IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received 18 July 2022, accepted 21 August 2022, date of publication 29 August 2022, date of current version 13 September 2022.

Digital Object Identifier 10.1109/ACCESS.2022.3202947

RESEARCH ARTICLE

A Hybrid Deep Dependency Parsing Approach Enhanced With Rules and Morphology: A Case Study for Turkish

ŞAZİYE BETÜL ÖZATEŞ[1], ARZUCAN ÖZGÜR[1], TUNGA GÜNGÖR[1], AND BALKIZ ÖZTÜRK BAŞARAN[2]

[1]Department of Computer Engineering, Boğaziçi University, Bebek, 34342 İstanbul, Turkey
[2]Department of Linguistics, Boğaziçi University, Bebek, 34342 İstanbul, Turkey

Corresponding author: Arzucan Özgür (arzucan.ozgur@boun.edu.tr)

This work was supported in part by the Turkish Directorate of Strategy and Budget under the TAM Project under Grant 2007K12-873, and in part by the Scientific and Technological Research Council of Turkey (TÜBİTAK) under Grant 117E971. The work of Şaziye Betül Özateş was supported by TÜBİTAK under Grant BİDEB 2211.

ABSTRACT Fully data-driven, deep learning-based models are usually designed as language-independent and have been shown to be successful for many natural language processing tasks. However, when the studied language is not high-resource and the amount of training data is insufficient, these models can benefit from the integration of natural language grammar-based information. We propose two approaches to dependency parsing especially for languages with restricted amount of training data. Our first approach combines a state-of-the-art deep learning-based parser with a rule-based approach and the second one incorporates morphological information into the parser. In the rule-based approach, the parsing decisions made by the rules are encoded and concatenated with the vector representations of the input words as additional information to the deep network. The morphology-based approach proposes different methods to include the morphological structure of words into the parser network. Experiments are conducted on three different Turkish treebanks and the results suggest that integration of explicit knowledge about the target language to a neural parser through a rule-based parsing system and morphological analysis leads to more accurate annotations and hence, increases the parsing performance in terms of attachment scores. The proposed methods are developed for Turkish, but can be adapted to other languages as well.

INDEX TERMS Dependency parsing, computational linguistics, recurrent neural networks.

I. INTRODUCTION

Current state-of-the-art dependency parsers usually rely solely on deep learning methods, where parsers try to learn the characteristics of the language from available training data [1], [2]. As expected, this approach works well when the training data size is big enough. However, these pure deep learning-based approaches cannot reach the desired success levels when the data size is insufficient [3]. It was observed that deep learning-based systems need large amounts of data to be able to reach high performance [4]. For languages with small data sets, there is a need for developing additional methods that meet the characteristic needs of these languages.

In this article, we propose to take into account the language grammar and integrate the information extracted from the grammar to a deep learning-based dependency parser. We propose two approaches for the inclusion of the grammar to the neural parser model. Our first approach is to integrate linguistically-oriented rules to a deep learning-based parser for dependency parsing of languages especially with restricted amount of training data. The rules are created to deal with the problematic parts in the sentences that are hard to predict. In our second approach, we give morpheme information as an additional input source to the parsing system.

The associate editor coordinating the review of this manuscript and approving it for publication was Ángel F. García-Fernández.

We experimented with different methods for inclusion of the morpheme information. We applied the proposed methods to Turkish and the experimental results suggest that both approaches improve the parsing performance of a state-of-the-art dependency parser for Turkish.

The proposed methods were evaluated on both projective and nonprojective sentences and currently hold the state-of-the-art performance in parsing the Turkish IMST-UD Treebank. To the best of our knowledge, this is the first study that integrates the parsing actions of a rule-based method and morphological elements into a deep learning-based dependency parsing system and may serve as a base for other low- or mid-resource languages.

The main contributions of this article are as follows:

- A novel rule-based enhancement method that can be integrated to any neural dependency parser.
- A morphology-based enhancement method with three different ways of including morphological information to the parser.
- A simple yet useful integration method that allows to combine the proposed enhancement methods with any neural dependency parser.
- State-of-the-art dependency parsing scores on the IMST-UD Treebank.

The rest of this article is organized as follows. Section II presents the related work on deep learning-based, rule-based, and morphology-based approaches to dependency parsing as well as previous studies for parsing of Turkish. In Section III, we describe our proposed models to dependency parsing that combine hand-crafted rules and the morphological information with a state-of-the-art deep learning-based dependency parser. Section IV gives the experiment details and results as well as a discussion on how the proposed models can be adapted to other languages. Finally, Section V concludes the article and suggests some future work.

II. RELATED WORK

Purely rule-based approaches to natural language processing (NLP) problems have been very popular in the past, from part of speech tagging [5] to aspect extraction in sentiment analysis [6]. Rule-based methods have also been applied to dependency parsing. There have been studies on rule-based parsing using grammar rules for Turkish [7] and for other languages [8], [9], [10], [11].

Recently, deep learning methods began to be frequently applied to dependency parsing and show promising performances in predicting the dependency parses of sentences [1], [2], [12]. In 2017, a state-of-the-art LSTM-based dependency parser [13] achieved the best performance in 54 treebanks including the IMST-UD Treebank at the CoNLL'17 Shared Task on Multilingual Parsing from Raw Text to Universal Dependencies [14]. This parser together with its enhanced versions [15], [16] presented at the CoNLL'18 Shared Task on Multilingual Parsing from Raw Text to Universal Dependencies [17] show state-of-the-art performance on the dependency parsing of many languages. However, these parsers do not have language specific features that can boost the parsing performance, especially for morphologically rich and under-resourced languages like Turkish.

Morphologically rich languages (MRLs) pose problems when state-of-the-art NLP models developed for the most widely studied languages like English and French are applied directly to them [18]. There are studies that include rule-based knowledge to data-driven parsers in order to increase parsing accuracy. Reference [19] experimented with different voting mechanisms to combine seven different dependency parsers including a rule-based parser. Another study applied a rule-based mechanism on the output of a dependency parser to create collapsed dependencies [20].

There have also been several approaches that use morphological information in the dependency parsing of the MRLs. Similar to [21] and [22], which utilize morphological features for the dependency parsing of Hindi and Hebrew respectively, [23] measured the effects of nine morphological features extracted from an Arabic morphological analysis and disambiguation toolkit [24] on the parsing performance of the MaltParser [25] for Arabic. These studies show that usage of some morphological features works well for the dependency parsing of MRLs. Reference [26] compared the strength of character-level modelling of words with an oracle model which has explicit access to morphological analysis of words on dependency parsing and observed that combining words with their corresponding morpheme information using a bi-LSTM structure in the word representation layer outperforms the character-based word representation models. Reference [27] proposed two morphology-based approaches to dependency parsing. In their first approach, they combined the vector representation of words with the representation of some of the morphological attributes given in treebanks. Their second approach represents the words by separating them to their corresponding lemma and suffixes for suitable languages. They observed that both of the models improve the parsing accuracy for agglutinative languages. Reference [28] proposed a multitask learning framework that makes use of language philogenetic trees to represent the shared information among the languages. They used gold morphological features for dependency parsing by summing the created vectors of each morphological attribute given in the treebanks and add this vector to the representation of the word, similarly to [27].

However, to the best of our knowledge, there does not exist any prior research on a hybrid approach for dependency parsing, where parsing decisions of hand-crafted rules together with morphological information are integrated into a deep learning-based dependency parsing model. Our inclusion of morphology also differs from the previous works in terms of extracting the morphological information. Instead of using morphological features of a word, our models utilize its suffixes explicitly. While two of the proposed morphology-based methods include the suffixes of each word to the word representation model directly, the third one represents each word

with the suffixes which can and cannot bind to the root of that word.

A. TURKISH DEPENDENCY PARSING

In this study, we propose both rule-based and morphology-based enhancement methods for dependency parsing and integrate our methods to a state-of-the-art dependency parser [13]. We applied the proposed approaches to Turkish. Because, unlike English, the resources for Turkish NLP in general are restricted, which makes it suitable for such enhancements. For dependency parsing, the English treebanks have a total of 34,631 sentences[1] annotated in the Universal Dependencies (UD) style [29], with the largest one, the EWT Treebank [30] including 16,622 sentences. On the other hand, for Turkish, until very recently the only data sets used for training and evaluation of the systems were the IMST-UD Treebank [31] which consists of 5,635 annotated sentences and the Turkish UD Parallel (PUD) Treebank [14] of 1,000 annotated sentences that is used for testing purposes. There are also IWT-UD Treebank [32], which includes 5,009 sentences crawled from the web and the Turkish UD GB Treebank [33] of 2,880 entries which was created by annotating the examples given in [34]. However these treebanks differ from the others in terms of the source of sentences. The IWT-UD Treebank was created from social media texts and has a noncanonical language that is completely different from the other well-edited treebanks [32]. The source of the GB Treebank is not a naturally generated corpus but examples in a grammar book which include lots of incomplete sentences and sentences with informal usage.

