A CCGbank for Turkish: From Dependency to CCG

Aslı Kuzgun
Boğaziçi University
Starlang Yazılım Danışmanlık
asli.kuzgun@boun.edu.tr

Oğuz Kerem Yıldız
Starlang Yazılım Danışmanlık
oguz@starlangyazilim.com

Olcay Taner Yıldız
Ozyegin University
olcay.yildiz@ozyegin.edu.tr

Abstract

In this paper, we present the building of a CCGbank for Turkish by using standardised dependency corpora. We automatically induce Combinatory Categorial Grammar (CCG) categories for each word token in the Turkish dependency corpora. The CCG induction algorithm we present here is based on the dependency relations that are defined in the latest release of the Universal Dependencies (UD) framework. We aim for an algorithm that can easily be used in all the Turkish treebanks that are annotated in this framework. Therefore, we employ a lexicalist approach in order to make full use of the dependency relations while creating a semantically transparent corpus. We present the treebanks we employed in this study as well as their annotation framework. We introduce the structure of the algorithm we used along with the specific issues that are different from previous studies. Lastly, we show how the results change with this lexical approach in CCGbank for Turkish compared to the previous CCGbank studies in Turkish.

1 Introduction

Semantic parsing is a vital tool for natural language processing (NLP) studies. Automated inquiry systems, chat-box tools, search engines, and all sorts of other NLP applications make use of semantic information. The semantic information, however, is not encoded in the dependency or phrase treebanks directly. The syntactic relations in these frameworks do not follow from the semantic types of tokens such as the argument structure of predicates. Therefore, there is a trend in converting these dependency annotated treebanks into a semantically transparent framework, which is CCG. CCG creates a categorical lexicon where each token is assigned a lexical category according to how it combines with the other tokens in a given sentence. This approach increases parsing scores compared to dependency parsing studies (Hockenmaier and Steedman, 2007; Bosco et al., 2000; Çakıcı, 2009; Ambati et al., 2018). However, the CCG approach requires a bigger corpus for the machines to learn each lexical type.

There are languages that have adequate dependency corpora such as English, Hindi, and Italian. Consequently, there are CCG induction studies over the dependency corpora of these languages (Hockenmaier and Steedman, 2007; Ambati et al., 2018; Bosco et al., 2000). Turkish, on the other hand, did not have a big dependency annotated corpus. There was only the METU-Sabancı Treebank (Oflazer et al., 2003; Atalay et al., 2003). The first CCGbank conversion studies in Turkish was conducted with a smaller corpus and therefore some rare categories were not repeated enough. Today, bigger dependency corpora are available in Turkish under the UD Framework. These corpora are not only valuable because they provide a bigger parsing tool, but also they are annotated according to a universal annotation scheme that can be used for parallel annotation studies. Today all the Turkish dependency corpora are standardized in order to be a part of the UD framework. In this study, we aimed to employ the UD annotation framework to induce a CCGbank for Turkish that can be used in all the Turkish annotated corpora in the UD that consists of 671K tokens. In contrast to the previous CCG studies in Turkish, we used a lexical approach in CCG instead of a morphemic approach. This is because the syntactic relations are defined based on lexemes and not morphemes in the UD framework for Turkish and employing this framework has several advantages explained above. However, the UD standards keep improving for all languages and it might become morphemic in the later releases. Then, the algorithm we provided here can be adapted to those changes in the UD framework and turn into a morphemic approach.

This paper consists of 6 sections. First section introduces the study and our motivations for this

study and it continues with an introduction to the CCG. The second section provides information about the CCGbank studies in different approaches and languages. In Section 3, we introduce the dependency treebanks we used in this study. After this, we explain the algorithm we used to convert this treebank into a CCGbank in Section 4. The last two sections are devoted to present the statistics from the resulting CCGbank corpus and to conclude our study.

1.1 Combinatory Categorial Grammar

Combinatory Categorial Grammar is a lexical grammar formalism that offers a transparent interface between syntax and semantics. CCG approaches define all kinds of language properties in the lexicon. The lexicon consists of the syntactic categories of words and CCG combines these categorical tokens together to derive sentences. This kind of derivation follows from the same logic behind the type-driven semantics where words are associated with functions and the sentences are built by the application of these functions to each other.

