, but not all phenomena, due to the fact that language is - as we all know - very flexible. Also, over the time of the annotation the guidelines dynamically change through the feedback of annotators. This source of problems can only be reduced by reliability checks on a regular basis where disagreement in annotation is analysed in detail (see next section). The other important source of mistakes is the lack of technical support which should provide the annotator with sufficient means to concentrate on the semantic aspect of the annotation task.

What we learnt from these errors is that an annotation tool should draw a clear line between annotator and data. In one direction data should visualised in a way that suits the annotator. In the other direction manipulation of data must only be allowed within very limited bounds. The original base document should not be accessible to the annotators directly.

The obvious way of doing this is by visualising only task-relevant parts of the tool's internal dialogue representation and open one level to manipulation. Writing back to the original partitur file then changes not the transliteration but only the level under consideration, e.g. the DAS level when annotating dialogue acts.

Our tool, named ANNOTAG, reads partitur files of one dialogue, visualises the transliteration of the turns and presents the annotator buttons for the various dialogue acts (see fig. 2). Human annotators are now to partition these turns into utterances by labelling a part of the text with one or more dialogue act(s). There maybe more than one illocutionary act performed in one utterance which forces us to either define a new dialogue act whose definition covers the phenomenon or to label two or more of the old ones (in the latter case we speak of a multiple dialogue act). The definition of our domain-specific dialogue acts should keep the number of multiple dialogue acts low.

Segmentation and annotation are done in a single step. We are convinced that a separation of these two steps is both unnecessary and impractical. The decision that there is a dialogue act boundary and the decision which dialogue act to annotate can be made in parallel (and *should* be made in parallel since our manual actually makes use of the dialogue act definition for segmentation (Alexandersson et al., 1997a)). Splitting it simply requires the annotator to look at the same data twice and artificially split the reasoning. Also, currently some people (most of them do not annotate large corpora) argue that you must listen to the speech data in order to be able to segment properly. We



Current dialogue: g4 [illegible] Annotator: Michael Kipp

turn: 1 utterance: 3 pos: 10 type: word Format: partitur

<A> [illegible] guten Tag , Herr Schr [illegible] {GREETING_BEGIN} ich h"atte gerne <A> drei Termine mit Ihnen vereinbaren [illegible] {INIT} und zwar [illegible] zwei zweit"agige Termine und einen eint"agigen Termin <A> f"ur Arbeitssitzungen [illegible] {INIT} [illegible] Herr Schr [illegible] , welche Termine w<Z>"aren Ihnen denn recht ? {REQUEST_SUGGEST}

<A> guten Tag Herr Strassburg {GREETING_BEGIN} gr"u"se Sie [illegible] {GREETING_BEGIN} also mit den [illegible] Terminen [illegible] w"urde das gut aussehen <Z> <A> die Woche [illegible] November bis <Z> [illegible] {SUGGEST} [illegible] {SUGGEST} da sieht das bei mir im Terminkalender <Z> also gut aus <Z> {INFORM} da [illegible] hab' ich noch einige Frei"aume [illegible] {INFORM}

[illegible] sich doch schon was machen [illegible] Herr Schr [illegible] <A> {POLITENESS_FORMULA} [illegible] ich habe mir notiert dreizehnten bis vierzehnter November <Z> <A> {CLARIFY} mir w"urde es passen am <Z> Mittwoch, den f"unfzehnten , und Donnerstag den sechzehnten <P> November <A> und <Z> [illegible] <A> [illegible] Dezember <P> und f"unften Dezember , das sind der Montag und der Dienstag <Z> <A> {SUGGEST} wollen wir diese Termine jetzt schon mal festhalten [illegible] ?

Figure 2: ANNOTAG annotation tool in dialogue act mode

also think this might help in some rare cases, but in general it is not necessary. For example, pauses are transliterated, so the annotators have access to this information, too. But pauses are no criterion for segmentation, since in spoken language pauses do not tell you that much about proper segmentation. Personal communication with other annotation project like Switchboard-DAMSL and MAPTASK shows that all other large scale annotation projects also do not separate the two steps of segmentation and annotation.

Annotation with ANNOTAG works like this: Mark the text by clicking on a word of a turn. ANNOTAG then highlights all the text beginning at the preceding segment border and ending in this word. A click on one of the dialogue act buttons inserts the corresponding tag into the text. Furthermore, we can remove tags (the segments left and right of the tag are merged to one segment), undo the last action and add more tags to an existing tag (making it a multiple dialogue act). To improve readability, the text can be filtered before being displayed. Furthermore, dialogue act buttons are colour-coded according to their position in the hierarchy and unknown dialogue acts in the text are marked red.

So far we have made true our twofold wishes: first, to provide the annotator with a comfortable, easy to use surface that filters out all technical details and discomforts; second, to keep the human user in safe distance from the valuable original data. What remains are questions of extendibility. The tool is wholly written in Tcl/Tk (plus the Tix[5] extension) and can easily be manipulated. E.g. to change the set of dialogue acts we only need to change a list of strings. Recently, we added a feature enabling the user to annotate turns with turn classes which are written to a particular level in the partitur file format.

Additionally, we developed tools that were integrated to serve our special needs. For example, when we revised the set of dialogue acts we used a facility which maps all dialogue acts from one one version to an-

[5]http://www.xpi.com/tix/

other. There is also a conversion tool that allows to use annotated files in old plain text transliteration format and maps them to the partitur format. The fact that both formats do not match perfectly (there are words omitted, different turn segmentation etc.) called for a semi-automatic mechanism, showing conversion suggestions for approval/dismissal. We successfully converted more than 400 dialogues this way. Further tools allow inspection of original files, extraction of special information, or conversion of dialogues to LaTeX format.

4 κ or not: Comparing Annotators

To be useful for training and test purposes in a speech processing system our hand-annotated dialogues have to fulfil very high quality standards.

In order to assess this we carried out a number of reliability studies for the segmentation and annotation of the data.

To measure the agreement between feature-attributed data sets the κ *coefficient* is of outstanding importance (for details see (Carletta, 1996)). In the field of content analysis a K value > 0.8 is considered good reliability for the correlation between two variables, while a K of $0.67< K <0.8$ still allows tentative conclusions to be drawn.

For measuring inter-coder replicability of dialogue act coding we used 10 dialogues which altogether consist of 170 utterances and had two human coders annotate this data. The utterance labels for the two coders coincide in as many as 85.30 % of the cases. The K value of 0.8261 shows that dialogue acts can be coded quite reliably.

For testing the stability of dialogue acts we asked one coder to relabel five dialogues which altogether contain 191 utterances. The K coefficient for this study is 0.8430 with an overall agreement of 85.86%.

Currently, we annotate German, Japanese, and English dialogues. The annotators are mostly students of these nationalities, with or without linguistic background. They are trained on a set of training dialogues, where the training consist, amongst others, in annotating a set of test dialogues that are compared to the annotation of the other annotators who also annotated this set. Proceeding this way, we can identify the overall agreement of the annotators and can identify classes where an annotator seems to misunderstand the definitions of the manual.

One major use of the dialogue act annotation is to train a statistical dialogue act recogniser (Reithinger and Klesen, 1997). To test the quality of the system, we also measure the K coefficient between the annotations from humans and the program. For a test set of 51 German dialogues, we achieved agreement in 63.48% of the cases, and a K value of 0.58, using 215 dialogues for training, which of course did not contain the tested dialogues. For 18 Japanese dialogues the agreement was 71.65% with a K value of 0.68, using 81 dialogues for training.

5 Future Work

At the time we write these lines, and using the approach and the tools presented above, we have about 830 German, English, and Japanese dialogues annotated with dialogue acts, and some 100 with turn classes. The partitur format demonstrates its power in daily use, since it can easily be processed for different purposes with common UNIX tools like sed or `perl`.

In the future, besides annotating more dialogues with dialogue act and turn class information, we will extend our reliability studies to detect and repair possible weaknesses in the definition and description of the tags. Also, we will annotate the prepositional content of the utterances using a domain description language.

As we now have a pool of annotated dialogues we will use them to bootstrap an automatic annotator which will propose the human annotator the most likely tag for a unit to be annotated. We hope that we finally will get a (semi-)automatic annotator.

References

Jan Alexandersson, Bianka Buschbeck-Wolf, Tsutomu Fujinami, Elisabeth Maier, Norbert Reithinger, Birte Schmitz, and Melanie Siegel. 1997a. Dialogue Acts in VERBMOBIL-2. Technical Report 204, DFKI SaarbrückenUniversitat Stuttgart, Technische Universität Berlin, Universität des Saarlandes.

Jan Alexandersson, Norbert Reithinger, and Elisabeth Maier. 1997b. Insights into the Dialogue Processing of VERBMOBIL. In *Proceedings of the Fifth Conference on Applied Natural Language Processing, ANLP '97,* pages 33-40, Washington, DC.

Thomas Bub, Wolfgang Wahlster, and Alex Waibel. 1997. Verbmobil: The combination of deep and shallow processing for spontaneous speech translation. In *Proceedings of ICASSP-97,* pages 71-74, Munich.

Jean Carletta. 1996. Assessing Agreement on Classification Tasks: The Kappa Statistics. *Computational Linguistics,* 22(2):249-254, June.

David S. Day, 1996. *Alembic Workbench User's Guide.* The MITRE Corporation, Bedford, MA, December.

Norbert Reithinger and Martin Klesen. 1997. Dialogue act classification using language models. In *Proceedings of EuroSpeech-97,* pages 2235-2238, Rhodes.

Annotating German Language Data for Shallow Processing

Judith Klein and Thierry Declerck
German Research Center for Artificial Intelligence (DFKI GmbH)
Stuhlsatzenhausweg 3, 66123 Saarbrücken (Germany)
<Firstname. Lastname>@dfki.de

1 Motivation

Evaluation of natural language processing (NLP) systems is necessary, both to extend the linguistic coverage of the NLP components independently of their integration in a broader application system but also to assess and improve the quality of the NLP functionalities with respect to an application-specific task. For NLP-based information extraction (IE) applications, for example, the reliable analysis of natural language input is a necessary prerequisite to ensure high quality results of the IE task. Profound evaluation cycles during system development and adequacy evaluations on prototypes must be carried out on suitable reference data in order to provide information on the performance of the NLP functionalities and direct application-oriented improvements.

Even though the importance of annotated text corpora as a means for quality assessment is widely recognized within the NLP community, the realization of evaluation studies is often hampered due to the lack of suitable language material - especially for languages other than English.

2 Evaluation Scenario for smes

This paper reports on the building of reference data for the evaluation of the smes system (Saarbrücker Message Extraction System), developed at DFKI (Neumann *et al.* 1997). An annotated German reference corpus is being constructed for the diagnostic and adequacy evaluation of the NLP functionalities of smes carried out within the PARADIME (Parameterizable Domain-adaptive Information and Message Extraction[1]) project.

For shallow text analysis smes employs a tokenizer, a morphological analyzer, a part of speech (PoS) tagger and a shallow parsing component. Each component needs a reference basis of different size and format according to its task within the system. While morpho-syntactic processing as basic processing step requires reference data which aims at a very general linguistic coverage, the parsing component is designed mainly for fragment analysis and hence its reference material will consist of selected text units, especially various types of noun phrases and verb groups.

Evaluation experiments on the morphology and the PoS component have already been carried out with the help of manually validated reference corpora. For the parsing module, a part of the PARADIME corpus has been annotated for noun phrases and first evaluation cycles have started on the NP grammar of smes.

Since the building of annotated language resources is a costly and time-consuming task, automatic annotation and re-usability are key issues in the context of NLP evaluation. In order to speed-up the annotation task a bootstrapping approach is followed where the processing modules themselves will be employed to support future annotation tasks. This approach is also followed by many other groups, c.f. (Skut *et al.* 1998), (Oesterle and Maier-Meyer 1998), (Kuroashi and Nagao 1998). But, before such semi-automatic annotation is possible the results of the evaluation studies on the smes components must be integrated in order to improve their performance and guarantee that the analysis results are reliable and need only minor manual post-editing.

