Hebrew Treebank 2.0

Background

The Hebrew Treebank Version 2.0 contains 6500 sentences of news items from the Ha'aretz daily newspaper, with full word segmentation and morpho-syntactic analysis. Morphological features that are not directly relevant for syntactic structures, like roots, templates and patterns, are not analyzed.

A version of the Treebank with only the morphological level is also supplied.

Version 2.0 of the Hebrew Treebank contains three enhancements and improvements as compared to previous versions:

- Corrected and streamlined annotations of previously analyzed material.
- An annotation of 1500 previously unanalyzed sentences.
- Most notably - an annotation of father-child dependencies (see below).

Tag set: The Hebrew Treebank was designed with a tag set that is as close as possible to that of the English Penn Treebank. A significant difference from English is that in the Hebrew Treebank the annotated words are separated into morphemes, which may have different POS tags and syntactic positions in the tree.

For example, the two words BBIT HGDWL ("in the big house") are analyzed as five different morphemes: B H BIT H GDWL in the house the big Note that the first occurrence of the morpheme H ("the") is covert in the word BBIT. Segmentation into morphemes makes it possible to analyze different morphemes of the same word as belonging to different constituents in the tree: [B [[H BIT] [H GDWL]]

Another significant difference from English is the relatively free order of constituents in Hebrew.
In order to encode syntactic functions of constituents within a tree structure, functional features (for subject, object etc.) were added to constituents, and the constituents themselves appear in a "flat" order.
Some tags were added to the tag set of the Penn Treebank, to accommodate special properties of Hebrew like construct states, accusative marking (AT), the definiteness morpheme H, and verbless predicates.

***NEW* Father-child dependencies** Version 2.0 of the Hebrew Treebank uses special annotation features to mark all cases where the morpho-syntactic features of a node are inherited from one or more of its children. We refer to such cases as "father-child dependencies". In a structure where X is a node and Y is one of X's children, a dependency between X and Y is marked by adding to Y the feature DEP_, where Z is the classification of the dependency.

We distinguish six classes of dependency features:

- **DEP_HEAD: Y** is the only child of X that determines X's features. For example: this is the relation between a simple noun (e.g. BIT) in a non-construct state, and an indefinite noun phrase (e.g. BIT GDWL). Note that head-dependencies are not necessarily marked when no features are inherited.
- **DEP_DEFINITE: Y** only determines X's definiteness feature. For example: this is the relation between the definite article H on a simple noun and a non-construct state NP containing it (e.g. H-BIT H-GDWL). Another important example of definiteness dependency is the relation between a post-construct nominal (somex) and the whole construct state NP (see below).
- **DEP_NUMBER: Y** only determines X's number feature. For example: this is the relation between the numeral (e.g. AXD) and the NP containing it in a partitive construction like AXD H-BTIM ("one of the houses"). This describes the number dependency in this case of the (singular) NP on the numeral, rather than the noun.
- **DEP_MAJOR: Y** determines most of X's features, but there is one or more brothers of Y that determine definiteness and/or number features of X. For example: this is the relation between a construct-state noun and a full NP, since (only) definiteness in Hebrew construct states is determined by the post- construct nominal (somex). These conventions lead to the following annotation of the construct BIT H-ILDH ("the girl's house"):

  ```
  (NP-ZR-H
  (NNT-ZY-DEP_MAJOR BIT )
  (NP-NY-H-DEP_DEFINITE
  (H-DEP_DEFINITE H )
  (NN-NY ILDH )))
  ```

  And similarly, in a partitive NP:

  ```
  (NP-ZY
  (CDT-ZY-DEP_NUMBER AXD )
  (NP-ZR-H-DEP_MAJOR
  (H-DEP_DEFINITE H )
  (NN-NY BTIM )))
  ```

- **DEP_ACCUSATIVE: Y** determines X's OBJ (object feature). This is the relation between the accusative marker AT and the containing NP.
- **DEP_HEAD_MULTIPLE: Y** is one of a few other children of X that determine its features in a parallel structure. This is the case in most X's that dominate coordinate structures, as well as some special prepositional phrases.

Further documentation

For a detailed description of the linguistic annotation scheme of the previous version (without the dependency annotation), see Sima'an et al (2001):
`http://mila.cs.technion.ac.il/treebank/tal.pdf`

For the full conventions used in Version 2.0, with sample examples, see the annotator guidelines.

Software

The following software was used for the development of the treebank:

SEMTAGS (Remko Bonnema, University of Amsterdam)
A GUI for tree annotation.
Morphological analyzer (Erel Segal, Technion)
Provides an initial tagging for the input sentences, which was manually fixed by the annotators.
Format translation script (Alon Altman, Technion)
Translates the output of the morhpological analyzer to the Treebankλ~@~Ys format, and creates initial flat trees for annotation.
Mapping trees to the original text (Avihai Dgani, Technion)
This program maps each word in the raw text to the corresponding morphemes in the treebank, and checks the synchronization between the parse trees and the original text.

Comments

Null trees: the original text was automatically segmented into sentences. In some cases, a single sentence in the original text was splitted into multiple sentences. Therefore, in the annotation process it was necessary to rejoin sentences. In order to maintain the synchronization between the tree numbers and the automatically segmented sentence numbers, null trees were inserted. For example, if the first sentences was splitted into three sentences, #1, #2 and #3, then in the treebank, tree #1 will include the three sentence parts, while tree #2 and tree #3 will be null trees.

A null tree looks like this:

```
S
|
yyDOT
|
yyDOT
```

and is represented as: ((S (yyDOT yyDOT)))

Duplicate sentences: sentences 24-36 in the original text do not appear in the treebank, since they are repeated in sentences 1358-1370. Sentences 1249-1293 do not appear in the treebank, since they are repeated in sentences 1204-1248.

Missing trees: the following sentences currently do not appear in the treebank (represented by null trees): 552, 772, 1350, 1382, 2044, 3206

Transliteration

א	ב	ג	ד	ה	ו	ז	ח	ט	י	ך	כ	ל	ם	מ	ן	נ	ס	ע	ף	פ	ץ	צ	ק	ר	ש	ת
a	b	c	d	h	w	z	x	v	i	k	k	l	m	m	n	n	s	y	p	p	c	c	q	r	e	t

Staff

Research supervision

Prof. Alon Itai, Prof. Yoad Winter, Dr. Khalil Sima'an

Development of the annotation scheme, and tree annotation:

Dr. Yuval Krymolowski, Yair Adiel, Nomi Guthmann, Shiri Kenan, Adi Milea, Noa Nativ, Roni Tenzman, Pnina Veisberg.

Technical Support and Design

Dr. Yuval Krymolowski, Alon Altman, Roy Bar-Haim.