# Overcoming the challenges in morphological annotation of Turkish in universal dependencies framework

**Seyyit Talha Bedir**
Dept. of Linguistics
Bogazici University, Turkey
`talha.bedir@boun.edu.tr`

**Karahan Şahin**
Dept. of Linguistics
Bogazici University, Turkey
`karahan.sahin@boun.edu.tr`

**Onur Güngör**
Computer Engineering
Bogazici University, Turkey
`onurgu@boun.edu.tr`

**Suzan Üsküdarlı**
Computer Engineering
Bogazici University, Turkey
`suzan.uskudarli@boun.edu.tr`

**Arzucan Özgür**
Computer Engineering
Bogazici University, Turkey
`arzucan.ozgur@boun.edu.tr`

**Tunga Güngör**
Computer Engineering
Bogazici University, Turkey
`gungort@boun.edu.tr`

**Balkız Öztürk**
Dept. of Linguistics
Bogazici University, Turkey
`balkiz.ozturk@boun.edu.tr`

## Abstract

This paper presents several challenges faced when annotating Turkish treebanks in accordance with the Universal Dependencies (UD) guidelines and proposes solutions to address them. Most of these challenges stem from the lack of adequate support in the UD framework to accurately represent null morphemes and complex derivations, which results in significant loss of information for Turkish. This loss negatively impacts the tools that are developed based on these treebanks. We raised and discussed these issues within the community on the official UD portal. This paper presents these issues and our proposals to more accurately represent morphosyntactic information for Turkish while adhering to guidelines of UD. This work aims to contribute to the representation of Turkish and other agglutinative languages in UD-based treebanks, which in turn aids to develop more accurately annotated datasets for such languages.

## 1 Introduction

Universal Dependencies[1] (henceforth, UD) is an international cooperative project for annotating treebanks with a standardized format to facilitate the development of automated support for natural language processing (NLP). At present (version 2.8) it includes 202 treebanks from 114 different languages. In this paper, we outline and propose solutions to two main challenges for the UD based

annotation of Turkish, which is a highly inflectional agglutinative language. These challenges stem from the inadequate representation of derivations and null morphemes. In addition, we also propose a thorough annotation scheme for a commonly used Turkish verb, *ol-* (to be), which has a very rich scope of usage.

The first challenge that we address is the lack of representation of derivations in UD, which causes annotators to take derived forms of nouns, verbs, adjectives, and adverbs as lemmas. Turkish has a morphology that uses very rich and productive derivational suffixes. Kapan (2019) states that a Turkish word may have up to six derivational suffixes at the same time. Without any tool for derivation, for example, a word like *okuldakilerden* (from the ones in school) would have the lemma *okuldaki*, losing the annotation for the morphemes *okul* (school), and *-da* (locative case suffix) that comes before the derivation. Since the English translation of the same word consists of five different words, *from the ones in school*, it is represented with five dependencies in the UD framework. It annotates much more information than the current annotation scheme of Turkish does. This unparallelism may harm future cross-linguistic analyses. Properly locating the roots of such derived nouns in the system might be also useful in lemmatization processes. In this paper, we aim to contribute to the representation of derivation in UD, which will be helpful for the annotations of agglutinative languages like Turkish.

The second challenge is the lack of official rep-

---

[1]The official website is https://universaldependencies.org/.

resentation for null morphemes. Accounting for this type of morphemes is also considered vital by our team, since the Turkish copula is frequently encountered as a null morpheme. Except for present tense copula, most Turkic treebanks split copulas which alleviates the problem. However, we did not find any account for the present tense copula which is always null. Here, we also propose an annotation for the Turkish present tense copular paradigm.

Besides these two challenges, we also target the presentation of a more complete annotation scheme for the commonly used yet ambiguous Turkish verb *ol-* (to be), for which no clear annotation scheme exists in the Turkish UD treebanks such as BOUN (Türk et al., 2020) and IMST-UD (Sulubacak et al., 2016b). We detected that different usages of *ol-* were usually being used interchangeably with each other. In this paper, we introduce a list of all the *ol-* types and propose an annotation scheme for each.

According to our proposed solutions for the outlined challenges, we started re-annotating the recently introduced BOUN Treebank (2,000 sentences have been annotated so far), which is one of the largest UD-based Turkish treebanks (Türk et al., 2020). Our re-annotation of BOUN Treebank entails all of the morphological and syntactic annotations in the UD framework. That is, we add, remove or change lemmas, parts-of-speech (POS) tags, features and their values, dependency relations and other miscellaneous information as per the decisions taken by the annotation team. Our team is comprised of three linguists and four computer scientists who are all native speakers of Turkish.

In the morphological part of the 2,000 sentences that we have annotated, we changed 5,418 parts-of-speech tags, added 12,654 features, and removed 9,513 features. In the syntactic part, we segmented 701 words into multiple words as well as altered 5,456 dependency relations and 7,333 heads. We also annotated 2,895 slots in the miscellaneous information tab.

While taking decisions for the annotation scheme, the main objective is to represent the Turkish morphosyntax as accurately as possible in accordance with the UD guidelines.[2] We believe, the proposed annotation solutions will contribute to more precise representation not only for Turkish, but also for other agglutinative languages in the UD framework as well.

The remainder of the paper is organized as follows. Section 2 includes a brief survey about Turkish treebanks. Section 3 deals with the problem of derivation in UD. We touch on the difficulties of representing Turkish in a derivation-free framework such as UD. In Section 4, we comment on the representation of the null present tense copula in UD which discourages the representation of any type of null elements in a sentence. In Section 5, we explain how we resolve the multilayered ambiguity problem the verb *ol-* (to be) - a polysemous multi-functional verb - creates for UD. Section 6 concludes the paper.

## 2 Related Work

As stated in the UD website UD started as a joint result of Stanford dependencies (De Marneffe et al., 2006; De Marneffe and Manning, 2008; De Marneffe et al., 2014), Google universal parts-of-speech tags (Petrov et al., 2011) and the Interset interlingua for morphosyntactic tagsets (Zeman, 2008).