Several aspects will ensure re-usability: Firstly, for the used tag set a mapping has been defined to the Stuttgart-Tubingen tag set (stts)[2], which is widely used in the German NLP community. Secondly, the annotation work at PoS and phrasal level will be documented providing guidelines for general and specific linguistic cases. Thirdly, even though the annotation format corresponds to the output of the respective components it will be as transparent as possible. Additionally, it is envisaged to develop an xml (extensible markup language) interface to easily exchange the annotated reference data.[3]

[1] Information on PARADIME is available through the www under http://www.dfki.de/lt/projects/paradime-e.html.

Information on the stts tag set is available under: http://www.sfs.nphil.uni-tuebingen.de/Elwis/stts/stts.html.

[3] Information on xml can be found under Henry Thompson's homepage: http-//www.ltg.ed.ac uk/ ht/.

3 Short Sketch of SMES

Information extraction systems aim at identifying and classifying parts of texts which carry information units relevant for an application-specific task. Since a deep and complete understanding of the text may not be necessary for IE, partial representations as provided by shallow processing methods proved very useful for this task.
The core engine of the text processing part of smes consists of a tokenizer, an efficient and robust German morphology, a PoS tagger, a shallow parsing module, a linguistic knowledge base and an output construction component (c.f. Neumann 1997).

Tokenizer The tokenizer, implemented in LEX, is based on regular expressions, against which the textual input is matched for identifying some text structures (e.g. words, date expressions, etc.). Number, date and time expressions are normalized and represented as attribute value structures.

Morphix++ The morphological analysis is performed by MORPHIX++, an efficient and robust German morphological analyzer which performs morphological inflection and compound processing[4]. With its core lexicon containing more than 120,000 stem entries MORPHIX++ achieves a very broad coverage. The output of MORPHIX++ consists of a list of all possible readings, where each reading is uniformly represented as a triple of the form <STEM,INFLECTION,POS>.

Brill Tagger Ambiguous readings delivered by MORPHIX+-|- are disambiguated wrt. PoS using an adapted version of Brill's unsupervised tagger (Brill 1995) for German, which has been integrated into the smes architecture for experimental purposes[5]. The integrated BRILL tagger consists of a learning component and an application component. Within smes the learning material for the tagger consists of the lexical mark-up of texts provided by MORPHIX++. In the learning phase the tagger induces disambiguation rules on the basis of the morpho-syntactic annotated texts. In a second phase those rules are applied to the morphologically analysed input texts. In PARADIME the integration of Brill's unsupervised tagging strategy is also investigated in order to see to what extent it can be used for the semi-automatic annotation of corpora for new application domains[6].

[4]MORPHix++ is based on MORPHIX (Finkler and Neumann 1988).

[5]First experiments done with the Brill's unsupervised tagger within smes were very promising and (Neumann *et al.* 1997) reported already a tagging accuracy as high as 91.4%.

[6]In the future we might investigate the combination of the supervised and the unsupervised strategy. The supervised Brill tagger is currently being trained for German at the University of Zurich, c.f. http://www.ifi.unizh.ch/CL/tagger.

Shallow Parsing The shallow parsing component consists of a declarative specification tool for expressing finite state grammars. Shallow parsing is performed in two steps: Firstly, specified finite state transducers (FST) perform fragment recognition and extraction on the basis of the output of the scanner and of MORPHIX++ - i.e. the Brill output is not yet taken into account. Fragments to be identified are user-defined and typically consist of phrasal entities (e.g. NPs, PPs) and application-specific units (e.g. complex time and date expressions). In a second phase, user-defined automata representing sentence patterns operate on extracted fragments to combine them into predicate-argument structures.

The linguistic knowledge base The knowledge base includes a lexicon of about 120,000 root entries. The design of the lexicon allows for structured extensions. So for example, information about sub-categorization frames - extracted from the Sardic lexical data base (see Buchholz 1996) developed at the DFKI - has been recently integrated into the verb lexicon. The smes system has now about 12,000 verbs with a total of 30,042 subcategorization frames at its disposal.

The output construction component An output construction component is using fragment combination patterns to define linguistically head-modifier constructions. The distinction between this component and the automata allows a modular I/O of the grammars. The fragment combiner is also used for the instantiation of templates, providing one of the possible visualization of the results of the IE task.

4 Reference Corpus for smes

Within the PARADIME project smes is being developed as a parameterizable core machinery for a variety of commercially viable applications in the area of information extraction (IE). The planned employment of smes in other projects (e.g. in the domain of tourist information) stimulates both diagnostic evaluation to improve the NLP functionalities of smes and adequacy evaluation to show what the system's performance currently is.

Corpus Selection An important aspect of the improvement of smes is the evaluation of the performance of its components applied to large texts. For this purpose a corpus from the German business magazine "Wirtschaftswoche" from 1992 with about 180,000 tokens was selected. It consists of 366 texts, on different topics written in distinct styles.

Annotation Schema From an earlier evaluation experiment on smes (Klein *et al.* 1997) we draw the following conclusions on the design of the annotation schema: (i) for easy result comparison the reference annotations and the system output must cover the same information and (ii) for automatic comparison the format of the annotations and the system output must be identical or at least allow an easy mapping. In order to provide suitable reference material for the different shallow processing modules the annotation schema must comprise information on various linguistic levels. The BRILL tagger needs morpho-syntactically annotated lexical items as reference basis while the grammars require annotations at phrasal level, possibly including syntactic category and grammatical function. Three main aspects characterize the new annotation approach for smes:

- the annotations are directly attached to the language data and not stored separately from the corpus texts;
- the format of the annotations and the system output will allow automatic comparisons;
- the improved NLP modules will be employed for semi-automatic annotation.

4.1 Morphological Annotation

Since MORPHIX+-1- should be employed to provide the morphological annotation for the reference corpus, several cycles of diagnostic evaluation were carried out in order to extend the linguistic coverage and improve the morphological analysis. No reference material was available for this evaluation process. All corpus texts were given as input, the morphological output was inspected manually and the necessary modifications and extensions - including the integration of the German SARDIC (Buchholz 1996, Neis 1997) lexicon comprising 600,000 word forms plus morphosyntactic information - were done accordingly. The current version of MORPHIX+-I- has reached about 94% lexical coverage for the applied text corpus, where most of the words not analyzed are in fact proper nouns or misspelled words.

In order to obtain a morphologically analysed corpus all 366 texts were run through the improved morphological module which provides as output the word form (full form plus stem) together with all possible morphological readings. Using the recently implemented parameterizable interface of MORPHIX++, the cardinality of possible tag sets ranges from 23, considering only lexical category (PoS), to 780, considering all possible morpho-syntactic information.

For the first annotation step, the morphological analysis was restricted to PoS information only, where ambiguous words are associated with the alternative lexical categories (see figure 1). Items marked as unknown words are tagged as being possibly a noun, an adjective or a verb. In a second round, the tag set was extended, adding a further specification to the lexical category, e.g. *V-psp* (past participle), *A-pred-used* (adjective in predicative construction, i.e. without inflection).

```
(" Fuer" ("filer" :S "VPREF") ("fuer" :S "PREP"))
("die" ("d-det" :S "DBF"))
("Angaben" ("angeb" :S "V") ("angabe" :S "N"))
("in" ("in" :S "PREP"))
("unseren" ("unser" :S "POSSPRON"))
("Listen" ("list" :S "N") ("list" :S "V") ("liste" :S "N"))
("wurde" ("werd" :S "V"))
("grundsaetzlich" ("grundsaetzlich" :S "A") ("grundsaet-
zlich" :S "ADV" ))
("die" ("d-det" :S "DEF"))
("weitestgehende" ("weitestgehend" :S "A"))
("Bilanz" ("bilanz" :S "N"))
("zugrunde" ("zugrunde" :S "VPREF"))
("gelegt" ("leg" :S "V"))
("." ("." :S "INTP"))
```

Figure 1: Morphix++ output for PoS only

The entire smes reference corpus is now morphologically annotated with both tag set versions.

4.2 Part-of-Speech Annotation

In order to obtain reliable PoS tagged reference material about 22,000 tokens of the first version of the morphologically annotated corpus were independently validated and disambiguated by two annotators. The manual annotation proved once more to be too time-consuming. Looking for a more economical solution, it was decided to support the annotation task by the un-supervised BRILL tagger, which has recently been integrated in the smes system.

The tagger consists of a learning component, which induces disambiguation rules on the basis of the PoS information resulting from MORPHIX++, and an application component, which applies the rules to the morpho-syntactically ambiguous input texts. Instead of the manual disambiguation of the MORPHIX++ output BRILL should provide automatically disambiguated annotated texts. But before the tagger can be employed for automatic annotation thorough evaluation must prove that BRILL'S performance is sufficient for this task.

First evaluation cycles have already been carried out using the validated PoS-tagged reference corpus. BRILL was run over the morpho-syntactically analyzed test corpus using a first version of a simple rule application strategy. A set of tools were implemented to support the automatic comparison and calculation of the results. The evaluation of the BRILL tagger is based on quantitative and qualitative measures. The quantitative result is calculated as the ratio between the number of disambiguations performed and the number of all word ambiguities provided by MORPHIX++. The accuracy of the disambiguation step is measured as the ratio between the number of correct disambiguations and all disambiguations performed by the BRILL tagger. The results were very promising: 62% of the ambigui-

```
((:HEAD "fuer")
 (•COMP (:QUANTIFIER "d-det") (:HEAD "angabe")
   (:END . 3) (:START . 1) (:TYPE . :NP))
 (:END . 3) (:START 0) (:TYPE . :PP))
((:HEAD "in")
 (:COMP (QUANTIFIER "unser") (:HEAD "list")
   (:END . 6) (:START . 4) (:TYPE . :NP))
 (:END . 6) (:START . 3) (:TYPE . :PP))

((:HEAD "bilanz") (:MODS "weitestgehend")
 (:QUANTIFIER "d-det") (:END . 11) (:START . 8)
 (:TYPE . :NP))
```

Figure 2: Simplified output of the NP grammar (excluding AGR information) for the sentence *"Für die Angaben in unseren Listen* wurde grundsätzlich *die weitestgehende Bilanz* zugrunde gelegt".

ties in the input text were disambiguated showing an accuracy of 95%. During further development and test phases the tagger will be improved until a performance is reached which proves suitable for automatic annotation. We will take advantage of the rule-based approach implemented in the BRILL tagger, which allows the editing of the set of the learned disambiguation rules, and thus to add some linguistically motivated rules. Nevertheless, the BRILL output must be manually checked but we expect that with the help of the tagger the annotation of language data will be sped-up considerably.

4.3 Phrasal Annotation

For shallow parsing smes makes use of various finite state automata defining sub-grammars, like NP grammar, verb group grammar, etc. To improve the coverage of the parsing module the grammars will in turn be evaluated. We have started with the NP automata since the recognition, i.e the identification and correct analysis of the NPs, including also PPs as post-modifiers and APs as pre-modifiers, is very important for information extraction.
The task of the NP automata is to identify the noun phrases and provide their internal structures in terms of head-modifier dependencies. The output of the grammar is a bracketed feature value structure, including the syntactic category and the corresponding start and end positions of the spanned input expressions (see figure 2).
The first part of the smes reference corpus, about 12,500 tokens, has been manually annotated and validated for noun phrase information[7], resulting in 4,050 NPs annotated with phrasal boundaries and agreement information attached at the mother node, as can be seen in figure 3. The used annotation format is easily convertible into the output format of the NP automata. The annotated NPs comprise about 3,100 noun phrases without post modification - but showing sometimes very complex prenominal part - and about 950 NPs with different kinds of post modifiers.

```
("Fuer" ("fuer" :S "PREP"))
<NP<NP ("die" ("d-det" :S "DBF"))
("Angaben" ("angabe" :S"N"))
[AGR a,p,f] NP>
("in" ("in" :S "PREP"))
< NP ("unseren" ("unser" :S "POSSPRON"))
("Listen" ("liste" :S "N"))
[AGR d,p,f] NP> [AGR a,p,f] NP>
("wurde" ("werd" :S "AUX"))
("grundsaetzlich" ("grundsaetzlich" :S "A"))
<NP ("die" ("d-det" :S "DEF"))
("weitestgehende" ("weitestgehend" :S "ATTR-A"))
("Bilanz" ("bilanz" :S "N"))
[AGR n,s,f] NP>
("zugrunde" ("zugrunde" :S "VPREF"))
("gelegt" ("leg" :S "MODV"))
("." ("." :S"INTP"))
```

Figure 3: NP annotation including agreement information (case, number, gender)

First evaluation experiments on the performance of the NP grammar are being carried out with the help of the NP reference corpus. In order to provide a fine-grained performance profile it is envisaged to examine the output of the NP automata at two levels:

- phrasal level: it is checked, if the identification of the NP, i.e. the external bracketing, is correct.
- internal structure: it is checked if the head-modifier dependencies are assigned correctly.

