Global
WordNet
Association

A CCGbank for Turkish: From Dependency to CCG

Aslı Kuzgun	Oğuz Kerem Yıldız	Olcay Taner Yıldız
Boğaziçi University	Starlang Yazılım Danışmanlık	Özyeğin University
Starlang Yazılım Danışmanlık		

Global WordNet Conference 2023

Donostia - January 25, 2023

Introduction

- The aim of this study is to create a tool for semantic parsing (to be used in automated inquiry systems, chat-box tools, search engines).
- Dependency or tree structures do not provide one to one correspondence between syntax and semantics.
    - Information on argument structures of verbs and semantic types of lexical items is missing.
- CCG offers a categorical lexicon and a more transparent structure between syntax and semantics.
- CCGbanks have higher parsing scores than their treebank equivalents (Hockenmaier and Steedman, 2007; Bosco et al., 2000; Çakıcı, 2009; Ambati et al., 2018).
- CCG approach requires a bigger corpus for the machines to learn each lexical type.
- In this study we automatically transferred an already existing Turkish dependency corpora to a CCGbank.

Previous Studies in CCG

- The dependency to CCG conversion studies started in 2006 by Hockenmaier for the German language.
- Other conversion studies are as follows:
    - English (Hockenmaier and Steedman, 2007),
    - Chinese (Tse and Curran,2010),
    - Italian (Johan et al., 2009),
    - Hindi (Bhatt et al., 2009).
    - Turkish, Çakıcı (2009)
        - Çakıcı (2009) aimed a morphemic CCGbank lexicon for the first time. That is, she assigns categories to the morphological units as well as the lexical units.
        - At the time, there was only one dependency corpus present in Turkish and it was not big enough for a CCGbank. (METU Turkish Corpus (Atalay et al., 2003; Oflazer et al., 2003) contained 60k word tokens)

CCG: Definition

- **Combinatory Categorial Grammar (CCG) (Steedman, 2000):** a lexical grammar formalism that offers a transparent interface between syntax and semantics.

CCG: Lexicon

- **Combinatory Categorial Grammar (CCG) (Steedman, 2000):** a lexical grammar formalism that offers a transparent interface between syntax and semantics.
  - **The Lexicon:**

yemek-ler
food-PL
NP

daha
more
(NP/NP)/(NP/NP)

ye-di
eat-PAST
S

sağlıklı
healthy
NP/NP

ye-di
eat-PAST
S\NP

ye-di
eat-PAST
S\NP\NP$_{[nom]}$

CCG: Combining Lexical Items

● Forward Application : X/Y➡ X applied to Y becomes X	atomic categories
● Backward Application : X\Y ➡ X applied to Y becomes X	
● Forward Composition : (X/Y) combined with (Y/Z) becomes X/Z	complex categories
● Backward Composition : (Y\Z) combined with (X\Y) becomes X\Z	
● Forward Type-raising : X becomes T/(T\X)	type mismatch
● Backward Type-raising : X becomes T\(T/X)	

The Input: Dependency Treebanks

The input we applied the CCG algorithm consists of the following dependency treebanks:

- The Turkish Penn Treebank
  - Consists of a total of 87,367 word tokens which are translated from the original Penn Treebank corpus.
  - The data consists of translated sentences taken from journals such as Wall Street Journal Articles.
- FrameNet
  - Consists of 19,221 tokens and 140 different semantic frames
- KeNet
  - The largest treebank of Turkish with 178,70 tokens
  - Sentences consist of example sentences taken from the Turkish National dictionary.
- Atis
  - A domain specific treebank that is built from the audio recordings of people inquiring for flight information from automated systems (translated from English)
  - Consists of 45,875 tokens
- Tourism
  - A domain specific treebank that includes written customer reviews for a booking company
  - Consists of 92,200 tokens