Treebank efforts date back much earlier. With Leech and Garside (1991) and Marcus et al. (1993), we see the earliest treebanks in English. About a decade later Turkish treebanks started to emerge when Atalay et al. (2003) and Oflazer et al. (2003) introduced the METU-Sabancı Treebank (MST) which consists of 5,635 sentences. This treebank was then revisited by Sulubacak et al. (2016a) to alleviate some issues attributed to the corpus like excessive parsing difficulty and cross-parser instability. As a result, they proposed the ITU-METU-Sabancı Treebank (IMST). In Sulubacak et al. (2016b) they converted IMST into UD framework via a semi-automated process and mapping, naming it as IMST-UD which is one of the first Turkish UD treebanks. This treebank was later re-annotated by Türk et al. (2019b), improving the annotation of embedded structures and core versus non-core dependents.

The first Turkish treebank that adopts the UD framework is Grammar Book Treebank (GB) by Çöltekin (2015). GB treebank draws 2,803 sentences or sentence fragments from the grammar book of Göksel and Kerslake (2005) and uses TR-Morph (Çöltekin, 2010) to provide morphological annotation. One downside of this treebank is that it mainly consists of textbook examples rather than naturally occurring language.

Another important treebank effort was Turkish-PUD Treebank which was published as a part of the

---

CoNLL 2017 Shared Task on Multilingual Parsing from Raw Text to Universal Dependencies (Zeman et al., 2017). Turkish-PUD Treebank consists of 1000 sentences which are first annotated as per the annotation guidelines of Google and then converted into the UD framework. This treebank is especially noteworthy, since it has parallel annotations for 18 different languages, making it a useful resource for cross-linguistic analyses. This treebank was later re-annotated by Türk et al. (2019a) to make it more linguistically consistent.

One of the largest Turkish UD based treebanks, BOUN Treebank[3], has been presented recently by Türk et al. (2020) with 9,761 manually annotated sentences. These sentences are a subset of Turkish National Corpus (TNC), a corpus first presented by Aksan et al. (2012) that consists of more than 50 million words. The BOUN Treebank sampled a great variety of text types, which are essays, broadsheet national newspapers, instructional texts, popular culture articles, and biographical texts. Before the manual annotation, they used an end-to-end pipeline (Kanerva et al., 2018) to parse raw texts to UD dependencies in CoNLL-U format. The output was then re-annotated by Turkish linguists. This is also the treebank our team is currently working on.

## 3  Problems in derivation

One of the UD's main pitfalls when it comes to annotating a language like Turkish is the lack of derivation in the system. Being mostly a syntax-based project, UD only prefers to mark syntactically relevant morphological distinctions, namely the inflectional morphology. The UD website gives the example of the word *organizations* in English whose lemma is marked as *organization* as opposed to *organize*. The suffix *-tion* is not annotated. In a way, the suffixes till the last derivation are ignored by the system. We can show the general idea as follows. Say, `Infl1`, `Infl2`,... are inflectional suffixes and `Der1`, `Der2`,... are derivational suffixes. For a word like `Root + Infl1 + Der1 + Infl2 + Infl3 + Der2 + Infl4`, the entirety of `Root + Infl1 + Der1 + Infl2 + Infl3 + Der2` is a fixed, unprocessed, single lemma according to the UD guidelines. `Der2`, the last derivative suffix here, *blocks access* to the annotation of the previous part (i.e. `Infl1 + Der1 + Infl2 + Infl3`), making all of them invisi-

ble to the system – examples are provided in Section 3.1 to illustrate this phenomenon. This is because derivation is not annotated and it is included as a part of the lemma under the UD framework.

The current solutions for the languages that use derivation are not unified. For example, Quechua, a language with an extremely agglutinative morphology (Rios et al., 2012), devised a derivation annotation strategy explained in a UD thread.[4] They use word split immensely and annotate verbs using expressions like `Root_VDeriv_VDeriv` instead of features. They also add the functions of those derivations like `VRoot|+Dir|+Inch` where `+Dir` and `+Inch` represent functions of each individual derivative suffix. Since this method annotates each derivative suffix precisely, it provides a more fine-grained approach for derivation annotation than our approach. However, one downside for this approach is that it has a very distinct annotation style than those we see on other treebanks. A different system than classical feature and value system proposed by the UD guidelines is utilized in this treebank. One other derivation scheme we examined was that of Finnish. In the Finnish TDT treebank, they annotate derivations under the `Features` tab using an additional feature called `Derivation`. The value of this feature is the derivative suffix itself, e.g. `Derivation=Minnen`. This approach is useful since it marks the derivative suffix itself. However, with this scheme, it is hard to annotate a derived item when it has more than one derivative suffix.

Turkish sometimes uses derivation so productively, even to the point that it becomes syntactically significant. In the following, we will introduce the derivation related problems Turkish presents and how we handled them via the tools permitted under the UD guidelines.

### 3.1  Utilizing MISC tab

In many languages, derivation is expected to occur closer to the root, i.e. inflections usually follow the derivation. In Turkish, albeit this is the general tendency, there are a lot of instances in which a derivational affix modifies an inflected item:[5]

---

(1) anne-m-siz
mother-1SG.POSS-DEPR
'without my mother'

*-sIz*[6] is a derivational deprivative suffix that adds the meaning of "without". In (1) it comes after the first person possessive suffix which is an inflection. This word will appear as a lemma with its full form (*annemsiz*) in UD. There are even instances in which a derivational affix modifies a phrase or a full clause as shown by the derivational suffix *-CI* in (2).