Turkish is not a well-studied language in natural language processing, and dependency parsing is no exception to this. Following the initial work in [7], another study presents a word-based and two inflectional group-based input representation models for dependency parsing of Turkish [35] which use a version of backward beam search to parse the sentences. They used a subset of 3,398 sentences of the Turkish Dependency Treebank [36] with only projective (noncrossing) dependency relations to train and test the proposed parsers. Later, a data-driven dependency parser for Turkish was proposed, which relies completely on inductive learning from treebank data for the analysis of new sentences, and on deterministic parsing for disambiguation [37]. The authors use a variant of the transition-based parsing system MaltParser proposed in [25], a linear-time, deterministic, classifier-based parsing method with history-based feature models and discriminative learning. However, the definition of a well-formed dependency tree for MaltParser is different from the conventions of the UD scheme such that the artificial root node in a dependency tree may have more than one child in the output of the MaltParser. Yet, UD restricts the artificial root node to have exactly one child and it is not possible to have MaltParser produce dependency trees that follow the UD convention.

[1] In UD version 2.3. Available at http://hdl.handle.net/11234/1-2895

Reference [38] extracted different multiword expression classes as a pre-processing step for a statistical dependency parser. Only the projective dependencies are considered. Reference [39] and [40] are other notable studies that are based on optimized versions of the MaltParser system. Later, a graph-based approach was proposed in [41], where a discriminative linear model is trained and a lattice dependency parser is created that uses dual decomposition. All these studies on Turkish dependency parsing used the first version [36] of the Turkish Treebank annotated in non-UD (non-Universal Dependencies) style.

To eliminate the inconsistencies in this treebank and to obtain better performance, a revised version of it was presented under the name of IMST Treebank [42] and this new version was evaluated again using the MaltParser. However, the IMST Treebank was also in non-UD style. To contribute to the unifying efforts of the UD project, the IMST Treebank was converted automatically to the UD annotation style [31] and was named as IMST-UD Treebank. This most up-to-date version of the Turkish treebank was evaluated using MaltParser in [32], however only a subset of the treebank was used by eliminating the nonprojective dependencies which allow crossing edges in a dependency tree in training and development sets. In our study, we included both projective and nonprojective dependencies in IMST-UD as the proportion of nonprojective sentences in Turkish is too high to be ignored [39].

III. THE PROPOSED METHODOLOGY

In order to improve the parsing performance of deep learning-based parsers, we design hybrid methods where the grammar-based information is fed into the deep network of a data-driven parser. We propose two different approaches for supplying the information extracted from the language grammar, the first one is via hand-crafted grammar rules for detecting dependency relations between words and the second one is by analyzing the underlying morphological structure of words.

We first give a brief description of the state-of-the-art neural parser used in this study in Section III-A. We then explain our rule-based parsing method in Section III-B and show how these two methods are integrated to get a better parsing mechanism in Section III-C. Finally, we describe our morphology-based enhancement method and its integration to the parser in Section III-D.

A. STANFORD's NEURAL DEPENDENCY PARSER

Stanford's graph-based neural dependency parser [13] is the leading system in the CoNLL'17 Shared Task on UD Parsing [14]. For the representation of input words, the model sums learned word embeddings, pre-trained word embeddings, and character embeddings and then concatenate the resulting word vector with their corresponding part-of-speech (POS) tag embeddings. The parser includes three BiLSTM layers with 100-dimensional word and tag embeddings. It uses two biaffine classifiers: the arc classifier takes

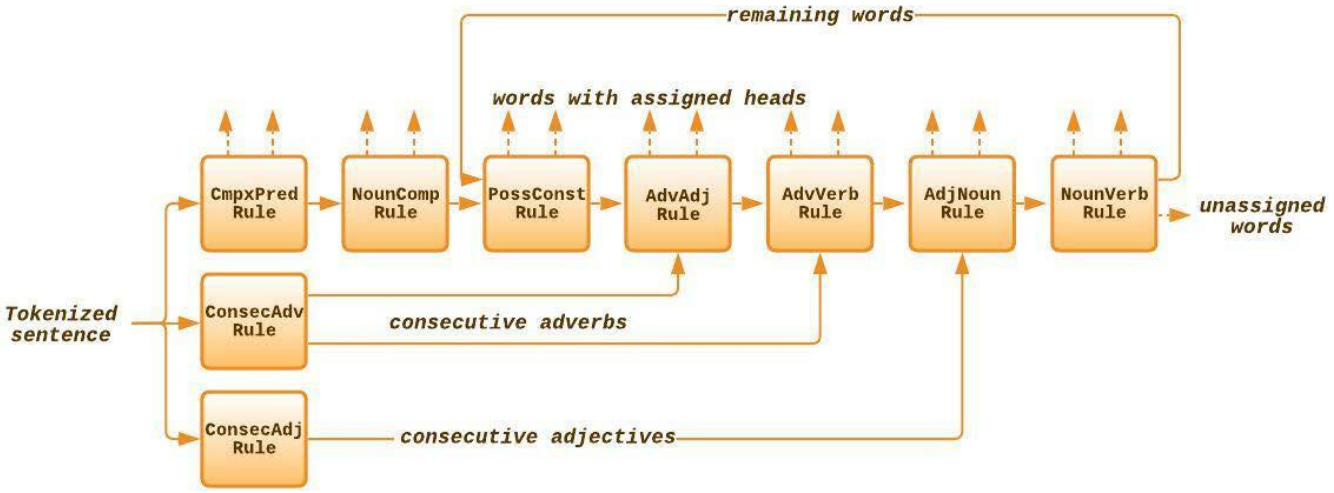


FIGURE 1. Our rule-based dependency parser.

400-dimensional head-dependent vectors produced by ReLU layers on top of the final BiLSTM layer to decide the head of a given token, and the label classifier uses 100-dimensional vectors to decide its label. For more information, see [13]. In this study, we use this parser as the baseline system.

B. A RULE-BASED UNLABELED PARSING APPROACH

We aim at enhancing the baseline parser by supplying linguistic information to the neural model. For this purpose, we design linguistically oriented rules for the dependency parsing of the Turkish language. These hand-crafted rules that are not completely free from false positives determine the head of the words in a sentence without assigning a label for the created dependency relation. Rather than applying them as a post-processing step[2] that directly modifies the predictions made by the baseline parser, we integrate these rules to the parser via the dense representation of words in order to make them have an implicit effect on the decision of the parser (see Section III-C).

Instead of a complete parser that creates a fully connected dependency graph, our rule-based system deals with the most difficult cases in predicting dependency relations in a sentence according to the parsing errors of the baseline system, such as complex predicates or multiple adverbs. The rules are created by considering the structural components of a sentence. We consider the relations between verbs, nouns, adverbs, and adjectives in a sentence as having the main importance and generate rules that deal with these relations. The rules are based on the existing grammar rules extracted from [34].

Fig. 1 shows the general mechanism of our rule-based parsing system. Our model takes a tokenized sentence as its input. Then the rules are applied in sequential order, considering the grammatical structure of Turkish. First, the ConsecAdv rule creates a list of the consecutive adverbs that are related to each other in the sentence, and sends this list to the AdvAdj and AdvVerb rules as an input. Similarly, the ConsecAdj rule creates a list of the consecutive adjectives in the sentence that are related to each other and sends this list to the AdjNoun rule. After that, the CmpxPred and NounComp rules are applied to the words of the sentence sequentially. Then, the PossConst, AdvAdj, AdvVerb, AdjNoun, and NounVerb rules are applied to the remaining words in an iterative manner. This process continues until the heads of all the words in the sentence are associated with a dependency relation or no more dependency relations can be found in the sentence. The following subsections explain each rule in detail.

1) COMPLEX PREDICATES AND VERBAL IDIOMS (CmpxPred) RULE

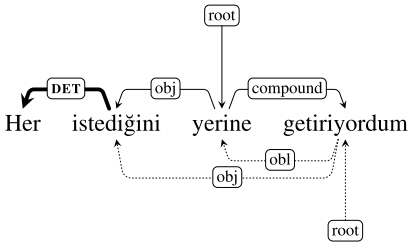
This rule processes the complex predicates (e.g., *kabul et (accept)*, *sebep ol (cause)* etc.) and verbal idioms (e.g., *göz yum (condone)* etc.) in a sentence. Complex predicates in Turkish are made up of a bare nominal followed by one of the free light verbs *ol, et, yap, gel, dur, kal, çık, düş, buyur, eyle* [34, p. 143]. However, verbal idioms can have verbs in a wide range of words and the meaning of these verbs are changed when they are used in an idiom.