The lexicon is built by considering the syntactic categories of words. For instance, an intransitive verb is labelled as category S \NP and a transitive verb is labelled as (S \NP) \NP in Turkish. The S corresponds to the root which is what is left from the sentence at the end of the derivation. The amount of NP's signifies the amount of arguments that a verb can take. The intransitive verb has only one NP because it has no object argument, the only NP this verb interacts with is the subject NP. A transitive verb has an object relation with an NP by definition, therefore, they are assigned an extra NP to their syntactic category.

Akkuş (2014) defines two types of CCG categories, namely, atomic and complex. The atomic categories consist of single units of parts of speech tokens such as NP, PP, S and so on. The complex categories, on the other hand, consist of the combination of other categories. For instance, S is an atomic category and it is the category of an intransitive verb that does not take any overt subject argument, which is a common instance in pro-drop languages like Turkish. S \NP, however, is a complex category which combines two atomic categories. (S \NP) \NP is also a complex category where the complex category S \NP is combined with the atomic category NP. The direction of the slashes in the complex categories label the direction of the argument in which that token will enter into a relationship. The S \NP tag provides the information that the verb has its subject on its left. Similarly, the category (S \NP) \NP shows that both the object and the subject are located on the left of the verb. Such a system also predicts that the object will be on the right of the subject. This is possible as a result of the derivation system of the CCG. For instance, an example of a category (S \NP) \NP verb is "to read". The semantic category of this verb in function terms would be as shown in (1). Here, the object x has to be defined before the subject y. Similarly, in (S \NP) \NP, the object is the NP on the right edge which will be closer to the verb and will be applied before the subject NP. Once the object NP enters into a relationship with the verb (S \NP) \NP, it drops the NP on the right edge and the verb becomes S \NP. This shows that there is only one argument left in the derivation for this verb to enter into a relation.

(1) λx. λy. y reads x

In addition to the composition operations defined above, there are also type raising operations. Type raising occurs when there is a case of ellipsis, movement, or a similar syntactic operation that causes a type mismatch between the two tokens in the derivation. Since CCG is completely transparent between syntax and semantics, these kinds of syntactic phenomena are covered by the type raising rules where the category of a word token is "raised" in order to continue the derivation.

The compositional and type raising rules used in the CCG can be formulated as follows:
Forward Application : X/Y applied to Y becomes X
Backward Application : Y applied to X\Y becomes X
Forward Composition : X/Y applied to Y/Z becomes X/Z
Backward Composition : Y\Z applied to X\Y becomes X\Z
Forward Type-raising : X becomes T/(T\X)
Backward Type-raising : X becomes T\(T/X)

2 Related Work

The first CCGbank was introduced by Hockenmaier and Steedman (2007) for English. This CCGbank was converted automatically from the first phrase structure corpus of English, the Penn Treebank (Marcus et al., 1993). In addition, Hocken-

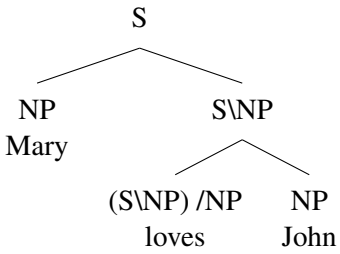


Figure 1: CCG labeled binary tree structure

maier (2006) converted the Tiger treebank (Brants et al., 2004) in German to a CCGbank. These conversion studies were held when the phrase treebanks were only recently being converted into dependency treebanks (De Marneffe et al., 2006). Therefore, several CCGbank studies in languages with already existing treebanks were also converted from phrase treebanks such as the Chinese Treebank developed by Tse and Curran (2010).

This type of conversion studies consist of four main stages. First, they preprocess the existing phrase treebank and correct any errors that could cause combination errors. Then, they determine the constituent types by considering the part-of-speech (POS) tag information and the mother node of each token. For instance, if an NP node branches from a VP node, then that NP is considered as an object constituent. Likewise, PP constituents in the phrase structure are considered as adjunct constituents. After this, they binarize the phrase structure. Binarization is a necessary step in CCG conversion since it is crucial to determine which word token is in the domain of the other to derive the correct compositions. Then, they assign CCG categories to each lexical token in the binarized structure according to the type of relationship between the two tokens. If there is a complement relationship between the two tokens, then the lexical category of the complement is added to the head token with a slash pointing its location to the head word. Figure 1 illustrates an example to this final structure.