(2) nasılsa  sabah  erken
anyways morning early
kalk-ar-ım-cı
get.up-AOR-1SG-PROF
'a person who lives by the motto: "I can get up early anyways"'

*nasılsa sabah erken kalkarım* (I can get up early anyways) is a finite sentence, *nasılsa sabah erken kalkarımcı* is a person who would say this a lot, procrastinating till midnight and still believing that he/she would get up early anyways. Although this type of usage is rare, in UD we have absolutely no way to deal with the syntax of this sort of expressions. For example, *kalkarımcı*, the last word in the phrase, would have the lemma *kalkarımcı*, since the derivation blocks access to the interior inflectional suffixes, hence *-(A)r* (the aorist TAM (Tense-Aspect-Modality) marker) and *-(I)m* (first person singular agreement marker) will be invisible to the UD.
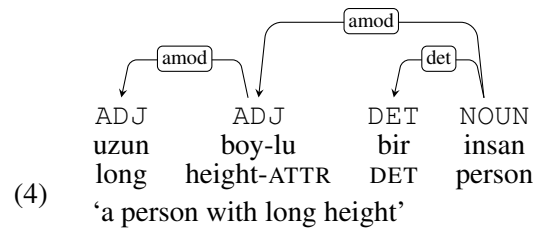
Although examples like (2) are rare, there are other very common situations like (3) where a derivational affix modifies an entire phrase and it is impossible to annotate it justly with the tools of UD.

(3) [uzun boy]-lu     bir  insan
[long  height]-ATTR DET person
'lit: a person with long height'

---

PST = past, PTCP = participle, SG = singular.

[6]The capital letters in the suffixes throughout the paper indicate vowel harmony. In Turkish, like many other Turkic languages, high vowels are subject to a phonological harmonization process with regard to frontness and roundness. That means a suffix like *-sIz* might come in four ways depending on the vowel in the syllable before it: *-sız*, *-siz*, *-suz* and *-süz*. Suffixes with lower vowels such as *-(y)A* only vary as per frontness: *-(y)a* and *-(y)e*. In some suffixes, there are also consonants that are written in capital such as *-DA*. This is because the consonant /d/ may change to /t/ if a voiceless consonant comes immediately before it.

*-lI* suffix is modifying the entire phrase in brackets. For *uzun boylu bir insan*, the current UD tree representation[7] is shown in (4).

(4)

| ADJ | ADJ | DET | NOUN |
|-----|-----|-----|------|
| uzun | boy-lu | bir | insan |
| long | height-ATTR | DET | person |

'a person with long height'

This annotation is problematic for two reasons:[8]

- Because *-lI* is a derivational suffix, we cannot represent it in the UD framework. Should we represent it in some way, in UD framework *-lI* would only modify *boy* whereas the correct syntactic explanation is that the suffix *-lI* modifies the whole noun phrase (NP) *uzun boy* rather than just *boy*.[9]

- In Turkish, like many languages, adjectives are modified by adverbs rather than adjectives. However, in the current annotation shown in (4); an adjective, *uzun*, modifies another adjective, *boylu*, via the dependency amod (short for "adjectival modifier").

To solve the problem, the first thing we did was to indicate the root of the derived item utilizing the miscellaneous slot[10] as UD lacks traditional ways of parsing derivation. We use our made up function called df, which is an abbreviation of "derived from". For instance, considering the word *organization* in English its MISC slot will read df=organize. For us to annotate the roots of

---

[7]In this format the arrows above represent dependencies. The first line gives POS tags, second line is the text, third line provides glosses for the text and the fourth line is the translation of the text.

[8]This example and the potential problems listed here have also been presented by one of the authors of this paper on https://github.com/UniversalDependencies/docs/issues/787. Although great ideas were exchanged by fellow linguists that participated in the discussion, the issue remains unresolved at the time of this writing.

[9]Note that this type of phrasal suffixation (a suffix modifying the entire phrase rather than a single word) can be inflectional as well. However, since inflectional suffixes are fully annotated in the UD framework, they do not cause any information loss. But, the derived elements are taken as whole units in UD causing information loss. Our goal in this paper is minimizing this information loss without leaving the boundaries of official UD guidelines.

[10]The miscellaneous slot or MISC is an additional slot that is placed on the rightmost tab in CONLL-U format that represents additional information. This field is usually looser than other fields in that it does not have any format restriction and it is up to the annotator team how to utilize it.

a derived word via the use of `df` under the `MISC` field, the derivational suffix(es) attached to the root have to be very productive. For example, the word *boylu* in (3) above would have `df=boy` on the `MISC` field, since the affix *-lI* is judged to be productive and the root *boy* is deemed accessible to any native Turkish speaker.

There are many unproductive derivative suffixes in Turkish as well. The derivations in (5) are excluded from the root annotation process explained above as they are not judged to be parsable by native Turkish speakers.

(5) a. yağ-mur  
      rain(v)-MUR  
      'rain(n)'

    b. bas-amak  
      step.on(v)-AMAK  
      'step(n)'

The suffixes in (5) generally are unproductive and selected by very few verbal roots. Others are a little bit more productive. The full list of the productive derivative affixes that we deemed accessible by native speakers is given in Appendix A. This list was determined by our linguistic team. Other derivations were not annotated.

By annotating the roots of the derived nouns, verbs, adjectives and adverbs by using the MISC tab, we are providing a simple input for derived items. This makes the information regarding the productive derivation accessible, which we believe to be an important step towards introducing derivation to UD.

### 3.2 The *-ki* derivation

Another case of productive derivation which introduces a challenge for UD is the derivative suffix *-ki*, which is so productive that we had to split it off even though it is discouraged in the UD framework.

It has been established in Turkish syntactic literature that Turkish mainly has two types of *-ki* suffixes: adjectivizer *-ki* and pronominal *-ki* (Hankamer, 2004, 2005). We separate both of them.