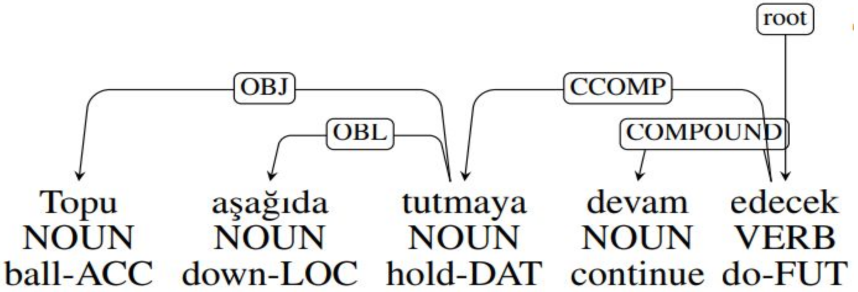
The Input: Framework

The treebanks we used were annotated under the framework provided by **the Universal Dependencies (UD).**

- The universal dependencies aim to achieve a cross-linguistically consistent treebank annotation.
- The Universal Dependency Project pioneered to develop treebanks for languages other than English since 2013.
- There are currently 200 treebanks over 100 languages released in the project.

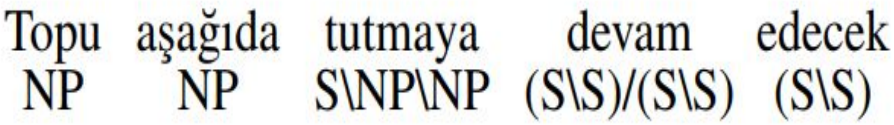
“From Dependency to CCG” in a nutshell

The dependency structure



"S/he will continue to hold the ball down"

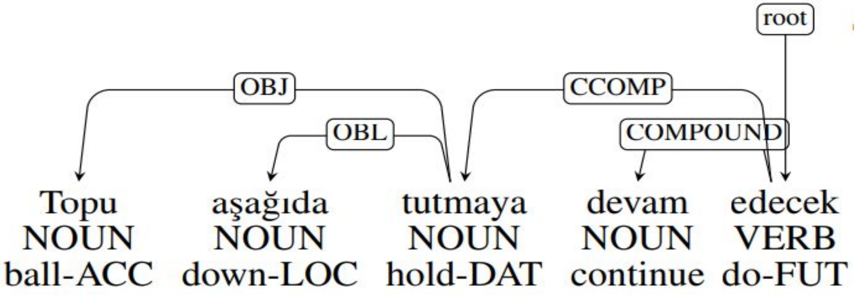
The CCG structure



Topu	aşağıda	tutmaya	devam	edecek
NP	NP	S\NP\NP	(S\S)/(S\S)	(S\S)

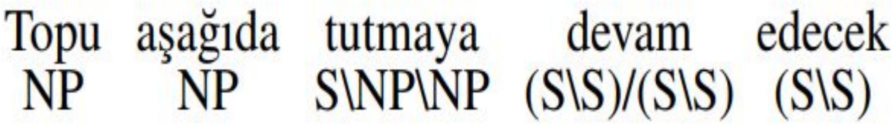
“From Dependency to CCG” in a nutshell

The dependency structure



"S/he will continue to hold the ball down"

The CCG structure



Topu	aşağıda	tutmaya	devam	edecek
NP	NP	S\NP\NP	(S\S)/(S\S)	(S\S)

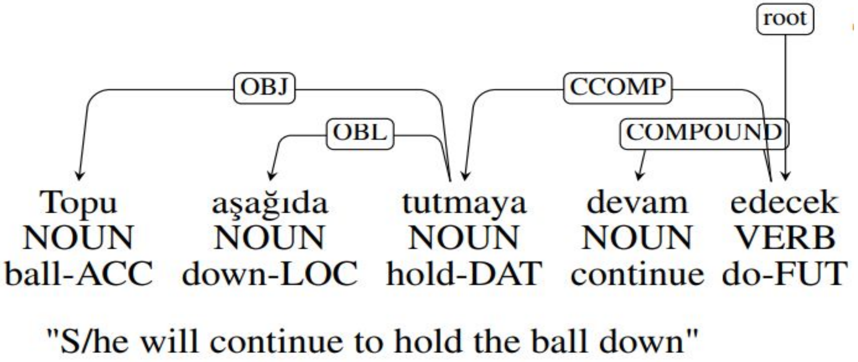
The CCG algorithm is based on:

- POS information of the word tokens,
- Head/complement relationship between the tokens,
- The dependency label between the tokens.