Languages that did not already have a phrase treebank started to build dependency treebanks to begin with. Therefore, in the following years CCGbank studies started to be converted from the dependency treebanks. Johan et al. (2009) converted The Turin University Treebank (TUT) (Bosco et al., 2000) in Italian, Çakıcı (2009) converted the METU-Sabancı Treebank (Oflazer et al., 2003; Atalay et al., 2003) in Turkish, Ambati et al. (2018) created Hindi CCGbank from the Hindi Dependency Treebank (Bhatt et al., 2009).

Unlike the previous studies, Çakıcı (2009) and Çakıcı (2005) offer a morphemic CCGbank lexicon. That is, she assigns categories to the morphological units as well as the lexical units. She argues that lexical category assignment cannot cover all the syntactic phenomena in agglutinative languages like Turkish. Following the work of Çakıcı (2005) in Turkish, Ambati et al. (2018) also employed a morphemic lexicon in Hindi, which is another agglutinative language.

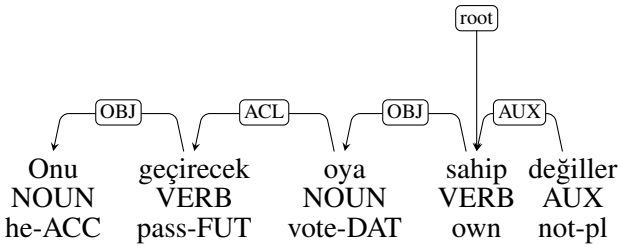
It should be noted that CCG approaches with morphemic lexicon do not assign CCG to each morpheme in a word but rather the derivational morphemes such as relativizers that derive adjectives from verbs. Çakıcı (2009) shows that assigning CCG categories to these morphemes decreases the amount of categories that each verb has and thus provides better parsing results by increasing the average frequency of the word roots. However, lexical rules can also account for the derivational changes with the combination of the case, POS tag, and dependency relation information.

3 The Input Corpora

The dependency treebanks that we induced a CCGbank corpus are Turkish version of The Penn Treebank (Marcus et al., 1993), FrameNet, KeNet, Atis, and Tourism. These treebanks employ the annotation framework provided by the Universal Dependencies (UD). All of our input corpora are manually annotated according to the UD annotation framework (de Marneffe et al., 2021) and they are in the CoNLL format. All of them are available online at UD[1], and free of license.

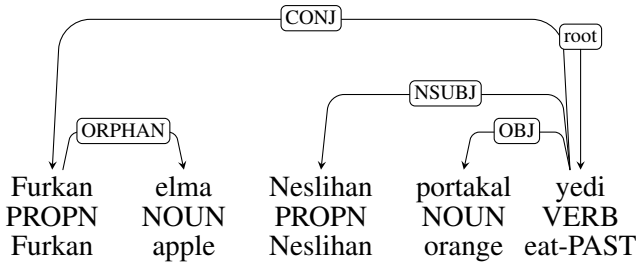
The dependency annotation employed in the treebanks we used is illustrated in Figure 2. The relations build constituents. The morpho-syntactic layer of the treebank consists of POS tag, and morphological information. The morphological features change according to the word category. For instance, definiteness is only defined for nouns, tense/aspect/modality are only defined for verbs, degree information is only defined for adjectives, and so on. As illustrated in figure 2, the relations used in this treebank differ from the previous treebanks used in the earlier works of the CCGbank studies in Turkish. Çakıcı (2005) employed The

[1] https://universaldependencies.org



"They don't have the votes to pass it."

Figure 2: Surface dependency structure



"Neslihan ate oranges, Furkan apples."