#### 3.2.1 Adjectivizer *-ki*

The adjectivizer suffix *-ki* is generally preceded by a temporal noun as in (6a) or a noun in the locative case which is an inflectional suffix as in (6b):

(6) a. geçen sene-ki program  
      last  year-KI program  
      'the program in the last year'

    b. okul-da-ki    öğrenci-ler-in  
      school-LOC-KI student-PL-GEN  
      'of the students at school'

Since there is currently no dependency relation in UD to mark any derivation, we had to make our own: `dep:der`. `dep` is a canonical dependency in the UD framework which is advised to be used when no other dependency fits as noted in the official UD website. `dep:der` is our subtype of `dep` which stands for "derivation".

The use of `dep:der` relation is exemplified in Figure 1b which shows the annotated form of (6a). The reason why it is chosen over the old annotation (Figure 1a) is that in the old annotation the features of the word *sene* are lost, and in the revised version the adjective *geçen* is correctly modifying the noun *sene*, rather than the adjective *seneki*.

By introducing the `dep:der` relation, we are able to account for all of the inflection before and after *-ki*. For instance, if the *-ki* suffix in *okuldaki* in (6b) is not separated, then the entirety of the word will be regarded as a single lemma and there will be no way to annotate the locative case suffix *-da* since it is situated before the derivative suffix *-ki*.
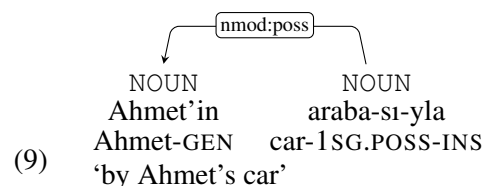
#### 3.2.2 The pronominal *-ki*

This *-ki* is a substitute for the possessive noun in Turkish classical genitive-possessive noun phrases (Hankamer, 2004; Öztürk and Taylan, 2018). This *-ki* is also split and its annotation is completely parallel to the item it substitutes.

Example (7) shows a genitive-possessive construction in Turkish and (8) illustrates how *-ki* substitutes the possessive noun *arabası* (his/her car):

(7) Ahmet-in    araba-sı-yla  
    Ahmet-GEN car-3SG.POSS-INS  
    'by Ahmet's car'

(8) Ahmet-in-ki-yle  
    Ahmet-GEN-KI-INS  
    'by that of Ahmet's'

The "that" in the translation of (8) is obviously a fill-in-the-blank item whose content is to be derived by the context. In our case, it refers to *arabası* (his car). Therefore, we kept the dependency construction parallel to that of (7), except for the possessive suffix which goes away during the substitution by *-ki*. That is, the *-ki* is regarded as a pronoun that substitutes the possessive noun in the phrase rather than as a suffix.

(9)



NOUN         NOUN  
Ahmet'in    araba-sı-yla  
Ahmet-GEN   car-1SG.POSS-INS  
'by Ahmet's car'

| Token | Form | Lemma | UPOS | XPOS | Features | Head | Deprel | MISC |
|---|---|---|---|---|---|---|---|---|
| 1 | geçen | geçen | ADJ | Adj | _ | 2 | amod | _ |
| 2 | seneki | seneki | ADJ | Adj | _ | 3 | amod | _ |
| 3 | program | program | NOUN | Noun | Case=Nom\|Number=Sing\|Person=3 | 0 | root | _ |

(a) The original annotation of *geçen seneki program* in CONLL-U format

| Token | Form | Lemma | UPOS | XPOS | Features | Head | Deprel | MISC |
|---|---|---|---|---|---|---|---|---|
| 1 | geçen | geçen | ADJ | Adj | _ | 2 | amod | _ |
| 2-3 | seneki | _ | _ | _ | _ | _ | _ | _ |
| 2 | sene | sene | NOUN | Noun | Case=Nom\|Number=Sing\|Person=3 | 4 | nmod | _ |
| 3 | ki | ki | PART | Attr | _ | 2 | dep:der | _ |
| 4 | program | program | NOUN | Noun | Case=Nom\|Number=Sing\|Person=3 | 0 | root | _ |

(b) The revised annotation of *geçen seneki program* in CONLL-U format

Figure 1: Comparison of the previous and the proposed annotations for *geçen seneki program*

| Token | Form | Lemma | UPOS | XPOS | Features | Head | Deprel | MISC |
|---|---|---|---|---|---|---|---|---|
| 1 | Ahmet'inkiyle | Ahmet'inki | NOUN | Noun | Case=Ins\|Number=Sing\|Person=3 | 0 | root | _ |

(a) The original annotation of *Ahmet'inkiyle* in CONLL-U format

| Token | Form | Lemma | UPOS | XPOS | Features | Head | Deprel | MISC |
|---|---|---|---|---|---|---|---|---|
| 1-2 | Ahmet'inkiyle | _ | _ | _ | _ | _ | _ | _ |
| 1 | Ahmet'in | Ahmet | PROPN | Prop | Case=Gen\|Number=Sing\|Person=3 | 2 | nmod:poss | _ |
| 2 | kiyle | ki | PRON | Partic | Case=Ins\|Number=Sing\|Person=3 | 0 | root | _ |

(b) The revised annotation of *Ahmet'inkiyle* in CONLL-U format

Figure 2: Comparison of the previous and the proposed annotations for *Ahmet'inkiyle*

(10)

nmod:poss

NOUN    PRON
Ahmet'in    ki-yle
Ahmet-GEN    KI-INS
'by that of Ahmet's'
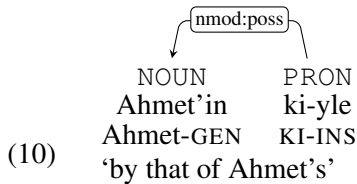
If we did not separate *-ki* like in Figure 2a, the entirety of *Ahmet'inki* would be taken as the root that only bears the instrumental case suffix *-yle*. In that case, we would lose the annotation of the genitive case suffix *-in* that is situated before *-ki* and be unaware of the fact that *-ki* acts as a pronoun that substitutes a noun phrase. The dependency tree in (10) and its CONLL-U form in Figure 2b exhibit our new annotation which correctly marks this *-ki* as a pronoun and annotates the features of *Ahmet'in* which were lost in Figure 2a.