The first measurement provides quantitative information on the performance of the identification part of the NP grammar while the second one measures the accuracy of the attributed dependency structures.
In order to direct the modification and extension of the existing NP automata it is very important how representative a specific NP phenomena is of a certain application domain. This information will be provided by text profiling where the typical characteristics of the corpus are identified. On the basis of one portion of the NP reference corpus (about 700 tokens) a text profile was established. The resulting NP classification contains prototypical structures ranging from bare plurals over simply modified NPs including adjectives and genitive NPs up to complex NPs containing various pre- and post-modifications. This first classification of the distinct NP types indicates the internal structure of the nominal constructions occuring in the corpus texts.[8] A

[7]One annotator has done the manual bracketing and the assignment of agreement information and another one has checked the annotation.

[8]The extraction and grouping of a second part of the cor-

more detailed subdivision will be worked out in order to provide the necessary representation format for the annotation of the NP intern dependency structures. The next step in text profiling will be to assign relevance values to the NP structures according to their frequency and importance in the validated part of the NP reference corpus.
In the next annotation step - which will be done with the help of the NP grammar - the NPs will additionally be annotated with their head-modifier dependencies. The grammar output will be manually checked and corrected to establish a reference corpus annotated with phrasal NP boundaries and the internal NP structure. This reference corpus will be used for further evaluations in order to check the accuracy of the NP sub-grammar.

5 Conclusion and Future Work

Annotated reference corpora are a necessary prerequisite to carry out profound evaluation on NLP systems. The time-consuming manual annotation forced us to look for a more economical solution namely the employment of the NLP modules for semi-automatic annotation. Even though the use of validated NLP modules for the building of annotated reference material can not dispense with manual work first results indicate that this approach provides substantial support for the annotation where the manual part might be reduced to the control of the automatically attached annotations. Ideally, the tools will provide highly reliable annotations - with only minimal manual checking necessary - thus allowing efficient annotation of large amounts of text material. Especially when smes will be employed for further application domains it will be a great advantage when application-specific reference data can be easily provided by the shallow text processing part. We will also investigate if the (syntactic) annotation tools, as for example those currently developed within DiET[9] and Negra[10], could smoothly be integrated into the smes environment. The graphical interfaces of those tools could facilitate the manual inspection of the automatically built reference data. In order to make the annotation schema re-usable for other systems and applications it is planned to provide an interface to the xml representation format.

Acknowledgment

The research underlying this paper was supported by grants from the German Bundesministerium für Bildung, Wissenschaft, Forschung und Technologie (BMB+F) for the DFKI project PARADIME (FKZ ITW 9704). We are grateful to Günter Neumann for his fruitful comments on the annotation work within PARADIME. We thank Milena Valkova and Birgit Will for their careful annotation work.

pus NPs made only slight changes in the classification schema necessary.

[9]More information on DiET is available under http://dylan.ucd.ie/DiET/.

[10]For information on the Negra project see http://www.coll.uni-sb.de/cl/projects/negra.html.

References

Brill, E. (1995). Unsupervised learning of dismabiguation rules for part of speech tagging. *Proceedings of the Second Workshop on Very Large Corpora,* WVLC-95 Boston, 1995

Buchholz, S. (1996). *Entwicklung einer lexikographischen Datenbank für die Verben des Deutschen.* Master Thesis, Universitat des Saarlandes, 1996

Finkler, W.; Neumann, G. (1988). Morphix: A fast Realization of a Classification-based Approach to Morphology. *Proceedings der 4- Österreichischen Artificial Intelligence Tagung, Wiener Workshop Wissensbasierte Sprachverarbeitung,* pp. 11-19, Berlin, 1988

Klein, J.; Busemann, S.; Declerck, T. (1997). Diagnostic Evaluation of Shallow Parsing Through an Annotated Reference Corpus. *Proceedings of the Speech and Language Technology Club Workshop, SALT-97,* pp.121-128, Sheffield, 1997

Kuroashi, S.; Nagao, M. (1998) Building a Japanese Parsed Corpus while Improving the Parsing System. *Proceedings of the first international Conference on Language Resources and Evaluation, LREC-98,* Granada, 1998, pp.719-724

Neis, I. (1997) *SarDic - das Saarbrücker Dictionary: eine morphologische Workbench und em morphologischer Server.* Master's Thesis, Saarbriicken, 1997

Neumann, G.; Backofen, R.; Baur, J.; Becker, M.; Braun, C. (1997). An Information Extraction Core System for Real World German Text Processing. *Proceedings of the 5th Conference on Applied Natural Language Processing, ANLP-97,* pp.209-216, Washington DC, 1997

Oesterle, J.; Maier-Meyer, P. (1998) The GNoP (German Noun Phrase) Treebank. *Proceedings of the first international Conference on Language Resources and Evaluation, LREC-98,* Granada, 1998, pp.699-703

Skut, W.; Brants, T.; Krenn, B.; Uszkoreit, H. (1998) A Linguistically Interpreted Corpus of German Newspaper Text. *Proceedings of the first international Conference on Language Resources and Evaluation, LREC-98,* Granada, 1998, pp.705-711

Adding Manual Constraints and Lexical Look-up to a Brill-Tagger for German

Gerold Schneider and Martin Volk
University of Zurich
Department of Computer Science
Computational Linguistics Group
Winterthurerstr. 190, CH-8057 Zurich
gschneid volk@ifi.unizh.ch

Abstract

We have trained the rule-based Brill-Tagger for German. In this paper we show how the tagging performance improves with increasing corpus size. Training over a corpus of only 28'500 words results in an error rate of around 5% for unseen text. In addition we demonstrate that the error rate can be reduced by looking up unknown words in an external lexicon, and by manually adding rules to the rule set that has been learned by the tagger. We thus obtain an error rate of 2.79% for the reference corpus to which the manual rules were tuned. For a second general reference corpus lexical-lookup and manual rules lead to an error rate of 4.13%.

1 Introduction

There already exist a number of taggers for German (Lezius et al., 1996). We have noticed, however, that none of them is rule-based. But as Samuelsson and Voutilainen (1997) have demonstrated rule-based taggers can be superior to statistical taggers. We have therefore adapted and trained the supervised version of the rule-based Brill-Tagger to German. To this end we have been building up a German training corpus, which currently consists of about 38'000 tagged words, where all tags have been manually checked.[1] In this paper we show how the tagging performance improves with increasing corpus size. In addition we demonstrate that the error rate can be further reduced by looking up unknown words in an external lexicon, and by manually adding rules to the rule set that has been learned by the tagger.[2]

1.1 Rule-Based Tagging

We have chosen the Brill-Tagger for the following reasons:

Practical Performance The rule-based Brill-Tagger (Brill, 1992, Brill, 1994) has shown good results for English. Samuelsson and Voutilainen (1997) show that a rule-based tagger for English can achieve better results than a stochastic one. Chanod and Tapanainen (1995) prove the same for French.

Theoretical Advantages While the constraints for French by Chanod and Tapanainen (1995) and for English by Samuelsson and Voutilainen (1997) are hand-written, the Brill-Tagger is self-learning. It employs a transformation-based error-driven learning method. Ramshaw and Marcus (1996) describe this as a compromise method, which means that it involves both a statistical and a symbolic component. Instead of pure n-grams the Brill-Tagger uses rule templates to restrict the search space.

Linguistic Accessibility and Extensibility Another advantage of rule-based tagging over statistical approaches is the linguistic control, as Ramshaw and Marcus (1996) point out. Linguistic knowledge first defines the linguistic principles to be statistically investigated, i.e. the Brill-Tagger set of rule templates. Second, they allow to tune an automatically abstracted

[1] We would like to acknowledge the help of Gero Bassenge, Alexander Glintschert, Sven Hartrumpf, Sebastian Hubner, Sandra Kubler, Andreas Mertens and Elke Teich in checking part of our training corpus.

[2] The web version of our tagger and information about its availability can be found at http://www.ifi.unizh.ch/CL/tagger.

description of the training corpus, i.e. the rule files. Third, they help in analysing the results and in pin-pointing the remaining errors.

1.2 The Brill-Tagger for German

The Brill-Tagger was originally developped for English.[3] For German, we had to start from scratch, first finding a suitable tag-set, adapting the tagger code slightly, and then manually tagging a German corpus. The changes in the tagger code needed for German are well documented in the tagger manuals. It is necessary to adapt the initial guess for capitalized words, which for English is a tag for proper noun in the original code. This had to be changed to the tag for common noun in our tag-set, because in German all nouns are capitalized.

1.2.1 Training Phase

The Brill-Tagger is trained in two steps. In the first step, each word is assigned its most likely tag, based on the training corpus. In the second step, the errors made in step one are recorded, and the tagger finds rules for the biggest possible error elimination based on the context or internal build-up of words. The tagger formulates these rules on the basis of the rule templates. Every rule is then tested against the training corpus, the number of corrected errors are weighed against the number of errors newly introduced by this rule. The rule with the greatest net improvement is included in the rule set. This learning procedure continues iteratively, until a certain threshold is reached. Due to its iterative character, newly acquired rules are already respected for the elimination of the next error.

Using this procedure, the Brill-Tagger generates a lexicon and two rule files, one for context rules and one for lexical rules.

1.2.2 Application Phase

For the application of the tagger to a text, the tagger uses the files generated during the training. The lexicon contains each word with all its tags as they occurred in the training corpus. The tag at the first position is the most likely tag, which will be assigned to a word as a first guess. These guesses are then corrected according to the learned context rules.

Context rules take the context of a word into consideration. The Brill-Tagger has an observation window of size 4: the furthest reaching rule template allows for the consideration of three words to the left or to the right. This is bigger than in most statistical taggers. We will give examples of context rules in section 4.1.

The other rule file contains lexical rules. Lexical rules are solely used for tagging unknown words. The following is an example of a simplified lexical rule

```
lich hassuf 4 ADV
```

This lexical rule will have the effect that unknown words with a four-letter suffix `-lich` are transformed into adverbs from whatever their first guessed tag was.

Brill rules are transformation rules, which means that a tag is transformed into another tag if a rule applies. But at any stage a word will have exactly one tag. In this sense, transformation rules (Ramshaw and Marcus, 1996) are different from constraint rules (Samuelsson and Voutilainen, 1997).

1.3 Tag-set and Corpora

We use a tag-set widely acknowledged for German, the so-called Stuttgart-Tubingen Tag-Set (STTS) (Schiller et al., 1995), which contains 51 part-of-speech tags plus some punctuation tags. Our corpus consists of texts from the University of Zurich annual report and currently contains about 38'000 words.

2 Performance of the Brill-Tagger for German

In this chapter we show how the tagging accuracy increases with increasing corpus size, until the progress flattens out, partly due to the tagging difficulties for German, which we will describe below. In separate experiments, which are documented in (Volk and Schneider, 1998), we show that the Brill-Tagger and the statistical tagger by Schmid (1995) achieve similar results for German.