Figure 3: Ellipsis in The Turkish Penn Treebank

METU-Sabancı Treebank corpus (Atalay et al., 2003; Oflazer et al., 2003). For instance, in figure 2, the adjectival modifier with the verbal root is labelled as ACL, and this signifies that it is a clausal adjective, otherwise, an adjectival modifier of an NP would be labeled as an AMOD. The morphemes are encoded in the morpho-syntactic layer, however, they are not labelled as a separate token in the surface dependency structure[2].

The morphological analysis of these corpora is processed by a semi-automatic morphological analyzer Yıldız et al. (2019). This semi-automatic approach increased the accuracy of the analysis. The analyzer provides the possible analyses of a word token to the annotator and the annotator selects the correct one considering the context of the sentence. This way, a consistent and contextually accurate morphological analysis was achieved. Semantic and dependency annotation is performed manually.

3.1 The Universal Dependencies

The UD framework provides an inclusive annotation scheme that enables parallel annotations between languages. There are more than 100 languages represented in this framework. Turkish has 8 up-to-date dependency annotated corpora represented in the UD. The METU-Sabancı Treebank used in the previous CCGbank studies unfortunately cannot be updated to meet the latest UD standards. However, the 8 other treebanks provide corpora in a variety of genres such as reviews, articles, automated inquiry system inputs, and so on. All of these manually annotated dependency corpora make the UD databank of Turkish an invaluable source for NLP studies.

The UD annotation framework offers labels to account for the ellipsis cases. There is a combination of two relations, namely, ORPHAN and CONJ to account for ellipsis phenomenon. These two relations overcome the problem of the lack of traces in the dependency treebanks. Figure 3 illustrates how ellipsis is encoded in the treebank. Since there cannot be two subjects in a sentence, one of them is labelled with the CONJ relation to the root, and the object of that subject/verb pair is linked to its subject with the ORPHAN relation.

3.2 The Penn Treebank

The Turkish Penn Treebank consists of a total of 9560 sentences and 87,367 word tokens which are translated from the original Penn Treebank corpus. The corpus only includes sentences that are less than 15 words long. The sentences are from the written texts such as Wall Street Journal articles, exchange rate information and also some advertisement dialogues. This corpus was first annotated according to an earlier version of the UD (Kuzgun et al., 2020), however, it is updated to fulfill the latest UD annotation standards (de Marneffe et al., 2021).

3.3 The FrameNet

Turkish FrameNet is a manually annotated dependency corpus that is built from the sentences taken from the Turkish FrameNet Project. It consists of 2,700 manually annotated example sentences and 19,221 tokens. The treebank can be separated according to the semantic frames. For instance, "cognitive comprehension" is a frame, and the sentences that include a verb that means anything related to this semantic concept can be filtered. There are 139 semantic frames that the treebank can be filtered into.

The dependency annotation of thic corpus is fully manual and also it can be combined with the framenet annotation. Which means the tokens of this corpus are annotated with thematic roles. For example in the frame "cognitive comprehension"

[2]ACC=accusative, DAT=dative, AUX=auxiliary

the subject is not only marked as the subject, but it also carries the information that it is the "thinker".

3.4 The KeNet Treebank

The KeNet is the largest treebank of Turkish. The sentences are not domain specific, they are mostly the dictionary example sentences of the Turkish National Dictionary. There are 18,700 manually annotated sentences and 178,700 tokens in this corpus.

3.5 The Atis Treebank

The Atis Treebank in Turkish consists of the translated sentences of the original Atis Dataset in English (Ward, 1990). This is a domain specific dataset which is built from the audio recordings of people inquiring for flight information from automated systems. The sentences were first translated by an automated translator. Then, human translators fine grained the sentences before the annotation to create the final version of the Turkish Atis corpus. The dependency annotation is made by human annotators as the other treebanks used in this study. The annotated Atis corpus in Turkish contains 5432 sentences and 45875 tokens.

3.6 The Tourism Treebank

The Tourism Treebank consists of a domain specific corpus of Turkish. There are 19,750 manually annotated sentences and 92,200 tokens in this treebank. The sentences are taken from the customer reviews of a booking company. The reviews were written unlike the Atis data. They were not subjected to a transcription process. Therefore they contain orthographic mistakes. In order not to distort the features of a natural speech data, we used the "GOESWITH" tag of the UD where we combined the tokens that were supposed to be together. When the tokens were mistakenly written together, then we separated them manually.

