The Annotation Process in the Turkish Treebank

Nart B. Atalay
Informatics Inst.
Middle East Technical Univ.
Ankara, Turkey
bedin@ii.metu.edu.tr

Kemal Oflazer
Fac. of Eng. and Natural Sci.
Sabancı Univ.
İstanbul, Turkey
oflazer@sabanci.edu

Bilge Say
Informatics Inst.
Middle East Technical Univ.
Ankara, Turkey
bsay@ii.metu.edu.tr

Abstract

We present a progress report of the Turkish Treebank concentrating on various aspects of its design and implementation. In addition to a review of the corpus compilation process and the design of the annotation scheme, we describe the details of various pre-processing stages and the computer-assisted annotation process.

1 Introduction

Recent years have seen an increase in the construction of treebank resources for a variety of languages and some of the treebank compilations have used an annotation scheme employing a dependency grammar representation (Abeillé, 2003; Hinrichs and Simov, 2002). The Turkish Treebank Project (Oflazer et al., 2003; Say et al., 2002) is a treebank compilation project comprising written Turkish samples, with full morphological and surface dependency annotation.

In Section 2, we review the relevant aspects of the METU Turkish Corpus, which is the source of the text samples for the treebank. In Section 3, the design of the annotation scheme is reviewed briefly, followed by a detailed account of the morphosyntactic preprocessing in the treebank in Section 4. In Section 5, we present the annotation tool that is used for assisting the manual disambiguation of morphological analyses and the annotation of dependency relations. Finally, the current status of the treebank is reviewed.

2 Corpus Description

The Turkish Treebank is a subcorpus of the METU Turkish Corpus, which is a 2 million word corpus of post-1990 written Turkish, sampled from various genres (Say et al., 2002). No representativeness scheme was applied statistically, but care was taken to balance the corpus across genres (approximately 16 main genres) and authors to the extent made possible by the copyright permission processes. The corpus has 2000-word samples. The samples are annotated in conformance with XCES, short for XML-based Corpus Encoding Standard which is based on TEI (Text Encoding Initiative) guidelines (Sperberg-McQueen and Burnard, 1994; Ide, 1996). The annotation scheme applied to the whole corpus is conformant with XCES on typographic-general level which includes tags for bibliographical information, paragraphs, lists, highlighted items, etc. Human error is minimized by using in-house developed annotation software, an XCES editor and annotation control procedures. The corpus will be announced with a supporting query workbench, which is currently under development and includes searching for terms and with regular expressions, along with various saving, and viewing options.

The Turkish Treebank is aimed at including 10,000 sentences that are annotated with morphological and syntactic annotations. The following sections describe the details of the annotation scheme and the annotation process. We include *whole* samples from the METU Turkish Corpus in the treebank and try to representatively select samples from the genres present in the main corpus.

3 Design of the Annotation Scheme

The treebank corpus has detailed annotations at both the lexical level and the syntactic level. At the lexical level we annotate each token with its unambiguous morphological analysis (though multiword collocations are treated a bit differently). These morphological analyses are encoded using an extensive set of inflectional, derivational and, where relevant, morphosemantic features (see (Oflazer et al., 2003) for the details of such features.) Since Turkish is an agglutinative language with very productive derivational phenomena, derivations are marked explicitly as they get involved in syntactic relations: groups of inflectional features separated by derivational markers are treated as full-fledged syntactic units as we will see below.

At the syntactic level, surface syntactic relations are encoded using a set of dependency relationships. Our rationale for choosing this representation was elaborated earlier (Oflazer et al., 2003). Even though we have encountered some complications when such representation schemes were applied to real sentences in corpora we have made principled amendments to the representation to alleviate almost all problems encountered. We have completed an annotation manual that guides people not familiar with the annotation scheme to annotate sentences or interpret annotations (Say and Oflazer, 2002).