2.1 Training the Brill-Tagger with our Corpus

We used a utility provided by Brill to split the 38'000 word corpus into two halves, say A and

[3] The Brill-Tagger is available from its author at http://WWB.cs.jhu.edu/~brill.

B. This utility repeatedly takes the next two sentences from the corpus and randomly puts one sentence to file A and the other one to file B. We divide these halves again by the same method to get four parts. We use the first three parts as the training corpus, which we call TC. The remaining quarter is reference material. We divide this reference material again in the same way. We call the reference corpora we thus get RC1 and RC2.

2.1.1 Training Progress

In order to illustrate the training progress, we first train the tagger with only 12.5% of the corpus and tag RC1 with the data obtained from the training. Then, we move on to 25%, 50% and 75% of the corpus and tag RC1 again. Table 1 illustrates the training progress. The 75% corpus is TC, i.e. the training corpus we will use for the rest of this paper. After training the Brill-Tagger with the training corpus TC, the training module has learnt 186 lexical rules and 176 context rules. When applying these rules to RC1 the error rate is at 5.04%. This means that 94.96% of all the tokens in RC1 receive the same tag as manually prespecified.

For illustration purposes, we also add RC2 to TC and tag RC1 again, reducing the error rate to 4.81%. As expected, the error rate in table 1 flattens out, suggesting that the increase in tagging accuracy from using bigger training corpora will become increasingly smaller.

Size of the training corpus	Error rate for RC1	Avg. Ambig. per Token for RC1
12.5%	11.96%	1.209
25.0%	9.01%	1.274
50.0%	6.33%	1.307
TC = 75.0%	5.04%	1.309
TC+RC2 = 87.5%	4.81%	1.373

Table 1: Error Rates on RC 1

Of course, we may also tag RC2 with TC. Coincidentally, the error rate is a little higher than with reference corpus RC1, at 5.59%.

When we add the reference material to the training corpus, the error rate drops significantly, as table 2 illustrates. Of course this error rate of 1.49% has no "real world" significance, because no matter how big a training corpus there will always be new words and new syntactic constructions in a new text to be tagged.

Size of training corpus	Error rate for RC1
TC+RC2+RC1 = 100%	1.49%

Table 2: Drastic Error Rate Reduction on Including Reference Material into Training Material

2.1.2 Unknown Words

But we found this increase so striking that we wanted to know if it is rather due to the known vocabulary or due to the larger number of transformation rules. When tagging RCl with the rule files learned from TC, but the lexicon learned from the entire corpus (i.e. TC+RC1+RC2) the error rate was only 1.86%, almost as good as when tagging with the rule files learned from the entire corpus. On the contrary, when tagging with the rule files learned from the entire corpus but using only the lexicon learned from TC, the error rate was 4.86% (even a little worse than when using the TC rules). This indicates that the Brill part-of-speech guesser for unknown words is still unsatisfactory (Brill, 1994). Section 3 describes one way to increase the tagging accuracy for unknown words.

2.1.3 Average Ambiguity per Token

Success and error rates alone are not enough as a measure for the efficiency of a tagger. If few words in the text to be tagged are ambiguous in the tagger lexicon, it is easy for the tagger to achieve good results. We therefore provide a figure on every training step for the average ambiguity per token. This figure is calculated for all tokens in a text (RCl in our case) that are contained in the tagger lexicon. Unknown words are not used in this calculation. Enlarging the training corpus has two effects on the tagger lexicon. First, there will be more tokens in the lexicon and second, many tokens are assigned multiple tags. This accounts for the increase in the average ambiguity of tokens listed in the third column of table 1.

2.2 Tagging Difficulties for German

Adjective vs. Past Participle: In German the distinction between predicative adjectives and past particles is difficult for linguistic experts and therefore also for the tagger. E.g. in the following corpus sentence

(1) ... während die technikbezogenen Disziplinen an der ETH Zurich vertreten sind.

it is difficult to judge whether *vertreten* is a past participle or an independent adjective. The sentence can be transformed into a similar active sentence, but it is debatable whether this involves a semantic change. In quantitative terms, however, only 3% of the errors from tagging RC1 are mistakes of this type.

Verb-Forms: The STTS tag set calls for a distinction of finite verb form, infinitive form and past participle form. But in German the finite verb form for the first and third person plural, present tense, is identical with the infinitive form. In addition there are many verbs where the past participle is identical with the infinitive or with a finite verb form. That means that one can decide on the verb form only by looking at the complete verb group in a clause.

But in German matrix clauses the verb group is a discontinuous constituent with the finite verb in second and the rest of the verb group in clause final position. This means that the distance between a finite auxiliary verb and the rest of the verb group can easily become too big for the window of a tri-gram tagger, as (Schmid, 1995) notes. Unfortunately, the Brill-Tagger window in many cases is not big enough either. In the following examples, our tagger mis-tagged beantragt as finite verb, while verlangt is mis-tagged as past participle.

(2) Hier hat der Ausschuss ... fur die ersten beiden Punkte der Erziehungsdirektion beantragt, ...

(3) Die Theologische Fakultät verlangt Kenntnisse in Latein, Griechisch und Hebräisch.

When analysing the remaining 5.04% errors from tagging RC1 with our TC we find that indeed 25% of the errors involve a wrong verb form.

Capitalisation: Unlike in English, all nouns are capitalised in German. This means that the tagger mis-tags many unknown proper names as common names, and that sentence-initial unknown words are also often mis-tagged as common nouns. When analysing the errors in RCl we find that 17% of the errors involve capitalisation.

3 The Impact of Lexical Lookup

As shown in 2.1.2, unknown words account for a large portion of the errors. We therefore experimented with sending the words not present in the tagger lexicon to the wide-coverage morphological analyser Gertwol (Oy, 1994). An automated mapping procedure over the Gertwol output extracts all possible STTS tags for a given word and temporarily appends these new words with their tags to the tagger lexicon. There is an obvious increase in the tagging accuracy, as table 3 shows.

Training Corpus	Lexical look-up	Error rate for RC1	Error rate for RC2
TC	no	5.04%	5.59%
TC	yes	4.33%	4.74%

Table 3: The impact of lexical lookup

But the impact of this lexicon-lookup is smaller than expected. The problem is that Gertwol delivers an unordered list of tags for a given word-form, which includes even rare readings. The Brill-Tagger, on the other hand, needs the most likely part-of-speech at the first position in the lexicon. We have not yet found out a method to weigh the Gertwol output appropriately.

4 The Impact of Manual Constraints

As stated in 1.1 the Brill-Tagger has the advantage that it finds linguistic rules from the training corpus which can be inspected, assessed and extended.

The net of automatically learned and partly interdependent rules might be a fragile system however, lenient to decrease in efficiency after manual editing. Since the rules depend on each other, their position within the rule file is relevant. How can a linguist know at which place he or she should insert rules? And to what an extent are the rules interdependent?

Ramshaw and Marcus (1996) have investigated this question. They state [p. 151-2]:

> The trees for a run on 50K words of the Brown Corpus bear out that rule dependencies, at least in the part-of-speech tagging application, are limited. ... [T]he great majority of the learning in this case came from templates that applied in one step directly to the baseline tags, with leveraging being involved in only about 12% of the changes. The relatively small amount of interaction found between the rules also suggests that the order in which the rules are applied may not be a major factor in the success of the method for this particular application, and initial experiments tend to bear this out.

Given this reassurance, we added rules manually to the end of the contextual rule file.

4.1 Examples of Context Rules

As the Brill-Tagger has only little built-in linguistic knowledge it is on the one hand almost language-independent, on the other hand it has to rely on statistical data for learning linguistic rules.

It is striking to see that the learning algorithm automatically learns many well-known and linguistically sound context rules like:

```
APPR PTKVZ NEXTTAG $.
```

This rule means that what is initially tagged as a preposition (APPR) should be transformed into a separated verb prefix tag (PTKVZ) if found at the end of a sentence (NEXTTAG $.) - if the word in question can be found as PTKVZ in the tagger lexicon. Prepositions never occur at the end of a sentence indeed.[4] In the sentence

(4) Ich gebe nie auf.

auf/APPR is thus correctly transformed into auf/PTKVZ. More surprisingly, the learning algorithm even detects rules one is hardly aware of:

[4] **Our tagset distinguishes prepositions and postpositions.**

```
VAINF VAFIN NEXTTAG ADV
```

This rule transforms the infinite auxiliary verb tag into a finite auxiliary verb if followed by an adverb. Indeed, German word order seems to forbid sentences in which infinite auxiliaries are post-modified by an adverb:

(5) * Wir werden haben sehr schemes Wetter.

(6) * ... um zu sein ganz sicher.

The learning algorithm also detects a number of linguistically more questionable context rules which, however, correctly work in the majority of language uses, but which may also lead to mistakes:

```
VVFIN WPP NEXT10R2TAG $.
```

This rule means that sentence-final finite verb tags should be transformed into verb participles. While this rule is correct for e.g.

(7) Der Arzt hat seine Patienten behandelt.

it will produce wrong results for - in our corpus apparently less frequent - sentences like

(8) Der Arzt ist aufmerksam, wenn er seine Patienten behandelt.

Moreover, the learning algorithm misses some basic linguistic facts. In a reference corpus, we find e.g.

(9) unseres/PPOSS Reformvorhabens/NN

PPOSS stands for substituting possessive pronoun. The tag-set distinguishes between substituting and attributive pronouns. Without any linguistic knowledge, the tagger cannot know that a substituting pronoun will hardly be followed by a common noun (NN). A transformation to attributive personal pronoun seems plausible. This is where our **manual rules** come in. We add the following rule the contextual rule file:

```
PPOSS PPOSAT NEXTTAG NN
```

After training our tagger with TC we tag RC1 with lexical look-up (cf. section 3). The error rate is 4.33%. We manually checked the 189 errors and wrote and individually tested manual rules where we believe rules are linguistically plausible and expressable in the formalism - like the one for attributive personal pronouns above. We added the manual rules to the automatically learned rules. With 97 manual rules the error rate drops to 2.79%. Coming up with and testing new rules is a time-consuming process. Determining these 97 manual rules took us around 4 hours.

4.2 Results with Manual Constraints

The fact that rule interdependence is low (cf.section 4) suggests that we can safely add rules. On the other hand this may also indicate that rules are so independent because each of them only has a limited effect. In order to answer this question, we tag RC2, first only with lexical look-up (as described in 3) and with the automatically learned context rules from TC - we get an error rate of 4.74% -, then together with the above 97 manual context rules, for which we get an error rate of 4.13%.

Because of the small interaction, we may freely add manual rules written at other occasions. At an earlier stage of our research, when our training corpus comprised of 28'000 words, we wrote 141 manual rules for test purposes (cf. 4.3). We now add these manual rules to the file containing the automatically learned rules and the manual rules tuned for RCl. If we use this rule file for RC2 (again with lexical look-up), we get another increase in accuracy to 4.09%.

One may fear that manual rules are corpus-specific and bear no general linguistic siginificance, which would entail that they lead to an error increase in other corpora. In order to test this, we used the above rule file (i.e. automatically learned rules plus manual rules tuned for RCl plus manual rules from earlier stage) to tag RCl (with lexical look-up). We get only a slight error increase from 2.79% to 2.86%. This indicates that only a small fraction of the tuned rules are indeed corpus-specific, while the majority are linguistically accurate, at least in the sense of linguistic performance.

We conclude that because the context rules of the Brill-Tagger are independent and each has only a limited effect, the knowledge can be freely accumulated and will lead to better results in most cases. Adding manual rules is thus a feasible and useful practice for Brill tagging.

4.3 The Limits of Tagging Performance

As mentioned above, at an earlier stage of our research, when the entire corpus comprised of about 28'000 words, we wrote context rules based on the results of tagging the training corpus itself.