## 4 Lack of null morphemes

In UD, there is no official way to represent null items. This presents a challenge for Turkish since one of the allomorphs of Turkish copula is a null morpheme. This section presents potential annotation problems in UD stemming from the lack of null morphemes, and proposes a solution without imposing radical changes such as introducing null morphemes to the UD framework.

Turkish copula has three allomorphs: *-y-*, *i-* and ∅ (Göksel and Kerslake, 2005; Kornfilt, 1996;

Kelepir, 2001). *i-* only surfaces when the copula is written separately. When it is connected to the predicate it can appear as *-y-* or ∅ depending on the last sound of the predicate, respectively, being a vowel or a consonant. Examples in (11) show the separated variation of the Turkish copula for the past tense. Examples in (12) show the same sentences but with the attached form of the copula. The only difference might be the slightly less formal register in the sentences in (12).

(11)  a.  Ben öğretmen i-di-m.
        I    teacher  COP-PST-1SG
        'I was a teacher.'

      b.  O     öğrenci i-di-∅.
        S/he student COP-PST-3SG
        'S/he was a student.'

(12)  a.  Ben öğretmen-∅-di-m.
        I    teacher-COP-PST-1SG
        'I was a teacher.'

      b.  O     öğrenci-y-di-∅.
        S/he student-COP-PST-3SG
        'S/he was a student.'

The sentences in (11) and (12) have the identical meaning. The only difference is that in (12) copulas are in affix form. Except for the present tense copula, all other affixed copulas are already segmented in previous Turkish treebanks.

We added segmentation of the present tense copula which is always null as shown in (13). The previous annotation of this sentence is shown in Figure 3a. We consider this annotation problematic since the `Features` slot contains only the features of the first person copular suffix *-(y)Im*, that is `Number=Sing|Person=1`. In order to represent the features of the nominal predicate, like *öğrenci* (student), we decided to split the present tense copula as shown in Figure 3b – not a common practice in Turkic UD treebanks.[11]

(13)  Ben bir  öğrenci-∅-yim.
      I    DET student-COP.PRS-1SG
      'I am a student.'

In Turkish, the 3rd person paradigm almost always bears a null suffix. This means that we are unable to split it as there is nothing to split. Table 1 shows the present tense copular forms in Turkish. In the 3rd person singular form, we end up with two successive null suffixes. The only case when an overt morpheme may be 3rd person is the optional *-lAr* after the third person plural copula and only if the subject is human. If *-lAr* shows up in such a case, we separate it just like in Figure 3.

| Person | Singular | Plural |
|---|---|---|
| 1 | öğrenci-∅-yim | öğrenci-∅-yiz |
|  | 'I am a student' | 'we are students' |
| 2 | öğrenci-∅-sin | öğrenci-∅-siniz |
|  | 'you are a student' | 'you are students' |
| 3 | öğrenci-∅-∅ | öğrenci-∅-(ler) |
|  | 'he/she is a student' | 'they are students' |

Table 1: Turkish present tense copular paradigm

One solution to solve this issue might be introducing empty nodes and categories into our UD representation. However, our team judged that it would be too unorthodox for the current form of the UD framework. Instead we decided to introduce a compromised solution which includes utilizing the `MISC` slot.

We introduce two new features, `nullcop=3s` and `nullcop=3p` which stand for, respectively, 3rd person singular null copula and 3rd person plural null copula. These are written under the `MISC` slot of the nominal or adjectival predicate that is in agreement with a third person subject.

---

[11]This decision emerged based on discussions initiated by an author of this paper on https://github.com/UniversalDependencies/docs/issues/774. The issue is closed and idea of splitting copulas is favored.

An example is shown below, and its CONLL-U annotation is given in Figure 4:

(14)  İşçi-ler     grev-de-∅-∅.
      worker-PL  strike-LOC-COP-3PL
      'Workers are on strike.'

## 5 Turkish *ol-* verb and its many faces

Turkish *ol-* (to be) is an extremely common verb which has many usages. Unfortunately, to this day, not all of its uses are dealt with properly in Turkish treebanks. Since this verb can act as a copula, auxiliary verb, light verb and as a verb in itself, it is extremely important to have a system that clearly differentiates among them. Since in UD it is not possible to input more than one entry for the same item, it is crucial to make that difference visible via part-of-speech tags `UPOS` and `XPOS`, as well as via features and dependency relations within the framework of UD. Six ol- variants (Table 2) are detailed in the following sections.

| ol- type | UPOS (XPOS) | dependency relation |
|---|---|---|
| *ol-* the intransitive verb | VERB | regular verb |
| *ol-* 'to become' | VERB | regular verb, where an `xcomp` interior argument is expected |
| *ol-* the embedded copula | AUX | connected to its dependent with `cop` like a regular copula |
| *ol-* the embedded *var/yok* | VERB (Exist) | dependencies of regular verb |
| *ol-* the auxiliary | AUX | connected to its host with `aux` |
| *ol-* the light verb | VERB | connected to its dependent with `compound:lvc` |

Table 2: The *ol-* types that we classified

### 5.1 *ol-* as intransitive verb

As an intransitive verb, the verb *ol-* can have a variety of meanings such as 'to be fit' or 'to be ripe'. In these cases, it is annotated just like a regular intransitive verb as seen in (15).

| Token | Form | Lemma | UPOS | XPOS | Features | Head | Deprel | MISC |
|---|---|---|---|---|---|---|---|---|
| 1 | Ben | ben | PRON | Pers | Case=Nom\|Number=Sing\|Person=1\|PronType=Prs | 3 | nsubj | _ |
| 2 | bir | bir | DET | Det | _ | 3 | det | _ |
| 3 | öğrenciyim | öğrenci | NOUN | Noun | Case=Nom\|Number=Sing\|Person=1 | 0 | root | _ |