Currently the following surface syntactic relations are used to annotate the dependency links:

1. Sentence
2. Subject
3. Object
4. Modifier
5. Modifier
6. Determiner
7. Focus-Particle
8. Question-Particle
9. Vocative
10. Classifier
11. Dative Adjunct
12. Ablative Adjunct
13. Locative Adjunct
14. Instrum. Adjunct
15. Coordination
16. Relativizer
17. Possessive

In Figure 1 the following sentence is shown with dependency links:

Bu	çocuk	okuldan	erken	geldi.
This	child	school+Abl	early	come+Past+3sg.

This child came from the school early.

The final period is considered to be the head of the whole sentence and the direction of the arrow is from the dependent to the head.

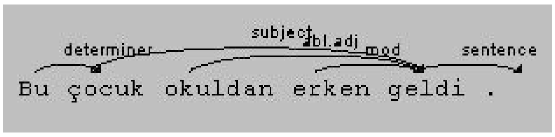


Figure 1: Surface Dependency Relations of an Example Sentence

4 Morphosyntactic Preprocessing

Since our representation scheme relies extensively on proper and unambiguous identification of morphological features, we use a full-scale morphological analyzer for Turkish (Oflazer, 1993) built using Xerox Research Centre Finite State Toolkit (Karttunen and Beesley, 2003). We then use a postprocessor that identifies various non-lexicalized and lexicalized collocations, so that dependency relations involving the components of such collocations can be more perspicuously expressed.

4.1 Morphological Analysis

The morphological analyzer takes tokens coming from a tokenizer module and produces all possible morphological interpretations of the token. The general representation of analyses comprises a sequence of *inflectional groups* separated by derivational boundary markers.

$$\texttt{root+Infl}_1\texttt{\^{}DB+Infl}_2\texttt{\^{}DB+}\cdots\texttt{\^{}DB+Infl}_n$$

where $\texttt{Infl}_i$ denote relevant inflectional features including the part-of-speech for the root or any of the forms. For instance, the simple derived modifier *evdeki* (... that is at the house) will have the morphological analysis

```
ev+Noun+A3sg+Pnon+Loc+^DB+Adj+Rel
```

and would be represented by a sequence 2 inflectional groups:

1. `ev+Noun+A3sg+Pnon+Loc` 2. `+Adj+Rel`

The grammatical role of this word as a modifier is determined by the 2_{nd} inflectional group while any preceding modifiers (of *ev* (house)) to the left of this token may link to the first inflectional group.

The morphological analysis process is very fast and can process thousands of tokens per second. The morphological analysis process also employs an unknown word processor but any tokens still not given an analysis could be manually annotated later.

4.2 Preprocessing of Collocations

A collocation, for our purposes, is defined as a group of lexical items that would better be considered as a single unit in terms of the dependency relations in sentences. In regard of this definition, collocations can be classified into the following groups:

- word sequences with certain patterns that can be generated productively according to certain rules (henceforth non-lexicalized collocations),
- proper nouns that are composed of more than one word,
- token sequences that express date or time,
- idiomatic word sequences with specific usage whose semantics is non-compositional,
- compound verb forms which are formed by a lexically adjacent, direct or oblique object and a verb.

The merging of a group of lexemes into a single one and the attachment of morphological analysis to the resultant lexeme is carried by two post-processing processes that follow the morphological analysis process.

The first process handles non-lexicalized collocations. Non-lexicalized collocations are sequences of 2 or 3 tokens which almost-always involve some form of partial or full reduplication of word forms.[1] For instance, the marked verb sequence *gelir gelmez* (`gel+Verb+Pos+Aor+A3sg gel+Verb+Neg+Aor+A3sg`) actually functions as a temporal adverbial meaning *as soon as ... comes*.