In complex predicates, head is the nominal part of the predicate. This is because light verb constructions in Turkish are not fully in parallel to their counterparts in languages like Persian where the light verb is considered as the source of all syntactic properties. In Turkish, the nominal part of the noun can still retain its argument structure and case-frame even within the absence of a light predicate as well observed in the literature [43], [44], [45], [46]. Since the nouns not only make a semantic contribution but also determine the case and argument structure properties of the complex predicate, we take the nominal part of the complex predicate as the head of the construction. Yet, due to insufficient amount of training data, parsers usually fail to detect these multiword

[2]We tried this method in our preliminary experiments and observed that though it made a little improvement on the parsing performance, integrating rule decisions at the input representation step of the neural model shows superior performance.

predicates [47] and consider the verb of such predicates as the head of the relation. Example (1) shows such a false annotation done by the trained baseline deep learning-based parser. In the examples, annotations predicted falsely by the baseline parser are shown with dotted lines, their corrected forms are shown with fine lines. Thick lines represent the annotations predicted correctly by the baseline parser. This representation is followed in all of the examples in this article. Note that, all examples that include dotted lines (false annotations) are taken from the output of the baseline parser. The parser falsely predicts that the verb *getir (bring)* is the `root` word and *yerine (to its place)* is an `oblique` of the `root` word. In fact, *yerine getir (fulfill)* is a verbal idiom and the verb *getir* is the verbal component of the complex predicate.

(1)



Her iste-di-ği-ni yer-in-e
every want-pst-3sg.poss-acc place-poss-dat
getir-iyor-dum.
bring-prs-pst-1sg

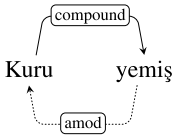
'I have been doing whatever he/she wants.'

This rule correctly constructs the dependency relations between the words of such predicates. In order to detect such verbal compounds however, it needs a dictionary that lists complex predicates and idioms in Turkish. We collected approximately 8K complex predicates using the Turkish Proverbs and Idioms Dictionary supplied by the Turkish Language Association (TDK) [48] and from various online Turkish resources.[3] The CmpxPred rule searches for complex predicates and idioms in a sentence. When such a predicate is found, the second word of the predicate is set as a dependent of the first word and given `cmp` as the rule-encoding. Since the head of the second word is found, it is eliminated from the remaining words list.

2) NOUN COMPOUNDS (NounComp) RULE

In Turkish, there are two types of noun compounds: bare compounds that are treated as head-level (X^0) constructions and $-(s)I(n)$ compounds where the first noun has no suffixes whereas the second noun in the compound takes the 3rd person possessive suffix $-(s)I(n)$ [34, p. 94]. In terms of dependency grammar representations, in (X^0) constructions the head word of the compound is the first noun, whereas in $-(s)I(n)$ compounds the first noun is dependent on the second noun. Differentiating between these two noun compound types is not easy for parsers in the absence of large amount of training data. Examples (2) and (3) show an (X^0) compound *kuru yemiş (dried nuts)* and a $-(s)I(n)$ compound *ev yemeği (homemade food)*, respectively. In Example (2), both of the words are in their bare forms with no suffixes and form a noun compound with the head word being *kuru (dried)*. Yet, the parser falsely predicts *kuru* as an adjective modifier of *yemiş (nuts)*.

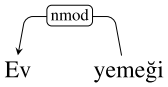
(2)



kuru yemiş
dry nut

'Dried nuts'

(3)

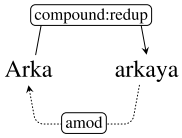


ev yemeği
home food-3sg-poss

'Homemade food'

Reduplicated compounds are also handled by this rule. Reduplication is a common process in Turkish [49]. Reduplicated words construct reduplicated compounds. However, the parser sometimes cannot recognize this idiomatic structure and fails to construct the compound. Example (4) shows this kind of confusion where the parser falsely assigns the first word of the compound as an adjective modifier of the second.

(4)



arka arka-ya
back back-DAT

'Repeatedly'

To detect these cases, the NounComp rule utilizes large lexicons and detects the noun compounds in sentences with the help of these lexicons. We extracted three different lexicons from the official noun compounds dictionary of Turkish language published by the Turkish Language Association. These three lexicons are for noun compounds, possessive compounds, and reduplicated compounds with, respectively, the sizes of approximately 1.5K, 7K, and 2K entries. We classified each entry in the dictionary as one of the three kinds of compounds according to their lexical classes. The NounComp rule searches through these lexicons and by this way detects noun compounds, possessive compounds, and reduplicated compounds in the sentences. In noun compounds and reduplicated compounds, the second word is set as a dependent of the first word, whereas in possessive compounds, the first word is set as a dependent of the second word. As rule-encodings, `com` is given to the dependent components of detected noun compounds, `fla` is given to the dependants of

[3]All lexicons collected for this study can be found at https://github.com/sb-b/BOUN-PARS/tree/master/rule-based-model.

reduplicated compounds, and nmo is given to the dependent words in possessive compounds. The words whose heads are found are then eliminated from the remaining words list.

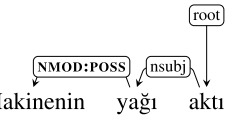
3) POSSESSIVE CONSTRUCTION (PossConst) RULE

The PossConst rule includes both *genitive possessive constructions* and *possessive compounds* that cannot be detected by the NounComp rule. In genitive possessive constructions, the first noun of a noun phrase takes a genitive suffix *−(n)In* that represents the ownership of the first noun over the second noun. The second noun takes a possessive suffix *−(s)I(n)* [34, p. 161]. In possessive compounds, there is no genitive suffix and the first noun in the compound appears without any case marking [34, p. 96]. Although it is easy for the parser to detect the genitive possessive construction relations due to the existence of a genitive suffix, detecting possessive compounds is a challenging task. Because in possessive compounds, there does not exist a genitive suffix in the first noun and the possessive suffix *−(s)I(n)* in the second noun is confusing since it appears the same as the accusative suffix *−(y)I* when the suffix initial *(s)* is dropped in nouns ending with a consonant.

This situation is depicted in Examples (5), (6) and (7). The only difference in Examples (5) and (6) is that the genitive suffix showing the possession exists in (5) and is omitted in (6). In both sentences, the subject of the sentence is the word *yağ (oil)* and the two nouns form a possessive construction. However, this is not the case in Example (7). Here, the subject of the sentence is the word *makine (machine)*, and the word *yağ (oil)* is the object of the verb *akıt- (drip)*. The confusion originates from the use of the same consecutive nouns, *makine yağı*, in both of the example sentences (6) and (7). However, the *-ı* suffix of the word *yağ* in (7) is actually an accusative suffix and hence the two nouns in (7) do not form a compound. In order to help the parser to differentiate between these two cases in sentences, we construct the PossConst rule that identifies whether there is a compound relation between two consecutive nouns or not.

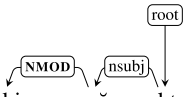
When two consecutive nouns are detected, the rule checks whether there is a genitive suffix in the first noun. If yes, it is set as a dependent of the second noun, given nmo as the rule-encoding, and dropped from the remaining words list. If the first noun is in bare form and the second noun has a possessive suffix, then the first noun is set as the dependent on the second noun. As stated, the third person possessive suffix *-(s)I(n)* can be confused with the accusative suffix *-(y)I* when they are attached to a word ending in a consonant. In this case, both suffixes reduce to the form *ı, i, u,* or *ü*. To prevent this confusion, the rule analyzes the morphological features of the word and checks if it is identified as accusative (Example (7)) or not (Example (6)). If it is identified as the possessive suffix, the PossConst rule assigns the first noun as a dependent of the second noun, gives nmo rule-encoding, and the first noun is eliminated from the remaining words list.

In addition, the PossConst rule deals with multiword proper nouns and determiner-noun relations. When the PossConst rule detects a multiword proper noun, all the following consecutive proper nouns are set as dependent on the first proper noun, given cop as the rule-encoding, and dropped from the remaining words list. When it detects a noun that is preceded by a determiner, it sets the determiner as dependent on the noun. The dependent word is given det rule-encoding and dropped from the remaining words list.



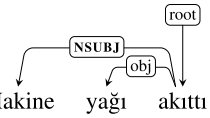
(5) Makinenin yağ-ı ak-tı.
machine-gen oil-3sg-poss leak-pst-3sg

'The machine's oil leaked.'



(6) Makine yağ-ı ak-tı.
machine oil-3sg-poss leak-pst-3sg

'Machine oil leaked.'



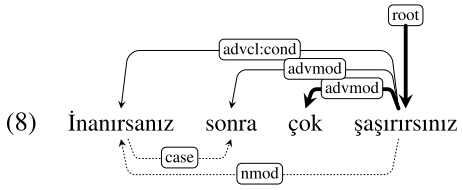
(7) Makine yağ-ı akıt-tı.
machine oil-acc drip-pst-3sg

'The machine dripped the oil.'