4 The CCG Algorithm

The CCG label assignment is carried out by an algorithm that makes use of the POS information of the word tokens, the head/complement relationship between the tokens, and the dependency label between the tokens. The algorithm starts from the left edge of the sentence, sees the POS tag of the first token, and then finds where that token is connected. Together with these, the relationship between the two tokens defines the CCG label of the token.

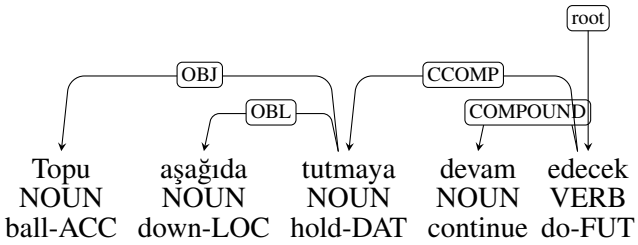
```
CASE: "NMOD"
    IF headcat="NP[nom]"
        WHEN myrel="NMOD"
            SET NP [nom]/ NP [nom]
    ELSE
        IF headcat="NP"
            WHEN myrel= "NMOD"
                SET NP/ NP
```

Table 1: Algorithm for nominal modifiers

Once one token is defined, the algorithm continues to the next token. However, CCG assignment algorithm does not end in one iteration because of the complex CCG categories that contain X. If the token being processed, or the head word of a constituent has an X in its rule, then it means that token is not identified yet. Therefore, these tokens are left for the following iterations. This process repeats itself until all the categories are defined.

Table 1 illustrates an instance from the algorithm. According to this rule, an NP token that is connected to another NP with the NMOD relation will take the NP/NP CCG label. If the modified NP is the subject of the sentence, then it will be labeled as $NP_{[nom]}$, therefore, the modifier token will not be NP/NP but $NP_{[nom]}/NP_{[nom]}$ for the subjects.

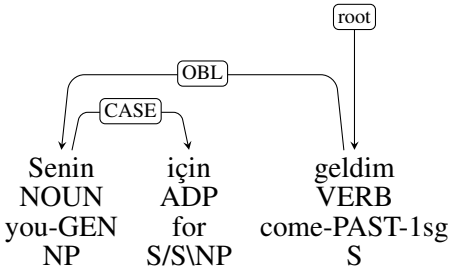
The algorithm is not morphemic as the previous Turkish CCGbank studies (Çakıcı, 2009; Akkuş, 2014). One reason for that is the dependency annotation structure of the treebank we employed. The UD Turkish framework offers a detailed and consistent annotation scheme, however, it does not separate the morphemes as Çakıcı (2009) did in her dissertation. Therefore, the input corpus does not include separate tokens. However, as the annotation scheme that was used in The Turkish Penn Treebank accounts for cases like ellipsis which other dependency annotation frameworks fail to cover, we employed the lexical approach as the previous studies on The Penn Treebank induction did (Hockenmaier and Steedman, 2007). Even though they were phrase treebanks, the dependency annotation scheme we employed offers similar kind of information. Our motivation in applying lexical approach is to make use of the universal and standard annotation scheme in our algorithm.



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

Figure 4: Clausal objects in the dependency structure

Figure 5: Clausal objects in the CCGbank



"I came for you"

Figure 6: An adposition in the dependency structure and its CCG label

4.1 Identifying Arguments

The main arguments in a sentence are identified in the first iteration. This is done by identifying the arguments of the root. Following Çakıcı (2005), we differentiated between the types of verbs according to the amount of arguments they can take and we labeled the subject NP argument as NP_{nom}. The category of the root tokens are defined in the first iteration. The algorithm counts the NSUBJ, CSUBJ, OBJ, OBL, CCOMP, and XCOMP arguments[3] that are linked to the verb root. In Çakıcı (2005), there are only three types of arguments defined, namely, subject, object, and oblique. However, the types of arguments are more detailed in the UD annotation framework, and we reflected these differences on our CCG labels. Even though the verbal nouns have the NOUN POS tag, they can take their own arguments. Figure 4 illustrates this in the dependency structure.