The postprocessor takes the morphologically analyzed token sequence as input. The collocational rules that will be operated on corpus can be entered interactively or can be given as a configuration file to the program. The program applies rules one by one and generates two output files. One is collocationally preprocessed corpus file and the other is the list of collocations in the corpus that were processed. For instance, for non-lexicalized collocations like above (for which the verbal root can be any verb), the postprocessor rule searches for two adjacent tokens whose possibly ambiguous morphological analyses have analyses matching the patterns specified: the first token would have an aorist marker with positive polarity and the second token would have the same root, also an aorist marker but a negative polarity. These tokens are then coalesced into a single token *gider_gitmez* with the new morphological analysis `git+Verb+Pos^DB+Adverb+AsSoonAs`.

It can be noted that any modifiers and the complements of the verb *git*, preceding the collocation will link to the first inflectional group while this collocation will link as a modifier to a verb via the second inflectional group. Thus the overall representational scheme is maintained.

Currently the following nonlexicalized collocations are implemented:

- Duplicated optative and 3sg verbs acting as manner adverb (e.g., koşa koşa), meaning "by .. *verb*-ing",
- duplicated verbal and derived adverbial forms from the same root acting as a temporal adverb (e.g., bildi bileli) meaning "ever since .. *verb*-ed",
- duplicated adjectives acting as a manner adverb (e.g., güzel güzel), meaning "*adj*-ly",
- duplicated nouns acting as manner adverb (e.g., ev ev), meaning "*noun* by *noun*"

A follow-up process similarly handles the semi-lexicalized collocations where one of the tokens is variable while the other is fixed, and lexicalized collocations.

[1] Note that these formations (usually involving full or partial reduplications of strings of the sort $\omega\,\omega$, $\omega\,x\,\omega$ or $\omega x\,\omega z$) are beyond the formal power of finite state mechanisms hence are not dealt within the finite state morphological analyzer.

4.3 Syntactic Preprocessing

Although the main burden of dependency relation annotation process was and is still carried out by human annotators, our annotation tool provides some simple syntactic preprocessing which attempts to identify automatically certain relations with a very high precision. This preprocessor uses heuristic rules that identifies relations between inflectional groups and automatically tags the relations. If the annotator disagrees with the annotations, s/he can modify the relations later in the annotation tool.

The linking rules implemented exploit the fact that the functions of nominal inflectional groups are essentially determined by the case marking features. The following heuristic rules are implemented:

- If a word final inflectional group has an accusative case marker and a suitable postposition follows it immediately, then the nominal inflectional group is linked to the postposition with the label OBJECT.
- If the next token is not a postposition then, we locate a verbal inflectional group to the right and link the noun as an OBJECT of the nearest of such verbal inflectional group.
- Similarly a dative case-marked nominal word final inflectional group can link to a suitable postposition that immediately follows it, as an OBJECT and if no postposition is found then it can link as a DATIVE ADJUNCT to a verbal inflectional group.
- A genitive case marked nominal word final inflectional group can link to a following noun as a POSSESSOR provided some additional agreement criteria are met. If this is not possible but an infinitive or participle nominal inflectional group is found, the genitive case-marked inflectional group is linked as the SUBJECT of the (subordinate clause headed by the) verb preceding the infinitive or the participle inflectional group.

The dependency relations that are handled in the syntactic preprocessing phase appear to annotators as default links. Annotators check these links and report errors that are encountered regarding the tags. This process enables refining of the rules.

5 Tagging with the Annotation Tool

To facilitate the tagging process for annotators, an annotation tool has been implemented in Visual Basic. This program helps annotators to easily tag dependency relations by enabling visual browsing and marking. The input of the program is the morphologically analyzed and preprocessed text. The output is a morphologically disambiguated and annotated treebank which is encoded according to XML document format. The tagging process requires two steps: morphological disambiguation and dependency tagging.

The first task of an annotator is to disambiguate morphological analysis (see Figure 2). The word and its morphological analyses are shown to the user in this screen. Morphological analyses are presented in text boxes and the user selects the appropriate analysis by clicking on the relevant check box. If some analyses are incorrect, the annotator can edit them by typing into the text boxes. After all words in a sentence are morphologically disambiguated, the annotator can pass to the second phase of annotation.