When a tagger tags its training corpus the error rate is naturally much lower than in tagging a new text. In the case of our 28'000 word corpus the error rate was at 1.81%. Based on these errors we wrote 141 manual context rules and added them to the 121 automatically learned context rules. The resulting error rate was just below 1%, at 0.95%. For the remaining 266 errors, no context rule could be found that resulted in any improvements. We therefore suggest that, given the window of the context rules in the Brill formalism and the restricted expressivity of the rules, and given the relatively free word order of the German language and the STTS tag set, an error rate of just slightly below 1% is about the best possible rate that can be achieved by a Brill-Tagger for German.

5 Conclusions

We have shown that the rule-based Brill-Tagger can be trained successfully over a relatively small annotated corpus. Tagging performance then suffers from unknown words but this can be alleviated by looking-up these words in an external lexicon. This lexicon should not only provide all possible tags but also identify the most likely tag. Current wide-coverage lexical resources like Gertwol do not contain this information. Perhaps a statistical analysis of online dictionaries, as proposed by Coughlin (1996), could help to compute this missing information.

Tagging performance can also be improved by adding manual rules to the automatically learned rule set. In our experiments a set of about 100 manual rules sufficed to increase the tagging accuracy from 95% to around 96%. We also demonstrated that the Brill-Tagger is relatively robust as to the order in which the manual rules are added. Unfortunately many of the remaining errors (e.g. verb form problems) lie outside the scope of the tagger's observation window. Therefore we need to add a more powerful component to the tagger or build a shallow parsing post-processor for error correction.

References

Eric Brill. 1992. A simple rule-based part-of-speech tagger. In *Proceedings of ANLP,* pages 152-155, Trento/Italy. ACL.

Eric Brill. 1994. A report of recent progress in transformation-based error-driven learning. In *Proceedings of AAAl.*

Jean-Pierre Chanod and Pasi Tapanainen. 1995. Tagging French - comparing a statistical and a constraint-based method. In *Proceedings of EACL-95,* Dublin.

Deborah A. Coughlin. 1996. Deriving part of speech probabilities from a machine-readable dictionary. In *Proceedings of the Second International Conference on New Methods in Natural Language Processing,* pages 37-44, Ankara, Turkey.

W. Lezius, R. Rapp, and M. Wettler. 1996. A morphology-system and part-of-speech tagger for German. In D. Gibbon, editor, *Natural Language Processing and Speech Technology. Results of the 3rd KONVENS Conference (Bielefeld),* pages 369-378, Berlin. Mouton de Gruyter.

Lingsoft Oy. 1994. Gertwol. Questionnaire for Morpholympics 1994. *LDV-Forum,* 11(1):17-29.

L.A. Ramshaw and M.P. Marcus. 1996. Exploring the nature of transformation-based learning. In J. Klavans and P. Resnik, editors, *The balancing act. Combining symbolic and statistical approaches to language.* MIT Press, Cambridge,MA.

C. Samuelsson and A. Voutilainen. 1997. Comparing a linguistic and a stochastic tagger. In *Proc. of ACL/EACL Joint Conference,* pages 246-253, Madrid.

A. Schiller, S. Teufel, and C. Thielen. 1995. Guidelines fur das Tagging deutscher Textcorpora mit STTS (Draft). Technical report, Universitat Stuttgart. Institut fiir maschinelle Sprachverarbeitung.

Helmut Schmid. 1995. Improvements in part-of-speech tagging with an application to German. Technical report, Universitat Stuttgart. Institut fiir maschinelle Sprachverarbeitung. (Revised version of a paper presented at EACL SIGDAT, Dublin 1995).

Martin Volk and Gerold Schneider. 1998. Comparing a statistical and a rule-based tagger for German. Manuscript.

Annotation Management for Large-Scale NLP

Remi Zajac
Computing Research Laboratory, New Mexico State University
zajac@crl.nmsu.edu

We describe a new flexible annotation scheme used in CRL's Document Manager which extends the Tipster Document Architecture in several innovative directions. Annotation types are defined using typed feature structure definitions; the annotations themselves are instances of these types. Annotations are stored in an object-oriented database system; the corpora files can be stored on a file system (local or remote) or in the database itself; the Document Manager maintains the relations between annotations and the documents

1 Introduction

The development of modern natural language processing system relies on the exploitation of corpora for either extracting linguistic information or testing the NLP systems. Although the extraction of information is certainly of primary importance, the use of large corpora for testing and evaluating NLP systems is an important feature in the development of large scale NLP tools. Part of the US Tipster program (Grishman 95, Caid et al. 96) is devoted in the development of a so-called Document Architecture aimed at facilitating the integration, the reuse and the evaluation of NLP components on very large collections of documents, as the one used in the Tipster and MUC programs. Although the Tipster architecture is primarily aimed at information retrieval and extraction, it has also been used at CRL for machine translation and machine-aided human-translation. Our previous extensive experience with the Tipster-conformant implementation of a Tipster Document Manager developed at CRL (Sharpies & Bernick 96)' lead us to a new generic implementation which can be used for NLP at large (see e.g., Zajac et al. 97). This new Document Manager, developed within the Corelli project at CRL, can be viewed as a specialized object-oriented database system for managing large collections of documents annotated with complex data structures. This implementation extends the basic concepts of the Tipster Document Architecture in the following directions:

- The structure of the set of annotations is designed to facilitate the implementation of an NLP system by reusing existing NLP components and minimizing the interaction between these components.
- The annotations objects themselves are typed feature structures. They extend the Tipster annotations, which are simple sets of attribute-value pairs; they are also used as the representation device for many modem NLP systems.
- The types of the annotations used in the Document Manager must be declared. The Document Manager uses the type definitions (see e.g., Zajac 92) to check that document annotations created by a component conform to the declared types.

1. Initially prototyped by Ted Dunning.

The remainder of the paper gives an overview of the Corelli Document Management Architecture, and of the extensions which are currently being implemented; presents the new annotation scheme based on feature structures and how the Document Manager makes use of this scheme to provide better integration and testing facilities; gives an overview of the implementation.

2 Document Management

In the Corelli Document Architecture, components do not talk directly to each other but communicate through annotations attached to the document being processed. Each component of an NLP system reads and writes annotations using the Document Manager interface. This model reduces inter-dependencies between components, promoting the design of modular applications (Figure) and enabling the development of blackboard-type applications such as the one described in (Boitet & Seligman 94). The Corelli Document Architecture provides solutions for

- Representing information about a document,
- Storing and retrieving this information in an efficient way,
- Exchanging this information among all components of an application.

If a system reuses components that were not designed to work with each other, there will still be an impedance mismatch to be resolved: the architecture does not provide ready-made solutions for translating linguistic structures (e.g., mapping two different tagsets or mapping a dependency tree to a constituent structure), since these problems are application-dependent and need to be resolved on a case-by-case basis; such integration is however feasible, as demonstrated by the various Tipster demonstration systems, and use of the architecture reduces significantly the load of integrating a component into an application.

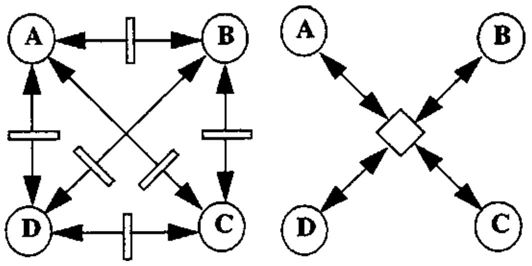


Figure 1: Centralizing document annotations enables a modular architecture and reduces the number of interfaces from the order of n^2 to the order of n.

2.1 Document Architecture

The basic data object of the architecture is the document: documents can have properties (a set of attribute-value pairs) and annotations, and can be grouped into collections. Annotations are used to store information about a particular segment of the document (identified by a span, i.e., start-end byte offsets in the document content) while the document itself remains unchanged. This contrasts with the SGML solution used in the Multext project where information about a piece of text is stored as additional SGML mark-up in the document itself (Ballim95, Thompson95). The Corelli architecture supports writable data as well as read-only data (e.g., data stored in a CD-ROM or on a remote file system); no copy or modification of the original documents is needed. This solution enables the processing of very large corpora such as the ones used in TREC with reasonable performances. Documents are accessible via a Document Manager which maintains persistent collections, documents and their attributes and annotations, using a commercial database management system to support persistency. The implementation of the architecture takes advantage of the client-server architecture of the database (which uses TCP/IP) and allows local as well as remote clients to connect to a Document Manager.

The Document Manager Graphical User Interface provides a set of tools for manipulating and browsing the documents and collections of documents; specialized viewers allows the display of document annotations.

2.1.1 Document Annotations

The original Tipster Document Architecture provides a relatively low-level structure for annotations: annotations are sets of attribute-value pairs; values can be either strings or numbers or recursively annotations; all annotations are stored as a set (actually, a bag). This imposes complex transformations between the data structure produced by a component and the Tipster annotation structures; if annotations are related to each other (as for example as edges of a chart parser or as nodes of a parse tree), an additional data structure is required to represent relations between annotations.

The Corelli Document Architecture partitions the set of annotations into labeled sub-sets, where each sub-set is the input and/or the output of a component. Each annotation sub-set relevant to a given component is structured as a lattice, which enables a direct representation of for example a word-lattice as produced by a speech-recognition system or as an ambiguous output of a morphological analyzer (Boitet & Seligman 94, Amtrup et al. 97). Each annotation is a (typed) feature structure. The annotation structure facilitates the integration of NLP components such as unification-based parsers or morphological analyzers since the input of such component is a word or a sentence and the data structures read and/ or produced are feature structures or graphs of feature structures. F3 shows that annotations can be structured in graphs where each graph represent the linguistic structure of a sentence as computed by a given component. An annotation is an edge in a graph of annotations and also contains a pair of pointers (byte offsets) to the span of text covered by the annotation.

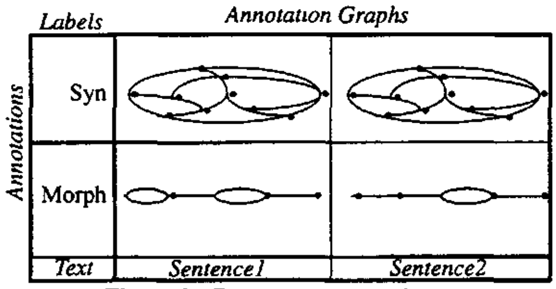


Figure 2: Document annotations.

The specification of a component includes the specification of the pre- and post-conditions of the component. A pre- or post-conditions specifies which annotation sub-sets are accessed by the components (using the sub-sets' labels) and for each sub-set, the required features and values (specified as a typed feature structure).

2.1.2 Annotation Management

The Corelli Document Architecture provides a way to declare annotation types and to check that any annotation added to a document at runtime conforms to the declared types. Annotations are typed feature structures, and allowed feature structures are defined using type definitions (Zajac 92). The Tango language developed at CRL provides the facilities for defining the types of feature structures. This language supports the notion of module (package) and includes a set of pre-defined types (integers, strings, lists and regular expressions). The runtime system provides a set of methods to type-check feature structures as well as a set of unification methods. This runtime engine is used in the implementation of several unification-based formalisms at CRL.

Tango modules are stored in a database, and the Tango development environment supports functionalities to define modules and types, and compile modules. The runtime modules can then be used in a variety of applications including the document manager: instances of these types are document annotations. The Graphical Programming Environment of the Document Manager gives access to the Tango toolset and an application programmer can import Tango modules and use the type definitions, but also modify and recompile Tango modules.

The Tango runtime system supports several levels of type checking and this set of functionalities is used by the Document Manager to check at runtime annotations which are created by a component before actually storing the annotations persistently in the Document Manager persistent store (this type-checking can be turned-off by the programmer). The Document Manager uses the declaration of pre- and post-conditions declared for a given component to perform runtime type checking (see below).

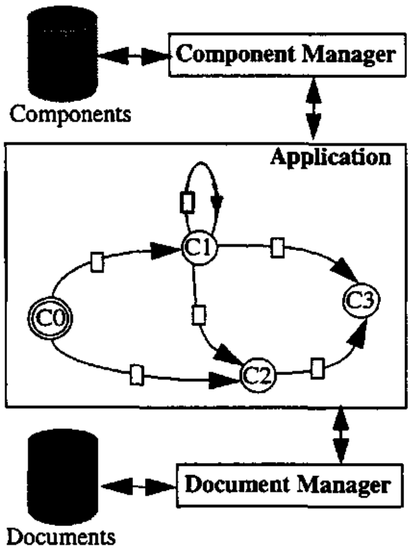


Figure 3: Integration of NLP components in the Corelli Document Architecture.