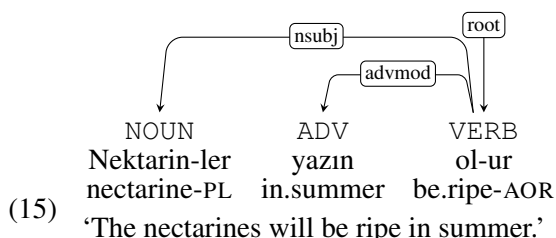(a) The original annotation of *Ben bir öğrenciyim* in CONLL-U format

| Token | Form | Lemma | UPOS | XPOS | Features | Head | Deprel | MISC |
|---|---|---|---|---|---|---|---|---|
| 1 | Ben | ben | PRON | Pers | Case=Nom\|Number=Sing\|Person=1\|PronType=Prs | 3 | nsubj | _ |
| 2 | bir | bir | DET | Det | _ | 3 | det | _ |
| 3-4 | öğrenciyim | _ | _ | _ | _ | _ | _ | _ |
| 3 | öğrenci | öğrenci | NOUN | Noun | Case=Nom\|Number=Sing\|Person=3 | 0 | root | _ |
| 4 | yim | N/A | AUX | Zero | Number=Sing\|Person=1\|Polarity=Pos\|Tense=Pres | 3 | cop | _ |

(b) The revised annotation of *Ben bir öğrenciyim* in CONLL-U format

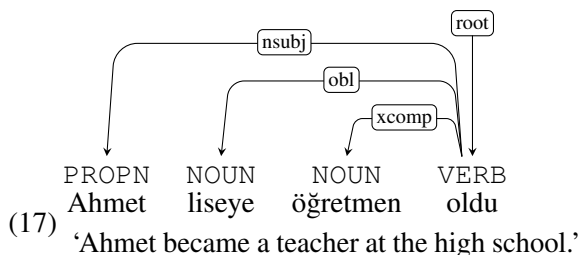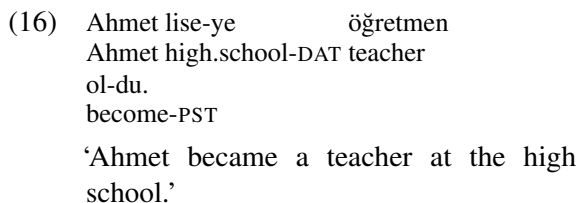Figure 3: Comparison of the previous and the proposed annotations for *Ben bir öğrenciyim*

| Token | Form | Lemma | UPOS | XPOS | Features | Head | Deprel | MISC |
|---|---|---|---|---|---|---|---|---|
| 1 | İşçiler | işçi | NOUN | Noun | Case=Nom\|Number=Plur\|Person-3 | 2 | nsubj | df=iş |
| 2 | grevde | grev | NOUN | Noun | Case=Dat\|Number=Sing\|Person-3 | 0 | root | nullcop=3p |

Figure 4: UD Annotation of *İşçiler grevde*

(15)

nsubj · advmod · root

| NOUN | ADV | VERB |
|---|---|---|
| Nektarin-ler | yazın | ol-ur |
| nectarine-PL | in.summer | be.ripe-AOR |

'The nectarines will be ripe in summer.'
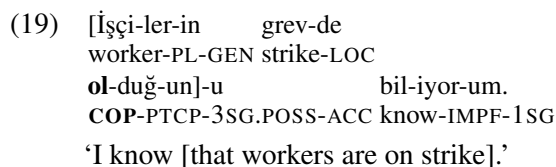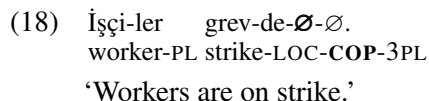
## 5.2 *ol-* that means 'to become'

*ol-* that means 'to become' is annotated just like a regular verb except that its internal argument is a small clause and this type of clauses is annotated with the dependency xcomp. An example is shown in (16) and its dependency tree[12] in (17).
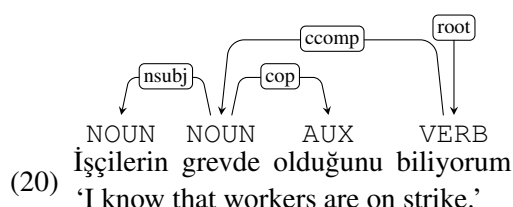
(16) Ahmet lise-ye öğretmen
Ahmet high.school-DAT teacher
ol-du.
become-PST

'Ahmet became a teacher at the high school.'

(17)

nsubj · obl · xcomp · root

| PROPN | NOUN | NOUN | VERB |
|---|---|---|---|
| Ahmet | liseye | öğretmen | oldu |

'Ahmet became a teacher at the high school.'

## 5.3 *ol-* used as a Turkish copula in the embedded structures

In Turkish, copula is represented via the verb *ol-* in genitive-possessive type nominalized embedded clauses (Göksel, 2001). In that regard, it could be considered as the fourth allomorph among other three Turkish copulas mentioned above: *i-*, *-y-* and ∅. Take the sentence in (18) whose embedded form is given in (19) in brackets (copulas are in bold):

(18) İşçi-ler grev-de-**∅**-∅.
worker-PL strike-LOC-**COP**-3PL

'Workers are on strike.'

(19) [İşçi-ler-in grev-de
worker-PL-GEN strike-LOC
**ol**-duğ-un]-u bil-iyor-um.
**COP**-PTCP-3SG.POSS-ACC know-IMPF-1SG

'I know [that workers are on strike].'

The way we annotate (19) is parallel to our annotation for the copulas as shown in (20).

(20)

nsubj · cop · ccomp · root

| NOUN | NOUN | AUX | VERB |
|---|---|---|---|
| İşçilerin | grevde | olduğunu | biliyorum |

'I know that workers are on strike.'