After finishing the disambiguation process, the user directs the program to the dependency screen (see Figure 3). In this part, the word, its positional index and the selected morphological analysis of the word are shown to the user. Three combo boxes are attached to each word. These combo boxes enable user to select the head word, the head derivational boundary and the type of the dependency link, respectively.[2] After dependency relation annotation is completed, the user saves his work in the output format.

Both disambiguation and dependency tagging screen includes a text box for taking notes. It is possible to modify the list of the dependency tags.

[2] It is possible to add a covert element (NULL) to the sentence where necessitated by the sentence structure, e.g. elliptical sentences involving coordination.

6 Current Status and Difficulties Encountered

Currently, the preprocessing and the annotation tool software are completed and tested, and we are halfway through annotating 10,000 sentences. The integration of treebank search into the query workbench is also under development. A viewing tool that allows the user to see the selected morphosyntactically annotated sentences as dependency diagrams has also been developed to be integrated into the query workbench.

As expected we have encountered numerous difficulties in all these processes. They fall into three different categories: First, agreement on the annotation scheme so that it has large coverage but is still practical to apply, has been difficult. We have left out some tags, such as thematic role tags, that at the outset we thought could be included. Second, the development of the supporting annotation software took more time than planned, mainly due to project personnel turnover and financial support difficulties. Finally, the manual annotation process has as expected proved to be quite time consuming and the development and testing of the syntactic preprocessing part which speeds up the process had to wait for the accumulation of substantial test data.

7 Conclusions and Future Work

To our knowledge, the Turkish treebank is the first sizeable effort to develop a principled treebank for Turkish with accompanying annotation support tools and search software. There are, though, several items on our wish list, which might be formed into further projects: one is to increase the number of tagged sentences in the treebank. Additional and alternative annotation schemes applied on the same subcorpus of the METU Turkish corpus will also be valuable: with more involvement from linguists working on Turkish, we could add to existing annotation, for example, tags for information structure. Alternative annotation schemes adhering to different grammar formalisms will enable comparative evaluation of the syntactic structure both for research in linguistics and computational linguistics.

Acknowledgements

We would like to express our gratitude to TÜBİTAK (Project No: EEEAG 199E026) for their financial support. Many thanks to Barış Demiral, Hacer Uke, Dilek H. Tür, Gökhan Tür, İrfan Nuri Karaca, Çağrı Kayadelen, Umut Özge and participants of METU Turkish Corpus Project for their contributions to various phases of this project.

References

Anne Abeillé, editor. 2003. *Building and Exploiting Syntactically-annotated Corpora*. Kluwer Academic Publishers.

Erhard Hinrichs and Kiril Simov, editors. 2002. *Proceedings of the First Workshop on Treebanks and Linguistic Theories*. Available on the WWW as: http://www.BulTreeBank.org/Proceedings.html.

Nancy Ide, 1996. *Corpus Encoding Standard: Document CES 1, Version 1.4, October*. http://www.cs.vassar.edu/CES/.

Lauri Karttunen and Kenneth R. Beesley. 2003. *Finite-State Morphology: Xerox Tools And Techniques*. CSLI Publications. Stanford University.