The algorithm may add an S instead of an NP to the argument structure of the root token as shown in Figure 5. This way, we differentiate between the verbs that take a clausal argument from the ones that take nominal arguments. The same process applies to the tokens that are linked with the PARATAXIS relation to the root token. This relation is built when two sentences occur together without any coordinator. When this happens, the main verb of the first sentence is linked to the main verb of the second sentence with the PARATAXIS relation in the UD framework. Since this token can have its own arguments, including subjects, the argument structure of such tokens is also identified in the first iteration, together with the root nodes.

[3]NSUBJ=nominal subject, CSUBJ=clausal subject, OBJ=object, OBL=oblique, CCOMP=clausal complement, XCOMP=open clausal complement
The difference between the XCOMP and CCOMP is that the former cannot have its own subject. They both define non-finite complement clauses.

4.2 Combining Adverbs

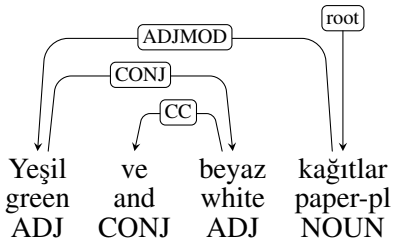
Adverbs can modify sentence heads as well as the other adjuncts such as adjectives. Çakıcı (2005) marks all of these adverbs as S/S categories in order to prevent the generation of giant categories. However, adjectives are modifiers of noun phrases and they cannot combine with an S/S category. They are type NP/NP. Anything that modifies an adjective is marked as ADVMOD. Therefore, we treated the adverbial modifiers as categories of X/X where X is the category of the modified token. The following illustrates this composition.

Daha	sağlıklı	yemekler	yedi
more	healthy	food-pl	eat-PAST
(NP/NP)/(NP/NP)	NP/NP	NP	S\NP

4.3 Adpositions

Turkish does not have any prepositions (Göksel and Kerslake, 2004). However, there are postpositions (PP's) and they are frequently used. In a phrase structure treebank, the PP heads would be the constituent heads. However, in the dependency treebanks, they are treated as the dependents of the head noun of a constituent. Postposition relations are labeled as either CASE or MARK. The former is used for the case marking elements that are connected to nouns and the latter is used for postpositions that introduce finite clauses. Figure 6 illustrates the backward relation of the postpositions to their head nouns.

We employed a type raising rule to account for the fact that the adposition is not a constituent head in the dependency structure but it actually defines



"Green and white papers"

Figure 7: Conjunction in the dependency treebank

the type of relation with the main verb. Therefore, the CCG of the tokens with MARK and CASE labels are determined as S/S\X. This way, they combine with their own constituent heads, and then combine with the matrix verb.

4.4 Conjuncts

We followed Çakıcı (2005) for the conjunctions that consist of the same category. These type of conjuncts are assigned the category (X\X)/X. However, since the corpus we employed was bigger, we had to cover the cases where the types of the two conjuncts are different. In the UD framework, the head of the two conjuncts is the first one. Therefore, we labelled the conjunction as category (X\X)/Y.

The conjuncts were annotated in a nested manner. Therefore, when we combine the relations between them, the category of the conjunct becomes bigger than the average complex categories. Figure 7 illustrates the dependency annotation rule for conjuncts. The conjunct in Figure 7 is connected to the second adjective, *beyaz* and the second adjective is linked to the first one, *yeşil*. The first adjective modifies the head noun *kağıtlar*. This back and forth relation between the constituents results in some bigger categories. However, we reflected these in the conjunct to keep our CCGbank transparent to the dependency structure.

4.5 Punctuation

The punctuations are defined with the PUNCT relation in The Turkish Penn Treebank. In the CCGbank, we treated punctuations according to where they occur. The sentence final ones are given the category S\S as they modify the whole sentence. Since the sentences end up with the category S, the sentence final punctuation can take this last S to its left to modify the whole sentence. The ones that occur inside the sentence are treated as modifiers of their head categories.