2.2 Application Framework

The Corelli Document Architecture includes an Application Framework which supports the construction of NLP applications by integration of NLP components through a high-level Graphical Programming Environment. The Application Editor supports a drag-and-drop graphical interface for integrating components in a single application in a way similar to the GATE GDE (Cunningham et al. 94, 96). The components themselves may be distributed and communicate with the application using a commercial agent-based architecture (from ObjectSpace). The Application Framework interpreter allows a step-wise execution of the application and stores all intermediary results (output of each component) in the Document Manager where they can be displayed using the Document Manager viewers.

2.2.1 Component Architecture

The data layer of the Corelli Document Architecture, as described above, provides a static model for component integration through a common data framework. This data model does not provide any support for communication between components, i.e., for executing and controlling the interaction of a set of components, nor for rapid tool integration. The Corelli Component Architecture fills this gap by providing a dynamic model for component integration: this framework provides a high-level of plug-and-play, allowing for component interchangeability without modification of the application code, thus facilitating the evolution and upgrade of individual components.

An NLP component is integrated in the architecture by implementing the Corelli Component interface which defines a standardized set of methods to execute a component's functionalities and provides high-level communications capabilities allowing distribution of components. This interface acts as a wrapper for the component's code and several integration solutions are possible:

- If the component has a Java API,[1] it can be encapsulated directly in the wrapper's code.
- If the component has an API written in one of the languages supported by the Java Native Interface (currently C and C++), it can be dynamically loaded into the wrapper at runtime and accessed via the Java front end.
- If the component is an executable, the wrapper must issue a system call for running the program and data communication usually occurs through files.

2.2.2 Component Management

In a way which is similar to the GATE component architecture (GATE), a Corelli Component has pre- and post-conditions. These conditions are defined as (typed) feature structures and the Document Manager can dynamically check the validity of annotations created by a component by checking that each input annotation is subsumed by the pre-condition and that each annotation produced by the component is subsumed by the post-condition.

The programmer building an application using the Application Framework defines for each component pre- and post-conditions. A component imports one or more Tango modules, and pre- and post-conditions are defined as expressions of typed feature structures which are instances of types declared in the imported modules. The Graphical Programming Environment type-checks the declarations or pre- and post-conditions using the Tango runtime facilities.

When the programmer defines an application as a graph of components (where the graph is used express the control flow between components), the Graphical Programming Environment also type-check the entire application by checking the compatibility of pre- and post-conditions for all possible execution paths in the application.

3 Conclusions

We described a new annotation scheme that:

- allows to annotate read-only documents;
- supports efficient annotation of very large document collections;
- allows to control the validity of annotations as they are added to a document;
- interfaces readily with modern unification-based NLP components.

An alpha version of the Corelli Document Manager has been released and is being tested at several research institutes. The Document Manager is implemented in Java and uses a Java OODBMS back-end from ObjectDesign. The Component Architecture has been prototyped using RMI and we are still exploring other options to implement the distributed Application Framework, including ObjectSpace's Voyager and CORBA. The Graphical Programming Environment and the Application structure will be derived from the GATE model (REF). The Tango package is also implemented in Java and supports a few unification and type-checking methods, but new optimized unification algorithms will be developed in the future.

1. The Document Manager is implemented in Java.

Acknowledgments Research reported in this paper is supported by the DoD, contract MDA904-96-C-1040.

4 References

Jan Amtrup. Henrik Heine, Uwe lost. 1997. "What's in a Word Graph - Evaluation and Enhancement of Word Lattices". *Eurospeech'97 — Proceedings of the 5th European Conference on Speech Communication and Technology*. Rhodes, Greece.

A. Ballim. 1995. "Abstract Data Types for Multext Tool I/O". LRE 62-05 Deliverable 1.2.1.

Christian Boitet and Mark Seligman. 1994. "The Whiteboard Architecture: a Way to Integrate Heterogeneous Components of NLP Systems". Proceedings of the *15th International Conference on Computational Linguistics - COLING'94,* August 5-9 1994, Kyoto, Japan. pp426-430.

Bill Caid, Jamie Callan, Jim Conley, Harold Robin, Jim Cowie, Kathy DiBella, Ted Dunning, Joe Dzikiewicz, Louise Guthrie, Jerry Hobbs, Clint Hyde, Mark Ilgen, Paul Jacobs, Matt Mettler, Bill Ogden, Peggy Otsubo, Bev Schwartz, Ira Sider, Ralph Weischedel and Remi Zajac. "Tipster Text Phase II Architecture Design and Requirements, Version 2.1". *Proceedings of the Tipster-H 24-month Workshop,* Tysons Corner, VA, 7-10 May, 1996. pp249-305.

H. Cunningham, M. Freeman, W.J. Black. 1994. "Software Reuse, Object-Oriented Frameworks and Natural Language Processing". Proceedings of the *1st Conference on New Methods in Natural Language Processing - NEMLAP-1,* Manchester.

H. Cunningham, Y. Wilks, R. Gaizauskas. 1996. "New Methods, Current Trends and Software Infrastructure for NLP". Proceedings of the *2nd Conference on New Methods in Natural Language Processing - NEMLAP-2,* Ankara, Turkey.

Ralph Grishman, editor. 1995. "Tipster Phase n Architecture Design Document". New-York University, NY, July 1995.

Nigel Sharpies and Philip Bernick. 1996. "A User's Guide to TDM: The CRL TIPSTER Document Manager", CRL Technical Report MCCS-96-298.

Henry Thompson. 1995. "Multext Workpackage 2, Milestone B, Deliverable Overview". LRE 62-050 Deliverable 2.

Remi Zajac, Mark Casper and Nigel Sharpies. 1997. "An Open Distributed Architecture for Reuse and Integration of Heterogeneous NLP Components". Proceedings of the *5th Conference on Applied Natural Language Processing - ANLP'97,* 31 March-3 April, Washington DC. pp245-252.

Remi Zajac. 1992. "Inheritance and Constraint-based Grammar Formalisms". *Computational Linguistics* 18/2, June 1992, pp!59-182.

Extending Grammar Annotation Standards to Spontaneous Speech

Anna RAHMAN
School of Cognitive and Computing Sciences
University of Sussex
Palmer, Brighton, UK. BN1 9QH
annar@cogs.susx.ac.uk

Geoffrey SAMPSON
School of Cognitive and Computing Sciences
University of Sussex
Palmer, Brighton, UK. BN1 9QH
geoffs@cogs.susx.ac.uk

Abstract

We examine the problems that arise in extending an explicit, rigorous scheme of grammatical annotation standards for written English into the domain of spontaneous speech. Problems of principle occur in connexion with part-of-speech tagging; the annotation of speech repairs and structurally incoherent speech; logical distinctions dependent on the orthography of written language (the direct/indirect speech distinction); differentiating between nonstandard usage and performance errors; and integrating inaudible wording into analyses of otherwise-clear passages. Perhaps because speech has contributed little in the past to the tradition of philological analysis, it proves difficult in this domain to devise annotation guidelines which permit the analyst to express what is true without forcing him to go beyond the evidence.

Background

To quote Jane Edwards (1992: 139), "The single most important property of any data base for purposes of computer-assisted research is that *similar instances be encoded in predictably similar ways"*. This principle has not often been observed in the domain of grammatical annotation. Although many alternative lists of grammatical categories have been proposed for English and for other languages, in most cases these are not backed up by detailed, rigorous specifications of boundaries between the categories. A scheme may define how to draw a parse tree for a clear, "textbook" example sentence, with node labels drawn from a large, informative label-alphabet, but may leave it entirely to analysts' discretion how to apply the annotation to the messy constructions that are typical of real-life data.

The SUSANNE scheme, developed over the period 1983-93 Sampson (1995; ftp://ota.ox.ac.uk/pub/ota/susanne/) is a first published attempt to fill this gap for English; the 500 pages of the scheme aim to define an explicit analysis for everything that occurs in the language in practice. No claim is made that the numerous annotation rules comprised in the SUSANNE scheme are "correct" with respect to some psychological or other reality; undoubtedly there are cases where the opposite choice of rule could have yielded an equally well-defined and internally consistent annotation scheme. But, without *some* explicit choice of rules on a long list of issues , one has only a list of category-names and symbols, not a well-defined scheme for applying them.

The SUSANNE scheme has been achieving a degree of international recognition: "the detail ... is unrivalled" Langendoen (1997: 600); "impressive ... very detailed and thorough" Mason (1997: 169, 170); "meticulous treatment of detail" Leech & Eyes (1997: 38). We are not aware of any alternative annotation scheme (for English, or for another language) which covers the ground at a comparable level of detail. (The other schemes that we know about seem to have been initiated substantially more recently than SUSANNE, as well as being less detailed; we do not survey them here.) Various research groups may prefer to use different lists of grammatical symbols, but it is not clear what value will attach to statistics derived from annotated corpora unless the boundaries between their categories are defined with respect to the same issues that the SUSANNE scheme treats explicitly.

Currently, the CHRISTINE project (http://www.cogs.susx.ac.uk/users/ geoffs/RChristine.html) is extending the SUSANNE scheme, which was based mainly on edited written English, to the domain of spontaneous spoken English. CHRISTINE is developing the outline extensions of the SUSANNE scheme for speech which were contained in Sampson (1995: ch. 6) into a set of annotation guidelines comparable in degree of detail to the rest of the scheme, "debugging" them by applying them manually to samples of British English representing a wide variety of regional, social class, age, and social setting variables. Figure 1 displays an extract from the corpus of annotated speech currently being produced through this process. The sources of language samples used by the CHRISTINE project are the speech section of the British National Corpus (http://info.ox.ac.uk/bnc/), the Reading Emotional Speech Corpus (http://midwich.reading.ac.uk/research/ speechlab/emotion/), and the London-Lund Corpus (Svartvik 1990). Figure 1 is extracted from file KSS of the British National Corpus. (Except where otherwise stated, examples quoted in later sections of this paper will also come from the BNC, with the location specified as three-character filename followed after full stop by five-digit "s-unit number". BNC transcriptions include punctuation and capitalization, which are of questionable status in representations of spoken wording; in Figure 1 these matters are normalized away, but they have been allowed to stand in examples quoted in the text below.)