4) CONSECUTIVE ADVERB (ConsecAdv) RULE

Consecutive adverbs are also hard-to-detect with data-driven parsers. For instance, the first adverb *sonra (then)* and the second adverb *çok (very)* are both dependents of the verb *şaşırırsınız (be surprised)* in Example (8). However, the parser falsely predicts the word *sonra* as the dependent of the previous word *inanırsanız (if you believe)* with case label. The ConsecAdv rule handles such consecutive adverbs in a sentence. We observe that, if there are two consecutive adverbs in a sentence, usually there are two cases: either the first adverb is dependent on the second adverb or they are both dependent on the same head word. So, when two consecutive adverbs are found, the method checks whether the first adverb belongs to the group of adverbs that emphasize the meaning of the next adverb or not [34, p. 213]. This is done via searching through a list of adverbs of quantity or degree taken from [34, pp. 210-211]. If yes, the first adverb is set as a dependent word of the second adverb, given adv as the rule-encoding, and dropped from the remaining words list. If not, these two adverbs are put in a list (which will be called *the consecutive adverbs list* throughout the article) and the first

one is dropped from the list of remaining words. When the head word of the second adverb is found later, the first adverb is also bound to the same head word.

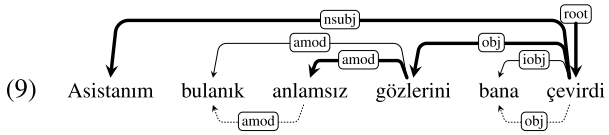


İnan-ır-sa-nız sonra çok şaşır-ır-sınız.
believe-aor-cond-2pl then very surprise-aor-2pl

'If you believe, then you will be very surprised.'

5) CONSECUTIVE ADJECTIVE (ConsecAdj) RULE

Consecutive adjectives are another troublesome word group which the parser sometimes fails to parse correctly. Example (9) shows such an annotation.



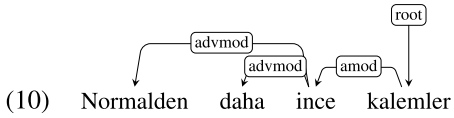
Asistanım bulanık anlamsız
assistant-1sg.poss blurred meaningless
göz-ler-in-i bana çevir-di.
eye-pl-3sg.poss-acc I-dat turn-pst-3sg

'My assistant turned his/her blurred meaningless eyes towards me.'

The parser falsely considers the word *bulanık (blurred)* as an adjectival modifier of the word *anlamsız* and assigns `amod` to *bulanık*. In fact, it is an adjective describing the word *gözler (eyes)* and should be an `amod` dependent of the word *gözler*. The ConsecAdj rule is created to prevent this type of errors. This rule finds all the consecutive adjectives in a sentence. Usually, two consecutive adjectives are dependent on the same word. So, when two consecutive adjectives are found, these adjectives are put into a list (which will be called *the consecutive adjectives list* from now on) and the first one is dropped from the list of remaining words. When the head word of the second adjective is found later by the parser, the first adjective is also set as a dependent of the same head word.

6) ADVERB-ADJECTIVE (AdvAdj) RULE

The AdvAdj rule handles adverb-adjective relations in a sentence. For every two consecutive words in the sentence, if the first word is an adverb and the second word is an adjective, and if the adverb is a quantity or degree adverb, then the adverb is set as a dependent of the adjective word [34, pp. 175-180] and given `adv` rule-encoding. When the head of an adverb is obtained in this way, the rule checks whether the adverb is in *the consecutive adverbs list* supplied by the ConsecAdv rule. If yes, the consecutive adverbs of this adverb are also set as dependents of the same head word and given `adv` as rule-encodings. So, in Example (10), when the AdvAdj rule detects the degree adverb *daha (more)* is followed by the adjective *ince (thin)*, it sets *daha* as a dependent of *ince*. The rule then checks the *consecutive adverbs list* previously created by the ConsecAdv rule and finds that the adverb *normalden (than normal)* is a consecutive adverb of the adverb *daha*. So, the adverb *normalden* is also set as a dependent of the adjective *ince*.

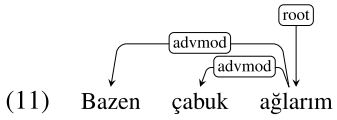


Normal-den daha ince kalem-ler
normal–abl more thin pencil-pl

'Pencils thinner than normal'

7) ADVERB-VERB (AdvVerb) RULE

For every two consecutive words in a sentence, if the first word is an adverb and the second word is a verb, and if the adverb is not one of *bile (even), -DAn önce (before something), -DAn sonra (after something)* that modify the preceeding word, then the AdvVerb rule sets the adverb as a dependent of the verb [34, p. 189] as in the relation between the adverb *çabuk (quickly)* and the verb *ağlarım (I cry)* in Example (11). Otherwise, it sets the previous word of the adverb as its head. As the head of an adverb is found, the AdvVerb rule checks whether the adverb is in *the consecutive adverbs list* supplied by the ConsecAdv rule. If yes, the consecutive adverbs of this adverb are also set as dependents of the same head word and given `adv` rule-encoding.

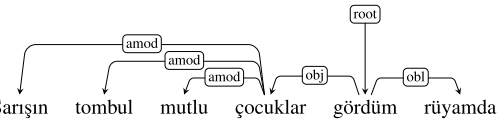


Bazen çabuk ağla-r-ım
sometime quick cry-aor-1sg

'Sometimes I cry quickly.'

8) ADJECTIVE-NOUN (AdjNoun) RULE

The AdjNoun rule constructs adjective-noun relations. For every two consecutive words in the sentence, if the first word is an adjective and the second word is a noun, then the adjective is set as a dependent word of the noun [34, p. 170] and `amo` is given as the rule-encoding. Like for the adverbs, when the head of an adjective is found, the algorithm checks whether the adjective is in *the consecutive adjectives list* supplied by the ConsecAdj rule. If yes, the consecutive adjectives of this adjective are also set as dependents of the same head word and given `amo` rule-encoding. So, in the case of Example (12), the three consecutive adjectives *sarışın (blonde)*, *tombul (chubby)*, and *mutlu (happy)* are all set as dependents of the noun *çocuklar (children)* by the AdjNoun rule.



(12) Sarışın tombul mutlu çocuklar gördüm rüyamda

Sarışın tombul mutlu çocuk-lar gör-dü-m
blonde chubby happy child-pl see-pst-1sg
rüya-m-da
dream-1sg-loc

'I saw blonde chubby happy children in my dream.'

9) NOUN-VERB (NounVerb) RULE

After complex predicates and noun, adverb, and adjective compounds are detected and eliminated from the sentence, the final NounVerb rule assigns any unassigned noun or pronoun followed by a verb as a dependent of that verb and gives `nov` rule-encoding.

A summary of the rules and their corresponding rule-encodings are depicted in Table 1. Fig. 2 depicts the application of each rule on an example sentence. In the example, the first rules that are applied to the sentence are the ConsecAdv and ConsecAdj rules. These rules prepare *the consecutive adverbs list* and *the consecutive adjectives list*, respectively. *The consecutive adverbs list* stores the consecutive adverbs that should be bound to the same head word. Similarly, *the consecutive adjectives list* stores the consecutive adjectives which should have the same head word.

After that, the CmpxPred and NounComp rules are applied consecutively. The CmpxPred rule finds the complex predicates and idioms in the sentence using a large lexicon whereas the NounComp rule detects the noun compounds, possessive compounds, and reduplicated compounds that also exist in the pre-built lexicons.

As the operations of these one-time rules are completed, the PossConst, AdvAdj, AdvVerb, AdjNoun, and NounVerb rules are applied to the sentence in a loop until none of the rules can be applied anymore. As for the example sentence in Fig. 2, only two words remained unassigned out of fifteen. The complete set of dependency relations extracted by the rule-based process is shown on the bottom of the figure. The last line in the figure shows the encoding of the rule applied to each word, which are then used by the dependency parser as will be explained in Section III-C.

C. INTEGRATING THE RULE-BASED APPROACH WITH STANFORD's GRAPH-BASED PARSING METHOD

Our approach for combining the rule-based method with Stanford's neural dependency parser is to embed the dependency parsing rule information to the dense representations of the words. The purpose of this approach is to give the parser an idea about finding the correct head of the corresponding word. The parser uses the dependent-head decisions made by the rule-based method in its learning phase and comes up with more accurate predictions about the syntactic annotation of sentences.

TABLE 1. Summary of rules in the rule-based system with examples for each encoding. Bold words in the *Example* column are dependents with an assigned rule-encoding. Note that, the second case of the ConsecAdv rule and the ConsecAdj rule do not assign a rule-encoding, but create lists to be used by other rules in subsequent steps.