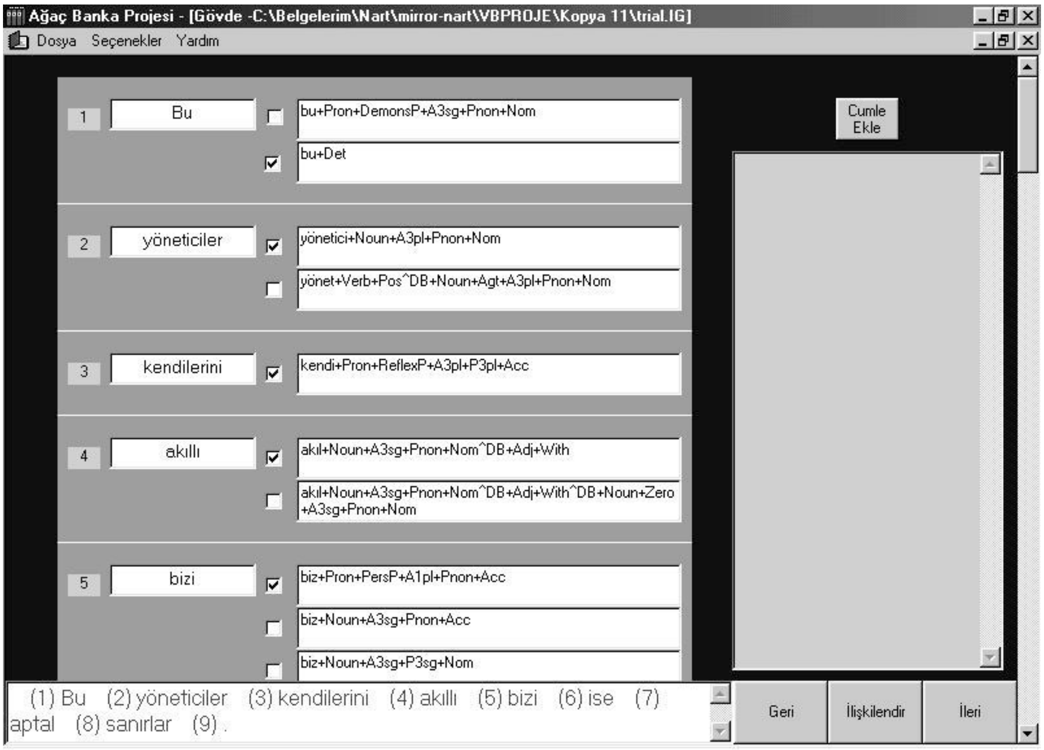
Kemal Oflazer, Bilge Say, Dilek Zeynep Hakkani-Tür, and Gökhan Tür. 2003. Building a turkish treebank. In Anne Abeillé, editor, *Building and Exploiting Syntactically-annotated Corpora*. Kluwer Academic Publishers.

Kemal Oflazer. 1993. Two-level description of Turkish morphology. In *Proceedings of the Sixth Conference of the European Chapter of the Association for Computational Linguistics*, April. A full version appears in *Literary and Linguistic Computing*, Vol.9 No.2, 1994.

Bilge Say and Kemal Oflazer, 2002. *Annotation Manual for Turkish Treebank*. June 2002.

Bilge Say, Deniz Zeyrek, Kemal Oflazer, and Umut Özge. 2002. Development of a corpus and a treebank for present-day written turkish. *11th International Conference on Turkish Linguistics*.

C. M. Sperberg-McQueen and Lou Burnard, 1994. *Guidelines for Electronic Text Encoding and Interchange (TEI P3)*. Chicago, Oxford: Text Encoding Initiative.


Ağaç Banka Projesi - [Gövde -C:\Belgelerim\Nart\mirror-nart\VBPROJE\Kopya 11\trial.IG]

Dosya Seçenekler Yardım

1 Bu
- □ bu+Pron+DemonsP+A3sg+Pnon+Nom
- ☑ bu+Det

2 yöneticiler
- ☑ yönetici+Noun+A3pl+Pnon+Nom
- □ yönet+Verb+Pos^DB+Noun+Agt+A3pl+Pnon+Nom