Figure 1 - *. . -

date April 1992, location South Shields,
locale home, activity conversation
PS6RC: f, 72, dial Lancashire,
Salvation Army, educ X, class UU
PS6R9: m, 45, dial Lancashire,
unemployed, educ X, class DE
(sonofPS6RC)

——PS6RC

PPIS1	I	[S[Nea:s.Nea:s]
VVOv	say	[V.V]
PPIS1	I	[Fn:o[Nea:s.Nea:s]
VDO	do	[Ve.
XX	+n't	
VVOv	know	.Ve]
RRQq	where	[Fn?:o[Rq:G101.Rq:G101]
PPHSlf	she	[Nas:s.Nas:s]
VHZ	+'s	[Vzut.
WGK	gon	.Vzut]
TO	+na	[Ti:z[Vi.
VVOv	get	-Vi]
NNln	cake	[Ns:o.Ns:o]
VDN	done	[Tn:j[Vn.Vn]Tn:j]
YG	plOl	.Ti:z]Fn?:o]
RRs	yet	[Rs:t.Rs:t]Fn:o]S]
?	<unclear>	[Y.Y]
PPY	you	[S[Ny:s.Ny:s]
VMo	ca	[Vce.
XX	+n't	-
VVOv	ice	.Vce]
ATI	a	[Ns:o.Ns:o]
YR	#	-
WOv/icei		[V[VVOv#.
YR	#	-
WOv	ice	.VVOv#]V]
ATI	a	[Ns:o.
NNln	cake	.Ns:o]
CSi	if	[Fa:c.
PPY	you	[Ny:s.Ny:s]
VHO	have	[Vef.
XX	+n't	
VVNv	got	.Vef]
MCI	one	[Ms:o.Ms:o]Fa:c]S]

In Figure 1, the words uttered by the speakers are in the next-to-rightmost field. The field to their left classifies the words, using the SUSANNE tagset supplemented by some additional refinements to handle special problems of speech: the part-word i at byte 0692161 is tagged "VVOv/ice" to show that it is a broken-off attempt to utter the verb *ice*. The rightmost field gives the grammatical analysis of the constructions, in the form of a labelled tree structure which again uses the SUSANNE conventions. All tagmas are classified formally, with a capital letter followed in some cases by lower-case subcategory letters: S stands for "main clause", Nea represents "noun phrase marked as first-person-singular and subject", Ve labels *don't know* as a verb group marked as negative. Additionally, immediate constituents of clauses are classified functionally, by a letter after a colon: Nea:s in the first line shows that is the subject of *say*, Fn:o in the third line shows that the nominal clause (Fn) *I don't know where ...* is direct object of *say*. Three-digit index numbers relate surface to logical structures. Thus *where* in the seventh line is marked as an interrogative adverb phrase (Rq) having no logical role (:G) in its own clause (that is, the clause headed by *+'s gon*, i.e. *is going)*, but corresponding logically to an unspoken Place adjunct ("plOl") within the infinitival clause *+na* (= *to) get cake done*. (The character "y" in column 3 identifies a line which contains an element of the structural analysis rather than a spoken word.)

Legal constraints permitting, the CHRISTINE Corpus will be made freely available electronically, after completion in December 1999, in the same way as the SUSANNE Corpus already is.

Defining a rigorous, predictable structural annotation scheme for spontaneous speech involves a number of difficulties which are not only additional to, but often different in kind from, those involved in defining such a scheme for written language. This paper examines various of these difficulties. In some cases, our project has already identified tentative annotation rules for addressing these difficulties, and in these cases we shall mention the decision adopted; but in other cases we have not yet been able to formulate any satisfactory solution. Even in cases where our project has chosen a provisional solution, discussing this is not central to our aims in the present paper. Our goal, rather, is to identify the types of issue needing to be resolved, and to show how devising an annotation scheme for speech involves problems of principle, of a kind that would have been difficult to anticipate before undertaking the task.

1 The Software Engineering Precedent

The following pages will examine a number of conceptual problems that arise in defining rigorous annotation standards for spontaneous speech. Nothing will be said about computational technicalities, for instance the possibilities of designing an automatic parser that could apply such annotation, or the nature of the software tools used in our project to support manual annotation. (The project has developed a range of such tools, but we regard them as being of interest only to ourselves.)

In our experience, some computational linguists see a paper of this type as insubstantial and of limited value in advancing the discipline. While it is not for us to decide the value of our particular contribution, as a judgement on a genre we see this attitude as profoundly wrong-headed. To explain why, let us draw an analogy with developments in industrial and commercial computing.

Writing programs and watching them running is fun. Coding and typing at keyboards are the programmer activities which are most easy for IT managers to perceive as productive. For both these reasons, in the early decades of computing it was common for software developers to move fairly quickly from taking on a new assignment to drafting code — though, unless the assignment was trivially simple, the first software drafts did not work. Sometimes they could be rescued through debugging — usually a great deal of debugging. Sometimes they could not: the history of IT is full of cases of many-man-year industrial projects which eventually had to be abandoned as irredeemably flawed without ever delivering useful results.

There is nowadays a computer science subdiscipline, software engineering e.g. Sommerville (1992), which has as one of its main aims the training of computing personnel to resist their instincts and to treat coding as a low priority. Case studies have shown Boehm (1981: 39-41) that the cost of curing programming mistakes rises massively, depending how late they are caught in the process that begins with analysing a new programming task and ends with maintenance of completed software. In a well-run modern software house, tasks and their component subtasks are rigorously documented at progressively more refined levels of detail, so that unanticipated problems can be detected and resolved before a line of code is written;

programming can almost be described as the easy bit at the end of a project.

The subject-matter of computational linguistics, namely human language, is one of the most complex phenomena dealt with by any branch of IT. To someone versed in modern industrial software engineering, which mainly deals with structures and processes much simpler than any natural language, it would seem very strange that our area of academic computing research could devote substantially more effort to developing language-processing software than to analysing in detail the precise specifications which software such as natural-language parsers should be asked to deliver, and to uncovering hidden indeterminacies in those specifications. Accordingly, we make no bones about the data-oriented rather than technique-oriented nature of the present paper. At the current juncture in computational linguistics, consciousness-raising about problematic aspects of the subject-matter is a high priority.

2 Wordtagging

One fundamental aspect of grammatical annotation is classifying the grammatical roles of words in context — wordtagging. The SUSANNE scheme defined an alphabet of over 350 distinct wordtags for written English, most of which are equally applicable to the spoken language though a few have no relevance to speech (for instance, tags for roman numerals, or mathematical operators). Spoken language also, however, makes heavy use of "discourse items" Stenstrom (1990) having pragmatic functions with little real parallel in writing: e.g. *well* as an utterance initiator. Discourse items fall into classes which in most cases are about as clearly distinct as the classifications applicable to written words, and the CHRISTINE scheme provides a set of discourse-item wordtags developed from Stenstrom's classification. However, where words are ambiguous as between alternative discourse-item classes, the fact that discourse items are not normally syntactically integrated into wider structures means that there is little possibility of finding evidence to resolve the tagging ambiguity.

Thus, three discourse-item classes are Expletive (e.g. *gosh),* Response (e.g. *ah),* and Imitated Noise (e.g. *glug glug).* Consider the following extracts from a sample in which children are "playing horses", one riding on the other's back:

KPC.00999-1002 speaker PS 1DV: ... *all you can do is <pause> put your belly up and I'll go flying! ... Go on then, put your belly up!* speaker PS 1DR: *Gung!*
KPC. 10977 *Chuck a chuck a chuck chuck! Ee eel Go on then.*

In the former case, *gung* is neither a standard English expletive, nor an obviously appropriate vocal imitation of anything happening in the horse game. Conversely, in the latter case *ee* could equally well be the standard Northern regional expletive expressing mildly shocked surprise, or a vocal imitation of a "riding" noise. In many such cases, the analyst is forced by the current scheme to make arbitrary guesses, yet clear cases of the discourse-item classes are too distinct from one another to justify eliminating guesswork by collapsing the classes into one.

Not all spoken words posing tagging problems are discourse items. In:

KSU.00396-8 *Ah ah! Diddums! Yeah.* , -

any English speaker will recognize the word *diddums* as implying that the speaker regards the hearer as childish, but intuition does not settle how the word should be tagged (noun? if so, proper or common?); and published dictionaries do not help. To date we have formulated no principled rule for choosing an analysis in cases like these.

3 Speech Repairs

Probably the most crucial single area where grammatical standards developed for written language need to be extended to represent the structure of spontaneous spoken utterances is that of speech repairs. The CHRISTINE repair annotation system draws on Levelt (1983) and Howell & Young (1990, 1991), to our knowledge the most fully-worked-out and empirically-based previously existing approach.

This approach identified a set of up to nine repair milestones within a repaired utterance, for instance the point at which the speaker's first grammatical plan is abandoned (the "moment of interruption"), and the earlier point marking the beginning of the stretch of wording which will be replaced by new wording after the moment of interruption. However, this approach is not fully workable for many real-life speech repairs. In one respect it is insufficiently informative: the Levelt/Howell & Young notation provides no means of showing how a local sequence containing a repair fits into the larger grammatical architecture of the utterance containing it. In other respects, the notation proves to be excessively rich: it requires speech repairs to conform to a canonical pattern from which, in practice, many repairs deviate.

Accordingly, CHRISTINE embodies a simplified version of this notation, in which the "moment of interruption" in a speech repair is marked (by a "#" sign within the stream of words), but no attempt is made to identify other milestones, and the role of the repaired sequence is identified by making the "#" node a daughter of the lowest labelled node in a parse tree such that both the material preceding and the material following the # are (partial) attempts to realize that category, and the mother node fits normally into the surrounding structure. This approach works well for the majority of speech repairs, e.g.:

KBJ.00943 *That's why I said* [Ti:o *to get ma ba* #, *get you back then]* ...
KCA.02828 *I'll have to* [VVOv# *cha # change*] *it*

"A:'

In the KBJ case, *to get ma ba* (in which *ma* and *ba* are truncated words, the former identified by the wordtagging as too distorted to reconstruct and the latter as an attempt at *back* as an adverb), and *get you back then,* are successive attempts to produce an infinitival clause (Ti) functioning as object (:o) of *said.* In the KCA case, *cha* and *change* are successive attempts to produce a single word whose wordtag is WOv (base form of verb having transitive and intransitive uses). In Figure 1, the "#" symbol is used at two levels in the same speaker turn: speaker PS6RC makes two attempts to realize a main clause (S), and the second attempt begins with two attempts to pronounce the verb *ice.*

However, although the CHRISTINE speech-repair notation is less informative than the full Levelt/Howell & Young scheme, and seems as simple as is consistent with offering an adequate description of repair structure, applying it consistently is not always straightforward. In the first place, as soon as the annotation scheme includes any system for marking speech repairs, analysts are obliged to decide whether particular stretches of wording are in fact repairs or well-formed constructions, and this is often unclear. Sampson (1998) examined a number of indeterminacies that arise in this area; one of these is between repairs and appositional structures, as in:

KSS.05002 *she can't be much cop ifshe 'd open her legs to a first date to a Dutch s- sailor*

— where *to a Dutch s- sailor* might be intended to replace *to a first date* as the true reason for objecting to the girl, but alternatively *to a Dutch s- sailor* could be an appositional phrase giving fuller and better particulars of the nature of her offence. Annotation ought not systematically to require guesswork, but it is hard to see how a neutral notation could be devised that would allow the analyst to suspend judgment on such a fundamental issue as whether a stretch of wording is a repair or a well-formed construction.

Even greater problems are posed by a not uncommon type of ill-formed utterance that might be called "syntactically Markovian", in which each element coheres logically with what immediately precedes but the utterance as a whole is not coherent. The following examples come from the London-Lund Corpus, with text numbers followed by first and last tone-unit numbers for the respective extracts:

S.I.3 0901-3 ... *ofcourse I would be willing to urn <pause => come into the common-room <pause => and uh <pause——> in fact I would like nothing I would like better* [speaker is undergraduate, age ca

36, describing interview for Oxbridge fellowship]

S.5.5 0539-45 *and what is happening <pause=> in Britain today <pause -> is ay- demand for an entirely new foreign policy quite different from the cold war policy <pause => is emerging from the Left* [speaker is Anthony Wedgwood Benn MP on radio discussion programme]

In the former example, *nothing* functions simultaneously as the last uttered word of an intended sequence / *would like nothing better* and the first uttered word of an implied sequence something like *there is nothing I would like better.* In the latter, the long NP *an entirely new foreign policy quite different from the cold war policy* appears to function both as the complement of the preposition *for,* and as subject of *is emerging.* In such cases one cannot meaningfully identify a single point where one grammatical plan is abandoned in favour of another. Because these structures involve phrases which simultaneously play one grammatical role in the preceding construction and a different role in the following construction, they resist analysis in terms of tree-shaped constituency diagrams (or, equivalently, labelled bracketing of the word-string). Yet constituency analysis is so solidly established as the appropriate formalism for representing natural-language structure in general that it seems unthinkable to abandon it merely in order to deal with one special type of speech repair.