Rule	Grammatical Structures	Rule encoding	Example
CmpxPred	Complex predicates	`cmp`	*kabul* → ***etmek*** *(to accept)*
NounComp	Noun compounds	`com`	*kara* → ***tahta*** *(blackboard)*
	Possessive compounds	`nmo`	***armut*** ← *sapı* *(pear stalk)*
	Reduplicated compounds	`fla`	*yamuk* → ***yumuk*** *(crooked)*
PossConst	Genetive possessive constructions	`nmo`	***armutun*** ← *sapı* *(stalk of pear)*
	Possessive compounds	`nmo`	***armut*** ← *sapı* *(pear stalk)*
	Determiner-noun relations	`det`	***bir*** ← *yol* *(a way)*
	Multiword expressions	`cop`	***zor*** ← *dur* *(it is hard)*
ConsecAdv	Consecutive adverbs:		
	1) adverb emphasizing adverb	`adv`	***cok*** ← *daha* *(much more)*
	2) adverbs sharing heads		*elbette böyle yaptım* *(course I did like this)* ***[elbette, böyle]***
ConsecAdj	Consecutive adjectives sharing heads		*küçük eski masa* *(small old table)* ***[küçük, eski]***
AdvAdj	Adverb-adjective relations	`adv`	***daha*** ← *uzun* *(longer)*
AdvVerb	Adverb-verb relations	`adv`	***hemen*** ← *geldi* *(came immediately)*
AdjNoun	Adjective-noun relations	`amo`	***uzun*** ← *ağaçlar* *(long trees)*
NounVerb	Noun-verb relations	`nov`	***eve*** ← *gitti* *(went to home)*

In our method, first the input sentences are pre-processed by the rule-based system. For each word in a sentence, the rules decide the head of the word if applicable, and then a three-letter encoding[4] that denotes the rule action applied (`cmp` for CmpxPred, `amo` for AdjNoun, etc.) is assigned to that word. During this process, the rule-encoded words are dropped from the remaining words list in the sentence and the rule-based system continues its process in a recursive manner until the head word of all words in the sentence are found or no more rule can be applied. Note that, each word is affected by at most one rule. The rules are applied sequentially and their order of application is defined considering the grammatical structure of Turkish. For instance, the noun compounds that are dealt with by the NounComp rule must be detected in the sentence before the PossConst rule that also deals with nouns because the components of noun compounds cannot be separated from each other. Similarly the AdvAdj rule that deals with adverbs followed by an adjective must be applied before the AdjNoun rule that handles adjectives followed by a noun because otherwise the AdjNoun rule would assign a rule-encoding to the adjective and it would be dropped from

[4]We also tried another representation where these three-letter rule-encodings are combined with the relative position of head for each word. However, this method gave similar performances with only providing the rule-encodings. So, we continued our experiments with the simpler approach of including only rule-encodings.

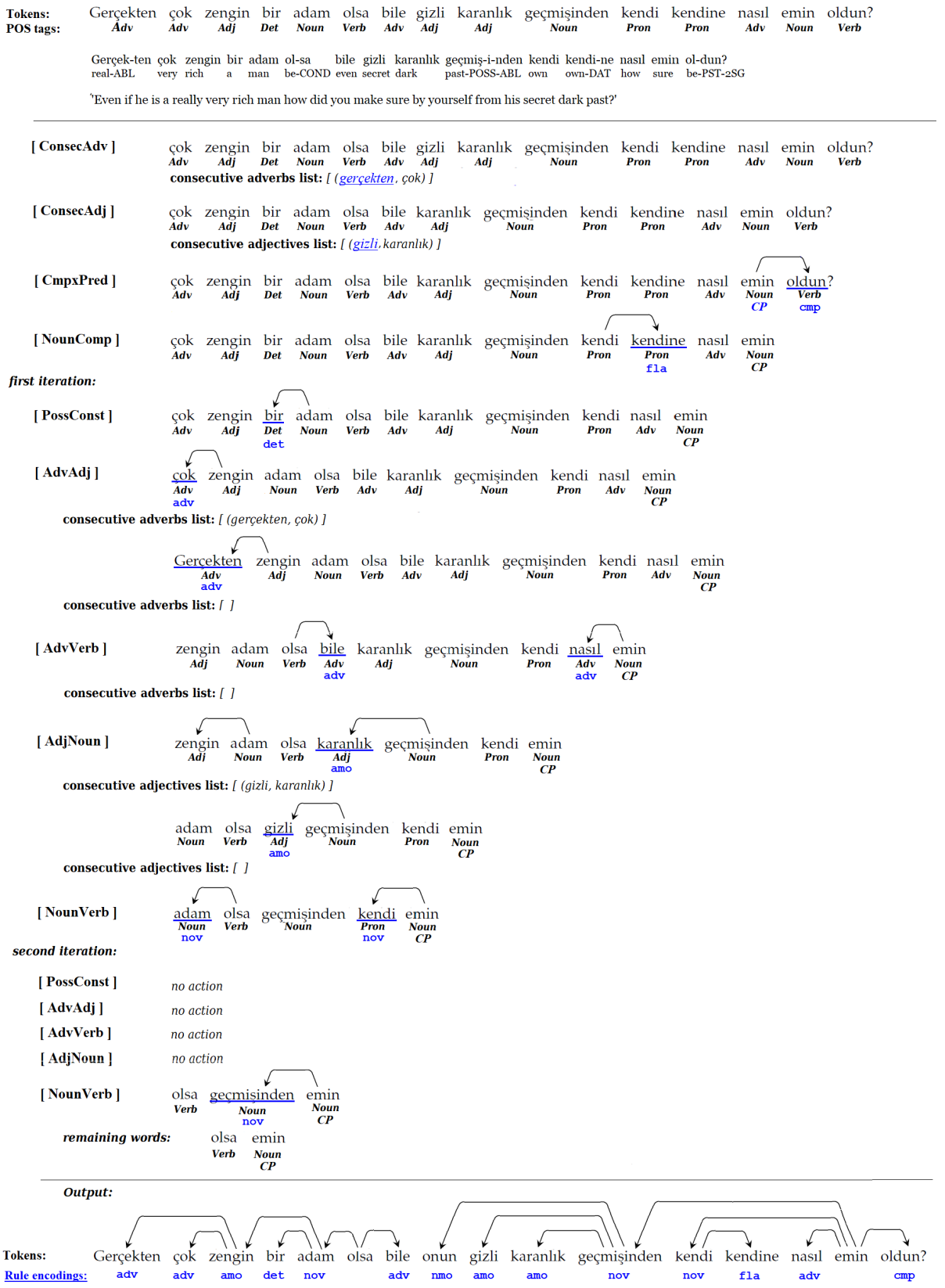


FIGURE 2. Operation of the rules on a sentence. Each rule is applied in accordance with Fig. 1. The resulting rule-encoding for each underlined word is shown with blue-colored three-letter encodings below that word. First ConsecAdv and ConsecAdj rules find consecutive adverbs and adjectives, respectively. Then CmpxPred and NounComp find complex predicates and noun compounds. The remaining rules (PossConst, AdvAdj, AdvVerb, AdjNoun, and NounVerb) are applied in an iterative manner until no rules can be applied anymore. Note that *emin* is marked as *CP* when it is identified as the head of a complex predicate to inform other rules that it is not a simple noun but the head of a complex predicate and hence acts like a verb in the sentence.

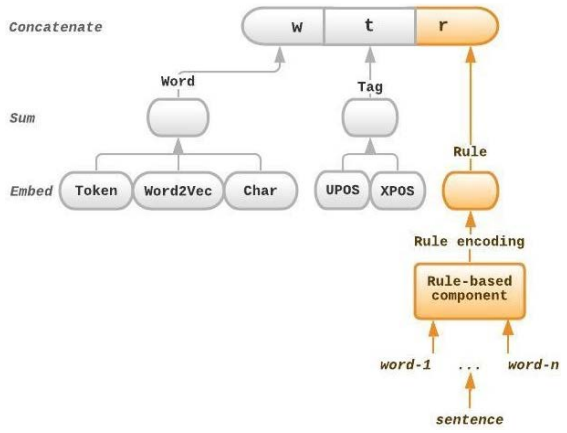


FIGURE 3. The word embedding representation of the hybrid model with the rule-based enhancement. As in the tag representation, rule-encodings are assigned randomly initialized embedding vectors at the beginning, which are then updated at each epoch during training.

the sentence and hence, the adverb-adjective relation in the sentence could not be detected. After this rule-based preprocessing step, the rule-encoding of each word is assigned a 100-dimensional embedding vector and concatenated to the embedding vector of that word. The rule embeddings are initialized randomly and trained together with the rest of the network. Fig. 3 depicts this scheme.