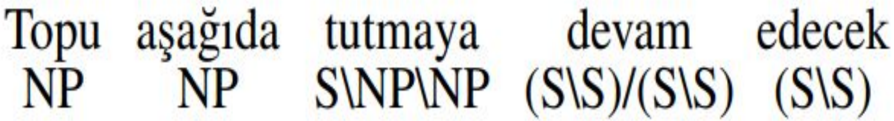
The CCG Algorithm: Identifying Arguments

- First step of the algorithm is to identify the arguments of the matrix predicate.
- Arguments can be nominal or clausal.
- Nominal arguments come from the relations such as OBJ or OBL.
    - The subject NP’s are marked as NP$_{[nom]}$.

- Clausal arguments such as CCOMP and XCOMP are added to the lexical item as S.



"S/he will continue to hold the ball down"

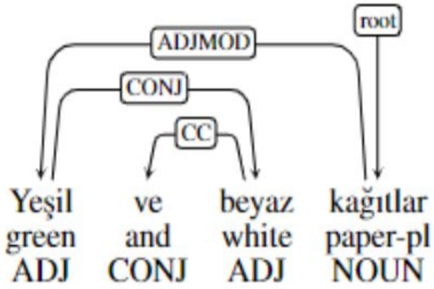


Topu	aşağıda	tutmaya	devam	edecek
NP	NP	S\NP\NP	(S\S)/(S\S)	(S\S)

The CCG Algorithm: Conjuncts

- Conjuncts are given the category of (X\X)/X in the first iterations
- Then the variables take the category of the conjuncts (e.g. X = NP/NP)

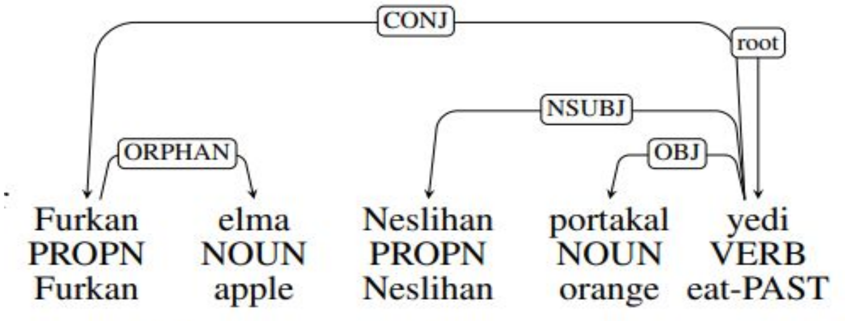
The dependency annotation:



The CCG annotation:

Yeşil	ve	beyaz	kağıtlar
Green	and	white	papers
(NP/NP)/(NP/NP)	(X\X)/X	NP/NP	NP

Ellipsis



"Neslihan ate oranges, Furkan apples."