4 Logical Distinctions Dependent on the Written Medium

There are cases where grammatical category distinctions that are highly salient in written English seem much less significant in the spoken language, so that maintaining them in the annotation scheme arguably misrepresents the structure of speech. Probably the most important of these is the direct/indirect speech distinction. Written English takes great pains to distinguish clearly between direct speech, involving a commitment to transmit accurately the quoted speaker's exact wording, and indirect speech which preserves only the general sense of the quotation. The SUSANNE annotation scheme uses categories which reflect this distinction (Q v. Fn). However, the most crucial cues to the distinction are orthographic matters such as inverted commas, which lack spoken counterparts. Sometimes the distinction can be drawn in spoken English by reference to pronouns, verb forms, vocatives, etc.:

KD6.03060 *... he says he hates drama because the teacher takes no notice, he said one week Stuart was hitting me with a stick and the teacher just said calm down <u>you boys</u> ...*

— the underlined *he* (rather than /) implies that the complement of *says* is indirect speech; *me* implies that the passage beginning *one week* is a direct quotation, and the imperative form *calm* and vocative *you boys* imply that the teacher is quoted directly. But in practice these cues frequently conflict rather than reinforcing one another:

KCT. 10673 [reporting speaker's own response to a directly-quoted objection]: */ said <u>well</u> that^s <u>his</u> hard luck!*

KCJ .01053-5 *well Billy, Billy says <u>well take</u> that and then he 'II come back and then he er gone and pay that*

In the KCT example, the discourse item *well* and the present tense of *fijs* after past-tense *said* suggest direct speech, but *his* (which from the context denotes the objector) suggests indirect speech. Likewise in the KCJ example, *well* and the imperative *take* imply direct speech, *he'll* rather than *I'LL* implies indirect speech. Arguably, imposing a sharp two-way direct v. indirect distinction on speech is a distortion; one might instead feel that speech uses a single construction for reporting others' utterances, though different instances may contain more or fewer indicators of the relative directness of the report. On the other hand, logically speaking the direct v. indirect speech distinction is so fundamental that an annotation scheme which failed to recognize it could seem unacceptable. (To date, CHRISTINE analyses retain the distinction.)

5 Nonstandard Usage

Real-life British speech contains many differences from standard usage with respect to both individual words and syntactic patterns.

In the case of wordtagging, the SUSANNE rule (Sampson 1995: §3.67) is that words used in ways characteristic of nonstandard dialects are tagged in the same way as the words that would replace them in standard English. This rule tends to be unproblematic for pronouns and determiners, thus in:

KP4.03497 it's *a bit offun, it livens up me day*
KCT. 10705 *she told me to have them plums*

the underlined words are given the tags for standard *my, those* respectively. It is more difficult to specify a predictable way to apply such a rule in the case of nonstandard uses of strong verb forms. Standard base forms can be used in past contexts, e.g.:

KCJ.01096-8 *a man bought a horse and give it to her, now it's won the race*

and the solution of tagging such an instance as a past tense is put into doubt because frequently nonstandard English omits the auxiliary of the standard perfective construction, suggesting that *give* might be tagged as *given* rather than *gave;* cf.:

KCA.02536 *What I done. I taped it back like that.*
KCA.02572 *What it is, when you sot snooker on and just snooker you're quite <pause> content to watch it...*

(Note that *done* might be seen as equivalent to standard *did*, rather than to standard *have done;* but *got* meaning "have" is not plausibly analysed as other than a perfective construction.) It is quite impractical for annotation to be based on fully adequate grammatical analyses of each nonstandard dialect in its own terms; but it is not easy to specify consistent rules for annotating such uses as deviations from the known, standard dialect. The CHRISTINE project has attempted to introduce predictability into the analysis of these cases, by recognizing a nonstandard-English "tense" realized as past participle not preceded by auxiliary, and by ruling that any verb form used in a nonstandard structure with past reference will be classified as a past participle (thus *give* in the KCJ example above is classified as a nonstandard equivalent of *given).* This approach does work well for many cases, but it remains to be seen whether it deals satisfactorily with all the usages that arise.

At the syntactic level, an example of a nonstandard construction requiring adaptation of the written-English annotation scheme would be relative clauses containing both relative pronoun and undeleted relativized NP, unknown in standard English but usual in various nonstandard dialects, e.g.:

KD6.03075 ... *bloody Colin who, he borrowed his computer that time, remember?*

Here the CHRISTINE decision is to treat the relativized NP *(he}* as appositional to the relative pronoun. For the case quoted, this works; but it will not work if a case is ever encountered where the relativized element is not the subject of the relative clause. Examples like this raise the question what it means to specify consistent grammatical annotation standards applicable to a spectrum of different dialects, rather than a single dialect. Written English usually conforms more or less closely to the norms of the national standard language, so that grammatical dialect variation is marginal and annotation standards can afford to ignore it. In the context of speech, it cannot be ignored, but the exercise of specifying annotation standards for unpredictably varying structures seems conceptually confused.

6 Dialect Difference v. Performance Error

Special problems arise in deciding whether a turn of phrase should be annotated as well-formed with respect to the speaker's nonstandard dialect, or as representing standard usage but with words elided as a performance error. Speakers often do omit necessary words, e.g.:

KD2.03102-3 *There's one thing I don't like <pause> and that's having my photo taken. And it will be hard when we have to photos.*

— it seems safe to assume that the speaker intended something like *have to show photos.* One might take it that a similar process explains the underlined words in:

KD6.03154 *oh she was shouting at him at dinner time <shift shouting> Steven <shift> oh god dinner time she was shouting him.*

where *at* is missing; but this is cast in doubt when other speakers, in separate samples, are found to have produced:

KPC.00332 *go in the sitting room until I shout you for tea*
KD2.02798 *The spelling mistakes only occurred when <pause> I was shouted.*

— this may add up to sufficient evidence for taking *shout* to have a regular transitive use in nonstandard English.

This problem is particularly common at the ends of utterances, where the utterance might be interpreted as broken off before it was grammatically complete (indicated in the SUSANNE scheme by a "#" terminal node as last daughter of the root node), but might alternatively be an intentional nonstandard elision. In:

KE2.08744 *That's right, she said Margaret never goes, I said well we never go for lunch out, we hardly ever really*

the words *we hardly ever really* would not occur in standard English without some verb (if only a placeholding *do),* so the sequence would most plausibly be taken as a broken-off utterance of some clause such as *we hardly ever really go out to eat at all;* but it is not difficult to imagine that the speaker's dialect might allow *we hardly ever really* for standard *we hardly ever do really,* in which case it would be misleading to include the "#" sign.

It seems inconceivable that a detailed annotation scheme could fail to distinguish difference of dialect from performance error; indeed, a scheme which ignored this distinction might seem offensive. But analysts will often in practice have no basis for applying the distinction to particular examples.

7 Transcription Inadequacies

One cannot expect every word of a sample of spontaneous speech recorded in field conditions to be accurately transcribable from the recordings. Our project relies on transcriptions produced by other researchers, which contain many passages marked as "unclear"; the same would undoubtedly be true if we had chosen to gather our own material. A structural annotation system needs to be capable of assigning an analysis to a passage containing unclear segments; to discard any utterance or sentence containing a single unclear word would require throwing away too many data, and would undesirably bias the retained collection of samples towards utterances that were spoken carefully and may therefore share some special structural properties.

The SUSANNE scheme uses the symbol Y to label nodes dominating stretches of wholly unclear speech, or tagmas which cannot be assigned a grammatical category because they contain unclear subsegments that make the categorization doubtful. This system is unproblematic, so long as the unclear material in fact consists of one or more complete grammatical constituents. Often, however, this is not so; e.g.:

KCT. 10833 *Oh we didn't <unclear> to drink yourselves.*

Here it seems sure that the unclear stretch contained multiple words, beginning with one or more words that complete the verb group (V) initiated by *didn't;* and the relationship of the words *to drink yourselves* to the main clause could be quite different, depending what the unclear words were. For instance, if the unclear words were *give you anything,* then *to drink* would be a modifying tagma within an NP headed by *anything;* on the other hand, if the unclear stretch were *expect you*, then *to drink* would be the head of an object complement clause. Ideally, a grammatical annotation scheme would permit all the clear grammar to be indicated, but allow the analyst to avoid implying any decision about unresolvable issues such as these. Given that clear grammar is represented in terms of labelled bracketing, however, it is very difficult to find usable notational conventions that avoid commitment about the structures to which unclear wording contributes.

Conclusion

In annotating written English, where one is drawing on an analytic tradition evolved over centuries, it seems on the whole to be true that most annotation decisions have definite answers; where some particular example is vague between two categories, these tend to be subcategories of a single higher-level category, so a neutral fallback annotation is available. (Most English noun phrases are either marked as singular or marked as plural, and the odd exceptional case such as *the fish* can at least be classified as a noun phrase, unmarked for number.) One way of summarizing many of the problems outlined in the preceding sections is to say that, in annotating speech, whose special structural features have had little influence on the analytic tradition, ambiguities of classification constantly arise that cut across traditional category schemes. In consequence, not only is it often difficult to choose a notation which attributes specfic properties to an example; unlike with written language, it is also often very difficult to define fallback notations which enable the annotator to avoid attributing properties for which there is no evidence, while allowing what can safely be said to be expressed.

Some members of the research community may be tempted to feel that a paper focusing on these problems ranks as self-indulgent hand-wringing in place of serious effort to move the discipline forward. We hope that our earlier discussion of software engineering will have shown why that feeling would be misguided. Nothing is easier and more appealing than to plunge into the work of getting computers to deliver some desired behaviour, leaving conceptual unclarities to be sorted out as and when they arise. Huge quantities of industrial resources have been wasted over the decades through allowing IT workers to adopt that approach. Natural language processing was one of the first application areas ever proposed for computers (by Alan Turing in 1948 — Hodges 1983: 382); fifty years later, the level of success of NLP software (while not insignificant) does not suggest that computational linguistics can afford to go on ignoring lessons that have already been painfully learned by more central sectors of the IT industry.

Effort put into automatic analysis of natural language implies a prior requirement for serious effort devoted to defining and debugging detailed standard schemes of linguistic analysis. Our SUSANNE and CHRISTINE projects have been and are contributing to this goal, but they are no more than a beginning. We urge other computational linguists to recognize this area as a priority.

Acknowledgment

The research reported here was supported by grant R000 23 6443, "Analytic Standards for Spoken Grammatical Performance", awarded by the Economic and Social Research Council (UK).

References

Boehm B.W. (1981) *Software Engineering Economics.* Prentice-Hall, Engelwood Cliffs, N.J.

Edwards Jane A. (1992) *Design principles in the transcription of spoken discourse.* In , "Directions in Corpus Linguistics", J. Svartvik, ed., Mouton de Gruyter, Berlin.

Hodges A. (1983) *Alan Turing: The Enigma of Intelligence.* Burnett Books, London.

Howell P. & K. Young (1990) *Speech repairs: report of work conducted October 1st 1989-March 31st 1990.* Department of Psychology, University College London.

Howell P. & Young K. (1991) *The use of prosody in highlighting alterations in* *repairs from unrestricted speech.* Quarterly Journal of Experimental Psychology 43A,pp. 733-758.

Langendoen D.T. (1997) Review of Sampson (1995). Language 73. pp.600-603.

Leech G.N. & Eyes Elizabeth (1997) *Syntactic annotation: treebanks* In Ch. 3 "Corpus Annotation", R.G. Garside et al., eds., Longman, Harlow, Essex.

Levelt W.J.M. (1983) *Monitoring and self-repair in speech.* Cognition , 14, pp.41-104.

Mason O. (1997) Review of Sampson (1995). International Journal of Corpus Linguistics, 2.1, pp.69-72.

Sampson G.R. (1995) *English for the Computer.* Clarendon Press, Oxford.

Sampson G.R. (1998) *Consistent annotation of speech-repair structures.* In "Proceedings of the First International Conference on Language Resources and Evaluation, Granada, May 1998"

Sommerville I. (1992) *Software Engineering* (4th ed.). Addison-Wesley, Wokingham, Berks.

Stenstrom Anna-Brita (1990) *Lexical items peculiar to spoken discourse.* In Svartvik (1990).

Svartvik J., ed. (1990) *The London-Lund Corpus of Spoken English.* Lund University Press.