The copular element *olduğunu* in (20) is connected to the embedded nominal predicate *grevde* (on strike) with the cop dependency and its UPOS is AUX as in a regular copula. Notice that in (15) and (17) the UPOS tags of the verb *ol-* were VERB.

---

[12]In longer sentences such as (17), the glosses are given as a separate example like in (16) in order not to exceed the column.
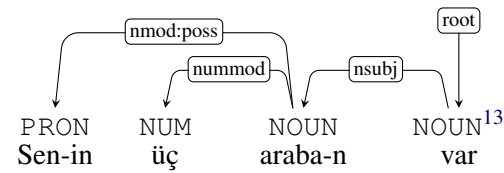
## 5.4 *ol-* which is the embedded form of Turkish existential predicates *var* and *yok*

Turkish utilizes two existential predicates: *var* (there is) and *yok* (there is not), as in:

(21)   Kabin-de  oksijen yok.
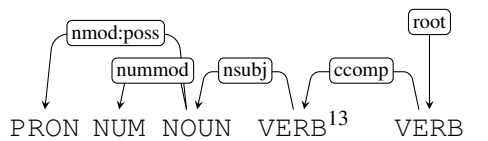        cabin-LOC oxygen YOK

       'There is no oxygen in the cabin.'

Possession is expressed with these existential predicates as:



(22)
PRON   NUM   NOUN   NOUN[13]
Sen-in   üç   araba-n   var
you-GEN  three  car-2SG.POSS  VAR
'You have three cars.'

where *senin* modifies *araban* with the nmod:poss dependency just like in a regular genitive-possessive NP. This is due to the syntactic structure of Turkish; which literally translates to 'Your three cars exist.'

Hence, we decided that its embedded form (within the brackets) should be parallel to (22) even though the predicate changes its form as shown in (23) and (24).

(23)   [Sen-in   üç   araba-n
       [you-GEN three car-2SG.POSS
       ol-duğ-un]-u         bil-iyor-um.
       OL-PTCP-2SG.POSS]-ACC know-IMPF-1SG
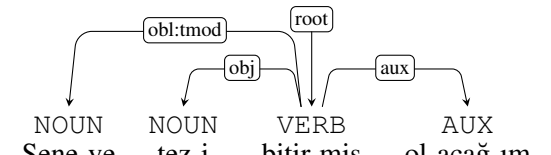
       'I know [that you have three cars].'



(24)
PRON NUM NOUN VERB[13]  VERB
Senin  üç  araban olduğunu biliyorum
'I know that you have three cars.'

Note that although the forms of *olduğunu* in (24) and (20) are identical, their UPOS and XPOS values and dependency structures differ.

## 5.5 *ol-* as auxiliary verb

Turkish also utilizes *ol-* as an auxiliary verb in order to give extra information about the aspect and tense of a verb. We annotate its POS tag as AUX and its dependency as aux. These cases are more easily recognizable since they are most often observed after a participle (e.g. *bitirmiş* in the example (25)).

---



(25)
NOUN   NOUN   VERB   AUX
Sene-ye  tez-i  bitir-miş  ol-acağ-ım
year-DAT tez-ACC finish-PTCP OL-FUT-1SG
'I will have finished the thesis by next year.'

## 5.6 *ol-* as light verb

Finally, Turkish employs *ol-* in light verb constructions. In verbs like *tatmin ol-* (to be satisfied), *rahatsız ol-* (to be bothered), *ol-* is considered as a light verb. Such verbs usually have an active counterpart with *et-*: *tatmin et-* (to satisfy) and *rahatsız et-* (to bother). Just like its *et-* counterpart it is annotated via the dependency compound:lvc (lvc stands for light verb construction) and UPOS VERB. We should inform that this particular variant of *ol-* has already been differentiated in previous Turkish treebanks and is included here for the sake of completeness.

## 6 Conclusion

Having a highly rich morphology, Turkish poses a great deal of challenges for linguists regarding representation in a dependency syntax framework. Some of these challenges come from the rich morphology and some from the unexpected behavior of derivation in Turkish. In this paper, we laid out some problems with respect to derivation and lexical ambiguities in the language and their adaptation to the UD framework. Especially in the derivation part, we feel that some significant amount of work awaits us and other linguists who aim to enhance annotation of Turkish in a syntactic framework based on dependency relations. We expect this study to contribute towards a more precise annotation scheme for Turkish and other agglutinative languages, which in turn may also lead to the training of more accurate NLP tools including lemmatizers and dependency parsers.

## 7 Acknowledgement

---

[13]*var* in (22) and *olduğunu* in (24) also have a significant XPOS that we name Exist which stands for *existential*. We annotate this tag to existential predicates.

# References

Yesim Aksan, Mustafa Aksan, Ahmet Koltuksuz, Taner Sezer, Ümit Mersinli, Umut Ufuk Demirhan, Hakan Yilmazer, Gülsüm Atasoy, Seda Öz, Ipek Yildiz, et al. 2012. Construction of the Turkish national corpus (tnc). In *LREC*, pages 3223–3227.

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*.

Çağrı Çöltekin. 2010. A freely available morphological analyzer for Turkish. In Proceedings of the Seventh International Conference on Language Resources and Evaluation (LREC'10), Valletta, Malta. European Language Resources Association (ELRA).

Çağrı Çöltekin. 2015. A grammar-book treebank of Turkish. In *Proceedings of the 14th Workshop on Treebanks and Linguistic Theories (TLT 14)*, pages 35–49.

Marie-Catherine De Marneffe, Timothy Dozat, Natalia Silveira, Katri Haverinen, Filip Ginter, Joakim Nivre, and Christopher D Manning. 2014. Universal Stanford dependencies: A cross-linguistic typology. In *LREC*, volume 14, pages 4585–4592.

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 and Christopher D Manning. 2008. The Stanford typed dependencies representation. In *Coling 2008: proceedings of the workshop on cross-framework and cross-domain parser evaluation*, pages 1–8.