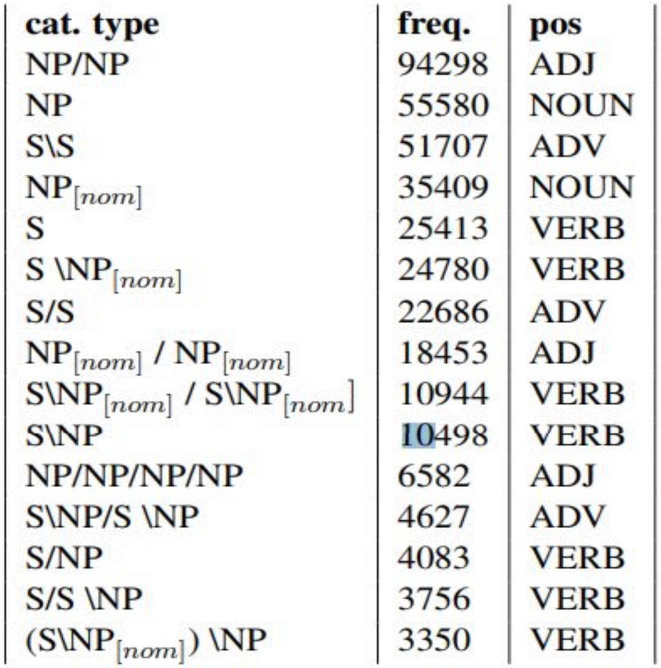
Figure1: ellipsis of the predicate

- The remaining argument is linked to the head of the clause with ORPHAN relation.
- ORPHAN relation signifies that “apple” is not an argument of Furkan.
- The CONJ relation adds an argument to the matrix predicate.

CCG translation:

Furkan	elma	Neslihan	portakal	yedi.
Furkan	apple	Neslihan	orange	eat-PAST
NP	NP	$NP_{[nom]}$	NP	$S\backslash NP_{[nom]}\backslash NP\backslash NP$

Results



cat. type	freq.	pos
NP/NP	94298	ADJ
NP	55580	NOUN
S\S	51707	ADV
$NP_{[nom]}$	35409	NOUN
S	25413	VERB
$S \backslash NP_{[nom]}$	24780	VERB
S/S	22686	ADV
$NP_{[nom]} / NP_{[nom]}$	18453	ADJ
$S\backslash NP_{[nom]} / S\backslash NP_{[nom]}$	10944	VERB
S\NP	10498	VERB
NP/NP/NP/NP	6582	ADJ
S\NP/S \NP	4627	ADV
S/NP	4083	VERB
S/S \NP	3756	VERB
$(S\backslash NP_{[nom]}) \backslash NP$	3350	VERB

Table: The most frequent 15 categories

- There are 630 different categories in this CCGbank with 516k words.
  - This number was around 530 in the previous CCG study in Turkish even with a corpus consist of 60k words.

- Simple/ atomic categories are more common.

Conclusion

- In this study, we presented the process of inducing a CCGbank for Turkish from an existing dependency treebank.
- Introduced an algorithm that can be applied to all dependency treebanks in Turkish with UD annotations.
- UD annotations are updated regularly, therefore, the algorithm might need updates for the upcoming treebanks and UD releases.
- The annotation frameworks become more and more morphemic in each release, so that we expect the algorithm to become less lexicalist in the future.

References

Bharat Ram Ambati, Tejaswini Deoskar, and Mark Steedman. 2018. Hindi ccgbank: A ccg treebank from the hindi dependency treebank. Language Resources and Evaluation, 52(1):67–100.

Cristina Bosco, Vincenzo Lombardo, Leonardo Lesmo, and Vassallo Daniela. 2000. Building a treebank for italian: a data-driven annotation schema. In LREC 2000, pages 99–105. ELDA.

Julia Hockenmaier. 2006. Creating a ccgbank and a wide-coverage ccg lexicon for german. In Proceedings of the 21st international conference on computational linguistics and 44th annual meeting of the association for computational linguistics, pages 505– 512.

Julia Hockenmaier and Mark Steedman. 2007. Ccgbank: a corpus of ccg derivations and dependency structures extracted from the penn treebank. Computational Linguistics, 33(3):355–396.

Ruket Çakıcı. 2005. Automatic induction of a ccg grammar for turkish. In Proceedings of the ACL student research workshop, pages 73–78.

Ruket Çakıcı. 2009. Wide-coverage parsing for turkish.