So, in our model, the rule vector representation is concatenated with the vector representation of [13]. The parser takes these word vector representations with the rule-based parsing decisions as input and learns the relations between the words in the training phase. In this way, we anticipate that the parser will benefit from the decisions of the rule-based system and arrive at a more accurate dependency tree for a given sentence.

D. A MORPHOLOGY-BASED ENHANCEMENT FOR DEPENDENCY PARSING

In addition to the rule-based enhancement to the deep learning-based parser, we also propose to include the morphological information directly to the system. In this approach, we use morphemes of a word as an additional source. Our motivation relies on the fact that Turkish is a highly agglutinative language where the word structure is described by identifying the different categories of suffixes and determining which stems the suffixes may attach to and their orders. The suffixation process in Turkish sometimes produces very long word forms that correspond to whole sentences in English [34]. The morphemes of a word hold important information in terms of the dependency relations that word belongs to. For instance, it was observed that the last inflectional morpheme of a word determines its role as a dependent in the sentence [36]. Based on such observations, we design three different methods to enhance the deep learning-based parser. The following subsections explain each model in detail.

1) THE INFLECTIONAL SUFFIXES MODEL

In this model, all of the inflectional suffixes are extracted from the morphologically analyzed form of the word, embedded, and then concatenated to the vector representation of that word. The integration method is the same as in Section III-C in the sense that inflectional suffixes of each input word are represented with a 100-dimensional randomly initialized embedding vector which is then concatenated with the word and tag embeddings of the same word. Fig. 4 depicts the creation of the dense representation of an example word *insanların (people's)*.

2) THE LAST SUFFIX MODEL

Slightly different than the Inflectional Suffixes Model, here the same process is performed for only the last suffix of an input word. The vector representation of the last derivational or inflectional suffix of a word is added to the vector representation of that word. Fig. 5 depicts the creation of the dense representation of the same example word in Fig. 4, but this time using the Last Suffix Model.

3) THE SUFFIX VECTOR MODEL

This model is a bit different than the previous two models. In the Suffix Vector Model, the input words are represented through a vector of all suffixes which the lemma of that word can and cannot take. The motivation behind this model comes from the idea that the role of a word form in a sentence can be determined by considering the suffixes that the lemma of that word never takes and the suffixes it frequently takes. For instance, inflectional suffixes in Turkish indicate how the constituents of a sentence are related to each other [34, p. 65]. For this purpose, we created a lemma-suffix matrix which consists of 40K unique lemmas and 81 inflectional and derivational suffixes in Turkish. The rows of the matrix list the lemmas and the columns show the normalized count of the times each lemma takes the corresponding suffix. To compute these statistics, we used the Newscor part of the Boun Web Corpus [50]. Newscor is created from news documents taken from three pioneering news portals in Turkish (Milliyet, Ntvmsnbc, and Radikal) and includes 184M words in Turkish. Morphological analyses of words in the corpus are predicted using the Turkish morphological analyzer and disambiguator tool [50]. A small subset of the lemma-suffix matrix is shown in Fig. 7 for demonstrational purposes.

This lemma-suffix matrix is then concatenated with the pre-trained word embedding matrix used by the parser. For each word entry in the pre-trained word embedding matrix, the row vector in the lemma-suffix matrix that corresponds to the lemma of that word is found and this vector is concatenated to the end of the embedding vector of that word. Fig. 6 depicts the word representation model of the system when we use the Suffix Vector Model.

IV. EXPERIMENTS

A. EXPERIMENTAL SETTINGS

We evaluated the Stanford's neural parser as the baseline system and the proposed hybrid parser with rule-based and morphology-based enhancement methods on the IMST-UD

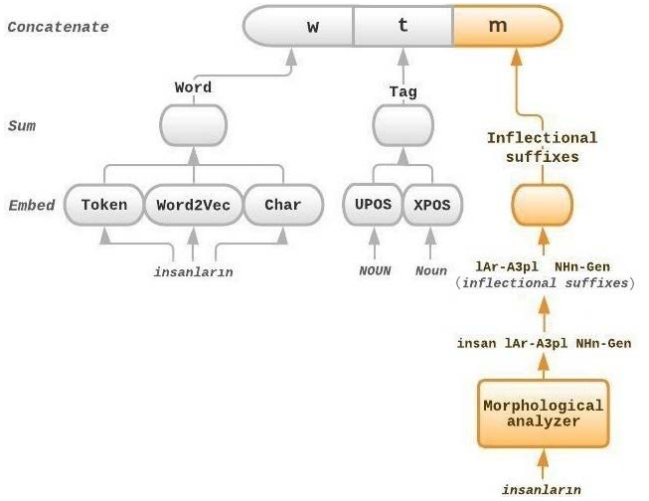


FIGURE 4. Dense representation of the word *insanların (people's)* using the Inflectional Suffixes Model.

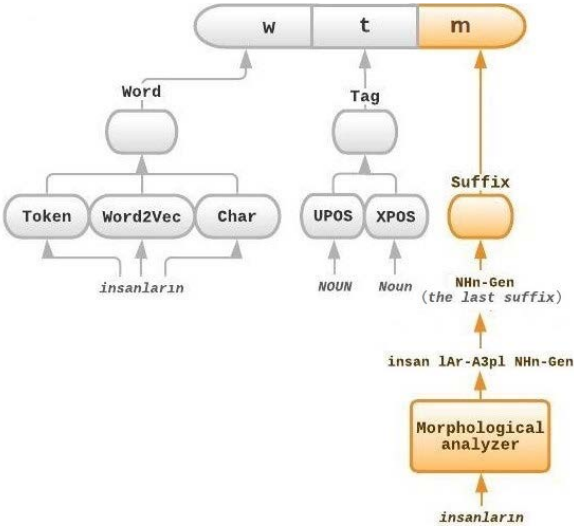


FIGURE 5. Dense representation of the word *insanların (people's)* using the Last Suffix Model.

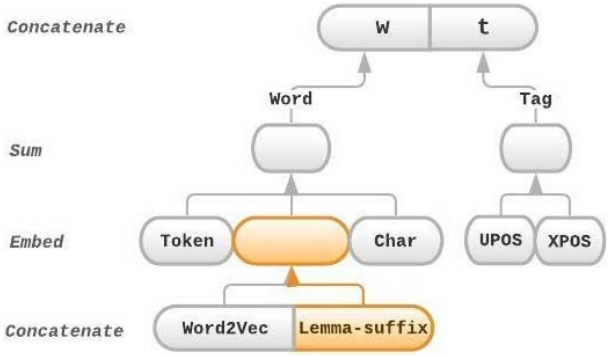


FIGURE 6. Word representation model of the system with the Suffix Vector Model.

Treebank,[5] the Turkish PUD Treebank [51], and the BOUN Treebank [52] which is a newly introduced treebank for Turkish. In all of the experiments, the default set of parameters are used for the deep network that produces the parse trees. We use 100-dimensional word vectors, POS tag vectors, and rule embedding vectors. The 3-layer BiLSTM modules of the parser have hidden layer size of 400 on each side. The arc MLP layer of the parser is 400-dimensional and the label MLP layer is 100-dimensional. All dropout probabilities are 0.33. We use Adam optimizer [53] with a learning rate of 0.002, training the models for a maximum of 30,000 iterations and with early stopping criterion as 5,000 iterations without improvement.

We evaluated each of the proposed models on the IMST-UD Treebank which is the most frequently used treebank in the literature. The training part of the IMST-UD Treebank has 3,685 annotated sentences and the development and test parts have 975 annotated sentences each. The PUD and BOUN treebanks were used in the second set of experiments where we measured the effect of increasing the size of the training data to the parsing models. We include both projective and nonprojective dependencies.

As in the baseline approach [13], we used 100-dimensional Turkish word vectors from the CoNLL-17 pre-trained word vectors [54]. In the evaluation of the dependency parsers, we used the word-based unlabeled attachment score (UAS) and the labeled attachment score (LAS) metrics, where UAS is measured as the percentage of words that are attached to the correct head, and LAS is defined as the percentage of words that are attached to the correct head with the correct dependency type.

In our system, both the rule-based and the morphology-based methods are dependent on a morphological analyzer and disambiguator tool. For this purpose, we used the Turkish morphological analyzer and disambiguator tool by [50]. This tool takes the whole sentence as input and analyzes and disambiguates the words with respect to their corresponding meanings in the sentence. This property is very useful for Turkish because there are many words in Turkish which have multiple morphological analyses that can be correctly disambiguated only by considering the context the word is in. The accuracy of the tool on a disambiguated Turkish corpus was reported as 96.45 per cent in [50].