Aslı Göksel. 2001. The auxiliary verb ol at the morphology-syntax interface. *The Verb in Turkish. Amsterdam: John Benjamins*, pages 151–181.

Aslı Göksel and Celia Kerslake. 2005. Turkish: A Comprehensive Grammar. Comprehensive grammars. Routledge.

Jorge Hankamer. 2004. An ad-phrasal affix in Turkish. *MIT Working Papers in Linguistics*, 46:289–299.

Jorge Hankamer. 2005. Why there are two –ki's in Turkish. *Ms., USC*.

Jenna Kanerva, Filip Ginter, Niko Miekka, Akseli Leino, and Tapio Salakoski. 2018. Turku neural parser pipeline: An end-to-end system for the CoNLL 2018 shared task. In *Proceedings of the CoNLL 2018 Shared Task: Multilingual parsing from raw text to universal dependencies*, pages 133–142.

Aysel Kapan. 2019. *Derivational networks of nouns and adjectives in Turkish*. Master's thesis, Boğaziçi University, İstanbul, Turkey.

Meltem Kelepir. 2001. *Topics in Turkish syntax: Clausal structure and scope*. Ph.D. thesis, Massachusetts Institute of Technology.

Jaklin Kornfilt. 1996. On copular clitic forms in Turkish. *ZAS Working Papers*.

Geoffrey Leech and Roger Garside. 1991. Running a grammar factory: The production of syntactically analysed corpora or treebanks. English Computer Corpora: Selected Papers and Research Guide, pages 15–32.

Mitchell Marcus, Beatrice Santorini, and Mary Ann Marcinkiewicz. 1993. Building a large annotated corpus of english: The penn treebank.

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

Balkız Öztürk and Eser Erguvanlı Taylan. 2018. Türkçede iyelik yapıları ve geçişli adlar. In *Dilbilimde Güncel Tartışmalar*. Ankara: Dilbilim Dernegi Yayınları.

Slav Petrov, Dipanjan Das, and Ryan McDonald. 2011. A universal part-of-speech tagset. *arXiv preprint arXiv:1104.2086*.

Annette Rios, Anne Göhring, and Martin Volk. 2012. Parallel treebanking spanish-quechua: how and how well do they align? *Linguistic Issues in Language Technology*, (7/13):online.

Umut Sulubacak, Gülşen Eryiğit, Tuğba Pamay, et al. 2016a. Imst: A revisited Turkish dependency treebank. In *Proceedings of TurCLing 2016, the 1st International Conference on Turkic Computational Linguistics*. Ege University Press.

Umut Sulubacak, Memduh Gökırmak, Francis Tyers, Çağrı Çöltekin, Joakim Nivre, and Gülşen Eryiğit. 2016b. Universal dependencies for Turkish. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 3444–3454.

Utku Türk, Furkan Atmaca, Şaziye Betül Özateş, Gözde Berk, Seyyit Talha Bedir, Abdullatif Köksal, Balkız Öztürk, Tunga Güngör, and Arzucan Özgür. 2020. Resources for Turkish dependency parsing: Introducing the boun treebank and the boat annotation tool. *arXiv preprint arXiv:2002.10416*.

Utku Türk, Furkan Atmaca, Şaziye Betül Özateş, Abdullatif Köksal, Balkiz Öztürk, Tunga Güngör, and Arzucan Özgür. 2019a. Turkish treebanking: Unifying and constructing efforts. In Proceedings of the 13th Linguistic Annotation Workshop, pages 166–177, Florence, Italy. Association for Computational Linguistics.

Utku Türk, Furkan Atmaca, Şaziye Betül Özateş, Balkız Öztürk, Tunga Güngör, and Arzucan Özgür. 2019b. Improving the annotations in the Turkish universal dependency treebank. In *Proceedings of the Third Workshop on Universal Dependencies (UDW, SyntaxFest 2019)*, pages 108–115.

Daniel Zeman. 2008. Reusable tagset conversion using tagset drivers. In *LREC*, volume 2008, pages 28–30.

Daniel Zeman, Martin Popel, Milan Straka, Jan Hajic, Joakim Nivre, Filip Ginter, Juhani Luotolahti, Sampo Pyysalo, and Slav Petrov. 2017. CoNLL 2017 shared task: Multilingual parsing from raw text to Universal Dependencies. In Proceedings of the CoNLL 2017 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies, pages 1–19, Vancouver, Canada. Association for Computational Linguistics.

# 8 Appendix

## A The derivative suffixes that are annotated using MISC column

| *Derivative suffixes* | examples | on MISC slot |
|---|---|---|
| *-lI* | *tatlı* (sweet) | df=tat (taste) |
| *-sIz* | *hissiz* (emotion-less, indifferent) | df=his (emotion) |
| *-lA* | *pencerele-* (to window) | df=pencere (window) |
| *-lAn (1)* | *hızlan-* (to accelerate) | df=hız (speed) |
| *-lAn (2)* | *gizlen-* (to disguise, to lie low) | df=gizle (to hide stg) |
| *-lAş* | *aptallaş-* (to be stupified) | df=aptal (stupid) |
| *-CA* | *aptalca* (foolish), bence (to me, in my opinion) | df=aptal (stupid), df=ben (I) |
| *-CI* | *simitçi* (bagel seller) | df=simit (bagel) |
| *-(y)IcI* | *koşucu* (runner) | df=koş (to run) |
| *-sAl* | *dönemsel* (periodical) | df=dönem (period) |
| *-CIk* | *gemicik* (ship *diminutive*) | df=gemi (ship) |
| *-mA* | *araştırma* (research) | df=araştır (to research) |
| *-(y)Iş* | *çıkış* (exit) | df=çık (to exit) |
| *-lIk* | *özellik* (attribute, specialty) | df=özel (special) |
| the enforcement prefix | *masmavi* (deep blue, rich blue) | df=mavi (blue) |