5 Results

The results follow the dependency based nature of the algorithm. The bigger categories reflect the more complicated dependency relations that need to combine with each other in a CCGbank. This reflects the direct relationship between the UD-style dependency and CCG categories. The frequency of these complex categories show that they do not pose a problem for learning.

Table 2 shows the most frequent 10 word tokens along with their most common CCG categories. The category frequency is higher than the previous works on Turkish CCG induction (Çakıcı, 2005). One reason for this is that our corpus was bigger and it consists of different genres of treebanks. However, our CCGbank is not morphemic, and this should reduce the categorical frequencies. We believe results show that a dependency relation based approach is convenient for CCG induction, given that this feature of the algorithm enables it to be used in a variety of treebanks.

One thing to notice in Table 2 is that the category of "*ve*" meaning "and" consists of X's. The category of this conjunct was not left like this as explained in the previous section. Its actual category is ((NP/NP\(NP/NP))/(NP/NP))/((NP/NP\(NP/NP))/(NP/NP))\((NP/NP\(NP/NP))/(NP/NP)) and this is reduced in the table for space reasons. Each X in this category corresponds to ((NP/NP\(NP/NP))/(NP/NP)). This category is larger than others because of the complex dependency annotation rule for the conjuncts explained previously in section 4.4. The transparent relationship between the UD-style dependency relations and CCG categories sometimes creates this big structures, however, the complexity is not an indicator of rareness. These structures occur frequently and the complex CCG information they have correctly represents which constituents combine with each other.

Another thing Table 2 shows is that the punctuations have categories depending on where they occur in the sentence. For instance, a period most frequently follows a sentence and is therefore labeled as S\S while a quotation mark is labelled as the category S/S because it is mostly combined with the predicate of the quoted sentence which comes after it.

Table 2 also shows that *için*, meaning "for", is one of the most frequent postpositions in the corpus. Its category type shows that this postposition was

mostly taking intransitive verbs. This is because NP_{nom} marks the subjects, and the lack of a bare NP in the category signals that these verbs do not have an object. This kind of information is rendered available by the application of previous approaches in the CCGbank induction that divides the transitive verbs from the intransitive ones (Çakıcı, 2005).

token **most freq. cat.**	**freq.** **cat. freq.**
. S\S	48274 46692
, S/S	13110 2675
bir NP/NP	10830 9596
ve X/X\X	4506 993
çok $S\NP_{[nom]}$ / $S\NP_{[nom]}$	4444 1657
bu NP/NP	3605 3098
da /de $NP_{[nom]}$/$NP_{[nom]}$	2795 730
için S/S\NP	2031 856
güzeldi $S\NP_{[nom]}$	1624 1270
ile S/S\/NP	1574 503

Table 2: The most frequent 15 tokens

Table 3 shows the 15 most frequent word categories, their frequency count and their parts of speech information in the CCGbank we created. The frequent categories reflect the translated nature of the sentences. For instance, the frequency of the verbs in pro-drop sentences is higher than the frequency of the verbs in non-pro-drop sentences. The adverb frequency also reflect this distribution. In the previous studies pro-drop sentences were also more common (Çakıcı, 2005). We think this correlation reflects the nature of the language. However, the amount of translated corpora in our study decreases the amount of pro-drop verbs. Further exploration is needed to study the effects of using translated corpora.

There are 630 different categories in this treebank. This number is only a hundred above the previous studies held in Turkish even though this corpus is 60 times bigger than the previous works.

cat. type	**freq.**	**pos**
NP/NP	94298	ADJ
NP	55580	NOUN
S\S	51707	ADV
$NP_{[nom]}$	35409	NOUN
S	25413	VERB
S \$NP_{[nom]}$	24780	VERB
S/S	22686	ADV
$NP_{[nom]}$ / $NP_{[nom]}$	18453	ADJ
$S\NP_{[nom]}$ / $S\NP_{[nom]}$	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
$(S\NP_{[nom]})$ \NP	3350	VERB

Table 3: The most frequent 15 categories

The IMST corpus used in (Çakıcı, 2005, 2009) had 60k words while the total word count of our corpus has 516k words. We think that this shows that a lexical approach that can be applied to all dependency treebanks of Turkish results in a quite convenient CCGbank corpus.