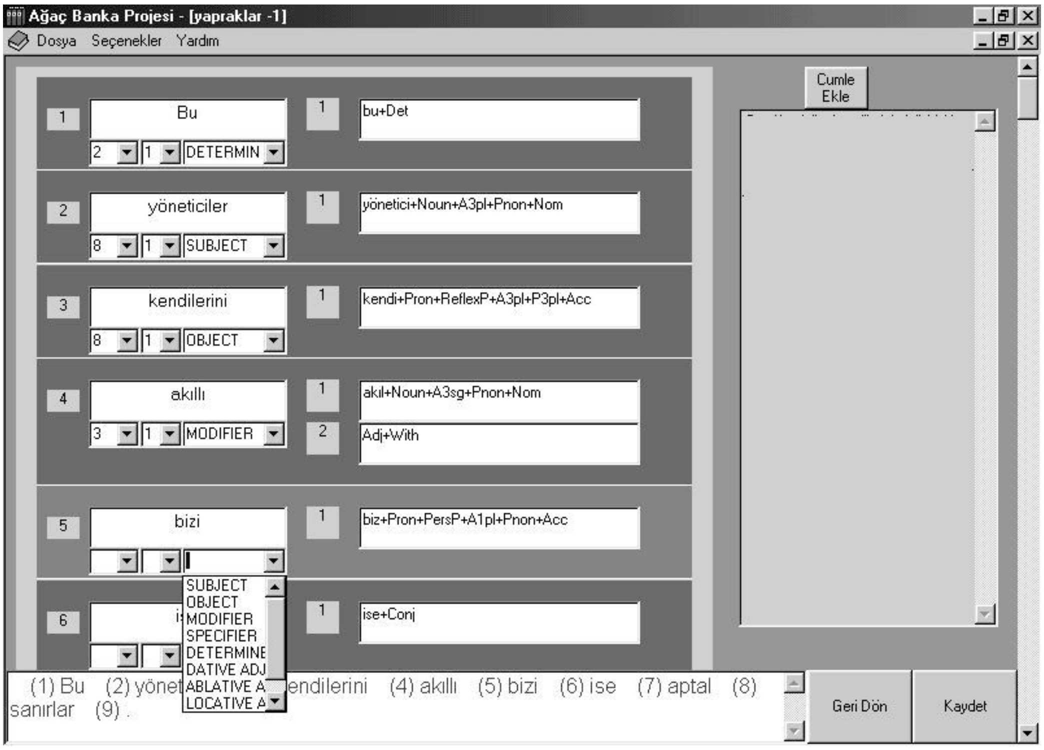
3 kendilerini
- ☑ kendi+Pron+ReflexP+A3pl+P3pl+Acc

4 akıllı
- ☑ akıl+Noun+A3sg+Pnon+Nom^DB+Adj+With
- □ akıl+Noun+A3sg+Pnon+Nom^DB+Adj+With^DB+Noun+Zero+A3sg+Pnon+Nom

5 bizi
- ☑ biz+Pron+PersP+A1pl+Pnon+Acc
- □ biz+Noun+A3sg+Pnon+Acc
- □ biz+Noun+A3sg+P3sg+Nom

(1) Bu (2) yöneticiler (3) kendilerini (4) akıllı (5) bizi (6) ise (7) aptal (8) sanırlar (9) .

Figure 2: Disambiguation Screen of Annotation Tool


Ağaç Banka Projesi - [yapraklar -1]

Dosya Seçenekler Yardım

1	Bu	2 ▾ 1 ▾ DETERMIN ▾	1	bu+Det
2	yöneticiler	8 ▾ 1 ▾ SUBJECT ▾	1	yönetici+Noun+A3pl+Pnon+Nom
3	kendilerini	8 ▾ 1 ▾ OBJECT ▾	1	kendi+Pron+ReflexP+A3pl+P3pl+Acc
4	akıllı	3 ▾ 1 ▾ MODIFIER ▾	1 2	akıl+Noun+A3sg+Pnon+Nom Adj+With
5	bizi	▾ ▾ ▾	1	biz+Pron+PersP+A1pl+Pnon+Acc
6	i	▾ ▾	1	ise+Conj

(1) Bu (2) yönet endilerini (4) akıllı (5) bizi (6) ise (7) aptal (8) sanırlar (9) .

Figure 3: Dependency Tagging Screen of Annotation Tool