Although our proposed enhancement methods use a morphological analyzer and disambiguator tool and do not rely on the gold lemmas, gold POS tags, or gold morphological features, the baseline parser used in this study needs input sentences in CoNLL-U format where the sentence is segmented to tokens and pre-processing operations such as lemmatization and tagging are supplied for each token. Since our main aim is to improve the performance of the parser, we supplied gold tokenization and POS tags to the parser when evaluating the proposed models and comparing their performance with the performance of the baseline model in order to observe the pure effect of the proposed methods on parsing. When we compared our best model with the state-of-the-art parsers on parsing raw text, we utilized the Turku Neural Parser Pipeline [15], an end-to-end system for parsing

[5]UD version 2.3.

	Ar	CA	CAsHna	CH	CHk	DA	DAn	DH	DHk	DHkçA	DHr	HI	Hm	HmHz	Hn	HnHz	Hr
bakan [Noun]	0	0.00003	0	0	0	0.00006	0.00219	0	0	0	0.00016	0	0.00079	0.00209	0.00002	0.00011	0
bakanlık [Noun]	0	0.00023	0	0	0	0.00129	0.00144	0	0	0	0.00004	0	0.00021	0.00062	0.00002	0.00017	0
ayrıl [Verb]	0	0	0	0	0	0.00012	0.00274	0.01416	0.00731	0.00002	0.00052	0	0.00083	0.00038	0.00008	0.00016	0.00534
parti [Noun]	0	0.00016	0	0.00003	0	0.00555	0.00632	0.00002	0.00002	0	0.00088	0	0.00079	0.00228	0.00006	0.00041	0.00002
başkan [Noun]	0	0	0	0	0	0.00008	0.00128	0	0	0	0.00034	0	0.00105	0.00251	0	0.00016	0
düş [Verb]	0.01035	0	0.00004	0	0	0.00112	0.00258	0.03132	0.01617	0.00056	0.00264	0.00315	0.00038	0.00084	0.00014	0.00017	0.00182
uğra [Verb]	0	0	0	0	0	0.00007	0.00092	0.01088	0.01059	0.00002	0.00073	0.00153	0.00041	0.00039	0.00043	0.00013	0.00253
sonuç [Noun]	0	0	0	0	0	0.01272	0.00196	0.00432	0.00126	0	0.00545	0.00167	0.00001	0.00006	0.00002	0	0.00117
yaşam [Noun]	0	0	0	0	0	0.00172	0.00101	0	0	0	0.00008	0	0.00149	0.00151	0.00001	0.00056	0
yitir [Verb]	0	0	0	0	0	0.00006	0.00064	0.00701	0.00294	0.00004	0.00033	0.00106	0.00007	0.00056	0	0.00005	0.00119
zirve [Noun]	0	0	0	0	0	0.00931	0.00179	0	0	0	0.00008	0	0	0.00003	0	0	0
deplasman [Noun]	0	0	0	0.00009	0	0.01238	0.00035	0	0	0	0.00003	0	0	0.00002	0	0	0
devir [Verb]	0	0	0	0	0	0.00006	0.00021	0.00336	0.00066	0	0.00003	0	0.00001	0.00005	0.00001	0.00002	0.00039
rakip [Adj]	0	0	0	0	0	0.00041	0.00046	0	0	0	0.00041	0	0.00033	0.00411	0.00001	0.00045	0
ön [Noun]	0	0	0	0.00024	0	0.02533	0.00045	0.00001	0	0	0.00024	0	0.00276	0.05162	0	0.00129	0
oyun [Noun]	0	0.00003	0.00001	0.16744	0.00001	0.00697	0.01125	0	0.00001	0	0.00101	0.00001	0.00325	0.00458	0.00013	0.00023	0
risk [Noun]	0	0	0	0	0	0.00009	0.00119	0	0	0	0.00038	0	0.00001	0.00009	0	0.00013	0
et [Verb]	0.05877	0.00001	0.00055	0.00001	0	0.00495	0.01453	0.32044	0.17271	0.00117	0.05277	0.37143	0.01006	0.01241	0.00112	0.00545	0.03713

FIGURE 7. A small subset of the lemma-suffix matrix created from the Newscor part of the Boun Web Corpus.

from raw text, by replacing its default parser component with our parsing model.

B. ABLATION STUDY

We made an ablation study to see how each rule contributes to the overall performance of the rule-based parsing model. We started from the baseline where no rule is applied and then add the rules one by one to the model. At each step, we trained multiple models using different seeds for each setting and evaluated them on the development set of the IMST-UD Treebank. The attachment scores shown in this section are the averages of the attachment scores of these multiple models.

Table 2 shows the rules the parser uses at each step of the ablation study. In Step 5, we added ConsecAdv and AdvAdj rules to the parser at the same time. The reason for including these rules together is that, the ConsecAdv rule finds the consecutive adverbs that will possibly share the same head word and these consecutive adverb pairs are given to the AdvAdj and AdvVerb rules as input. When a dependency relation is constructed between an adverb and an adjective by the AdvAdj rule or between an adverb and a verb by the AdvVerb rule, these rules look up the consecutive adverbs list sent by the ConsecAdv rule and assign the same head to their corresponding pairs.

Similarly, we introduced the ConsecAdj and AdjNoun rules to the parser at the same time in Step 7. Because, the ConsecAdj rule finds the consecutive adjective pairs that should be set as dependent words to the same head word, and the list of these consecutive adjectives is given to the AdjNoun rule as input. When the AdjNoun rule forms a dependency relation between an adjective and a noun, it searches through the consecutive adjectives list and assigns the same head noun to the corresponding pairs of that adjective.

Fig. 8 shows the effect of each rule to the parsing performance in terms of the attachment scores. We observe that, all the rules except the AdvVerb and NounVerb rules improve the parsing performance whereas the AdvVerb and

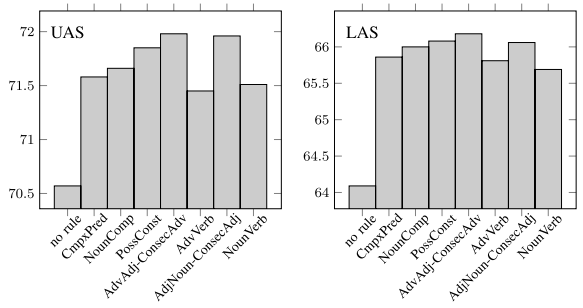


FIGURE 8. The effect of each rule to the parsing performance on the development set of the IMST-UD Treebank. Each rule is added on top of the previous rules. So, in the first step with the label *no rule*, there is no rule used in the model. In the second step, the CmpxPred rule is added to the model. In the third step, the NounComp rule is added to the model which means both the CmpxPred and NounComp rules are present in the model. The integration of the other rules proceeds in the same manner.

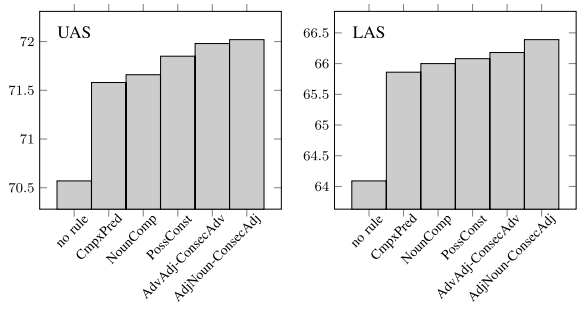


FIGURE 9. The effect of each rule to the parsing performance on the development set of the IMST-UD Treebank, when the AdvVerb rule and the NounVerb rule are removed from the rule-based parsing system. Each rule is added on top of the previous rules.

NounVerb rules cause a drop in both of the UAS and LAS scores. The possible reason behind this performance drop might be the over-generalizing structures of these rules. Considering the high frequency of complex sentences in Turkish which include one or more subordinate clauses in addition to the main clause, the risk of assigning the wrong verb as the head of an adverb is high for the AdvVerb rule. Similarly in the case of the NounVerb rule, there is a high probability of constructing a relation between a noun and a verb falsely when there are multiple verbs in a sentence.

TABLE 2. The ablation study steps for the rule-based parser.

Steps	Rules
Step 1	No rule
Step 2	CmpxPred
Step 3	CmpxPred + NounComp
Step 4	CmpxPred + NounComp + PossConst
Step 5	CmpxPred + NounComp + PossConst + ConsecAdv + AdvAdj
Step 6	CmpxPred + NounComp + PossConst + ConsecAdv + AdvAdj + AdvVerb
Step 7	CmpxPred + NounComp + PossConst + ConsecAdv + AdvAdj + AdvVerb + ConsecAdj + AdjNoun
Step 8	CmpxPred + NounComp + PossConst + ConsecAdv + AdvAdj + AdvVerb + ConsecAdj + AdjNoun + NounVerb

TABLE 3. Attachment Scores of the baseline parser, proposed models, and a state-of-the-art multilingual BERT-based parsing model (Udify) on the IMST-UD Treebank.