6 Conclusion

In this study, we presented the process of inducing a CCGbank for Turkish from an existing dependency treebank. We employed a transparent algorithm that can be applied to all the Turkish treebanks in the UD framework without any adjustment. We introduced the dependency treebanks used in this study along with their annotation framework. We stated the consequences of a direct induction from dependency structures to the CCG approach through certain phenomenon that was also argued in the previous literature. We also showed the similarities and differences between our algorithm and the previous studies conducted in Turkish for CCGbank induction.

This approach already results in a consistent and parsable CCG corpus. However, the Turkish annotation scheme in the UD framework becomes more morphemic in each release and we believe the adaption to the future releases of the UD annotations can easily turn our lexeme based algorithm to a morphemic one without any complication. We hope this corpus to be useful in the upcoming Turkish semantic parsing studies.

References

Burak Kerim Akkuş. 2014. Supertagging with combinatory categorial grammar for dependency parsing. Master's thesis, Middle East Technical University.

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.

Nart B Atalay, Kemal Oflazer, and Bilge Say. 2003. The annotation process in the turkish treebank. In *Proceedings of 4th International Workshop on Linguistically Interpreted Corpora (LINC-03) at EACL 2003*.

Rajesh Bhatt, Bhuvana Narasimhan, Martha Palmer, Owen Rambow, Dipti Misra Sharma, and Fei Xia. 2009. A multi-representational and multi-layered treebank for hindi/urdu. In *Proceedings of the Third Linguistic Annotation Workshop (LAW III)*, pages 186–189.

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.

Sabine Brants, Stefanie Dipper, Peter Eisenberg, Silvia Hansen-Schirra, Esther König, Wolfgang Lezius, Christian Rohrer, George Smith, and Hans Uszkoreit. 2004. Tiger: Linguistic interpretation of a german corpus. *Research on language and computation*, 2(4):597–620.

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.

Marie-Catherine De Marneffe, Bill MacCartney, Christopher D Manning, et al. 2006. Generating typed dependency parses from phrase structure parses. In *Lrec*, volume 6, pages 449–454.

Marie-Catherine de Marneffe, Christopher D Manning, Joakim Nivre, and Daniel Zeman. 2021. Universal dependencies. *Computational linguistics*, 47(2):255–308.

Aslı Göksel and Celia Kerslake. 2004. *Turkish: A comprehensive grammar*. Routledge.

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.

Bos Johan, Cristina Bosco, and Alessandro Mazzei. 2009. Converting a dependency treebank to a categorial grammar treebank for italian. In *Eight international workshop on treebanks and linguistic theories (TLT8)*, pages 27–38. Educatt.

Aslı Kuzgun, Neslihan Cesur, Bilge Nas Arıcan, Merve Özçelik, Büşra Marşan, Neslihan Kara, Deniz Baran Aslan, and Olcay Taner Yıldız. 2020. On building the largest and cross-linguistic turkish dependency corpus. In *2020 Innovations in Intelligent Systems and Applications Conference (ASYU)*, pages 1–6. IEEE.

Mitchell P. Marcus, Beatrice Santorini, and Mary Ann Marcinkiewicz. 1993. Building a large annotated corpus of English: The Penn Treebank. *Computational Linguistics*, 19(2):313–330.

Kemal Oflazer, Bilge Say, Dilek Zeynep Hakkani-Tür, and Gökhan Tür. 2003. Building a turkish treebank. In *Treebanks*, pages 261–277. Springer.

Daniel Tse and James R. Curran. 2010. Chinese CCGbank: extracting CCG derivations from the Penn Chinese treebank. In *Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010)*, pages 1083–1091, Beijing, China. Coling 2010 Organizing Committee.

Wayne Ward. 1990. The cmu air travel information service: Understanding spontaneous speech. In *Speech and Natural Language: Proceedings of a Workshop Held at Hidden Valley, Pennsylvania, June 24-27, 1990*.

Olcay Taner Yıldız, Begüm Avar, and Gökhan Ercan. 2019. An open, extendible, and fast turkish morphological analyzer.