Parsing models	IMST-UD	
	UAS	LAS
Baseline [13]	$72.14_{\pm 0.4}$	$66.12_{\pm 0.3}$
Hybrid - rule	$74.03_{\pm 0.3}$	$67.99_{\pm 0.1}$
Hybrid - inflectional suffixes	$73.57_{\pm 0.3}$	$67.81_{\pm 0.2}$
Hybrid - last suffix	$73.95_{\pm 0.1}$	$68.25_{\pm 0.2}$
Hybrid - suffix vector	$73.09_{\pm 0.3}$	$66.96_{\pm 0.3}$
Hybrid - rule and last suffix	$\mathbf{74.37}_{\pm 0.4}$	$\mathbf{68.63}_{\pm 0.4}$
Udify [55]	$74.32_{\pm 0.2}$	$67.35_{\pm 0.3}$

So, we removed the AdvVerb and NounVerb rules from the rule-based parser and performed the ablation study again with the new setting. The effect of the rules to the performance is depicted in Fig. 9. We observe that each rule now improves the parsing scores which means that none of the rules blocks the other and each of them contributes to the parsing performance of the system. All of the experiments on the rule-based parser were performed using this final configuration of the rules.

C. RESULTS

a: COMPARISON OF THE MODELS

Table 3 shows the unlabeled and labeled attachment scores of the baseline system, our proposed enhancement models, and a state-of-the-art multilingual BERT-based parsing system Udify [55] on the test set of the IMST-UD Treebank. For each setting, the average and standard deviation across five runs are reported.

We have five different hybrid models that are built on top of the baseline model. Our first hybrid model is using the proposed rule-based approach explained in Section III-B. The second, third, and fourth hybrid models are the ones where we apply the corresponding versions of the morphology-based enhancement methods explained in Section III-D. The last hybrid model is the combination of the rule-based and morphology-based approaches. We selected the Last Suffix Model for this combination since it is the best performing one in the morphology-based methods. We combine these two models by simply concatenating their corresponding embedding vectors to the end of the original input word vector representation.

The results of the experiments show that all of our hybrid models outperform the baseline parser on the IMST-UD Treebank with p-values lower than 0.01 according to the performed randomization tests. We observe that the best performing model is the combination of the rule-based model and the Last Suffix Model with more than 2 and 2.5 points differences in, respectively, UAS and LAS scores when compared with the baseline model.

We see that, among the three morphology-based models, the best performing one is the Last Suffix Model. The success of this model matches with the observation that the last suffix of a word determines its role in a sentence [7]. This result suggests that the Last Suffix Model can accurately group the words using the last morpheme information for dependency parsing. Both of the UAS and LAS differences between this model and the other two morphology-based models are found to be statistically significant on the performed randomization test. From these results, we can conclude that the Last Suffix Model can be preferred over the other two morphology-based models with respect to the parsing performance and model simplicity. Although it outperforms the baseline model on both scores, the Suffix Vector Model is the worst performing one among the proposed methods. Its relatively low performance can be attributed to its complex structure which includes all of the unique suffixes in Turkish. Filtering some of the suffixes by putting a frequency threshold during the construction of the lemma-suffix matrix and lowering the dimension of the vectors might improve the performance of the Suffix Vector Model.

The performance of the rule-based model significantly outperforms the Suffix Vector Model on both UAS and LAS scores. When compared with the Inflectional Suffixes Model, the rule-based model outperforms the Inflectional Suffixes Model significantly on UAS score. However, its LAS score is only slightly better than the Inflectional Suffixes Model which leads the parsing accuracy difference to be insignificant in terms of LAS score. The rule-based model performs slightly worse than the Last Suffix Model according to the LAS score and slightly better than the Last Suffix Model according to the UAS score. Both differences are small and the performed randomization test results show that both of the differences are insignificant.

Yet, the best performance is reached when we combine the rule-based model with the Last Suffix model. The combined model outperforms all of the other models and the performance differences are found to be statistically significant. This result suggests that rule-based and

TABLE 4. P-values for determining the statistical significance between the performance of models of this study.

			hybrid			
		rule	I.S.	L.S.	S.V.	rule & L.S.
baseline	UAS	< 0.01	< 0.01	< 0.01	< 0.01	< 0.01
	LAS	< 0.01	< 0.01	< 0.01	< 0.01	< 0.01
hybrid - rule	UAS	-	0.013	0.17	< 0.01	0.058
	LAS	-	0.15	0.07	< 0.01	0.015
hybrid - I.S.	UAS		-	0.035	<0.01	<0.01
	LAS		-	0.02	< 0.01	<0.01
hybrid - L.S.	UAS			-	< 0.01	0.03
	LAS			-	< 0.01	0.06
hybrid - S.V.	UAS				-	< 0.01
	LAS				-	< 0.01
udify	UAS					0.18
	LAS					< 0.01

TABLE 5. Comparison of our best hybrid model on the IMST-UD test set with the top performing parsing systems in CoNLL 2018 shared task on multilingual parsing from raw text to universal dependencies.

Parsing models	IMST-UD		
	LAS	MLAS	BLEX
Hybrid	65.06	**55.94**	**60.69**
HIT-SCIR	**66.44**	53.81	56.72
TurkuNLP	64.79	55.73	60.13
UDPipe Future	63.07	54.02	56.69
ICS PAS	63.54	52.51	58.89
CEA LIST	63.78	55.00	54.29

morphology-based approaches improve different aspects of the deep learning-based parser on the dependency parsing of Turkish. Actually we observe that, the rule-based model is more successful in establishing dependency relations between words, whereas the Last Suffix Model is better at determining the relation types.

We also compared our models with the current state-of-the-art parsing system Udify, a transformer-based multilingual multitask model that reached or exceeded state-of-the-art performance on many languages [55]. It utilizes multilingual BERT as its language model. We fine-tuned the multilingual Udify model on the training set of the IMST-UD Treebank and evaluated on the test set of the IMST-UD Treebank using gold segmentation and tokenization. We observe that our best hybrid model outperforms Udify significantly on LAS and performs slightly better than Udify on UAS.

All experiments were repeated five times. The p-values of the model comparisons were obtained by performing the approximate randomization test [56] as described in [57] on the model outputs and can be found in Table 4. We set the number of shuffles in the approximate randomization testing to 10,000.

b: PARSING PERFORMANCE ON RAW TEXT

We also measured the parsing performance of our best hybrid model when the task is parsing from raw text where there is no gold segmentation or tokenization available. As in every other parsing system, the performance of our model was also affected negatively by the usage of automatic segmentation, tokenization, and tagging instead of the gold ones. Table 5 shows the comparison of our best parsing model with the state-of-the-art on parsing raw text. The systems in the table are the five best performing parsers in the CoNLL 18 Shared Task on Multilingual Dependency Parsing from Raw Text to Universal Dependencies [17]. The shared task used three evaluation metrics to sort the participating systems. These metrics are namely LAS, MLAS, and BLEX. MLAS (morphology-aware labeled attachment score) is an extension of the LAS metric, however it mainly focuses on dependencies between content words and treats function words as features of content words. It also takes the POS tags and morphological features into account. BLEX (bi-lexical dependency score) is similar to MLAS in focusing on relations between content words. However it includes lemmatization instead of morphological features to the evaluation [17].

In terms of LAS, our best model (using both rule-encodings and last-suffix information) is ranked second among the top five best systems participated to the shared task. The best performing system is HIT-SCIR [16] which incorporates an ensemble of three instances of Stanford's neural parser [13] trained with different initializations and contextual word embeddings. Although HIT-SCIR is ranked first in LAS, our model outperforms it and is ranked first in MLAS and BLEX metrics. Our model also outperforms TurkuNLP, which again uses Stanford's neural parser as the parsing component, in all three metrics.

c: EFFECT OF RULE- AND MORPHOLOGY-BASED ENHANCEMENTS ON PARSING

In Table 6, we depict the results of an analysis made on the proposed hybrid method to see how rules and morphology are affecting the parsing performance and what percentage of the input tokens are covered with these methods. We performed this analysis on the test set of the IMST-UD Treebank and we excluded the punctuation from the analysis. We observe that the rules are covering 36.07 per cent of the total tokens in the test set whereas the last-suffix information is included in 52.38 per cent of the tokens. We further analyzed the tokens that get a rule-encoding to see how much different rules contributed to this amount. We counted the tokens encoded by the NounComp rule and the PossConst rule together because the areas of operation of these rules overlap and sometimes the same rule-encoding is used by both rules. We did the same thing for AdvAdj-ConsecAdv and AdjNoun-ConsecAdj rule pairs too. From the statistics in the first part of Table 6, we see that 62.07 per cent of the rule-encodings are resulted from the NounComp-PossConst rule pair. The AdjNoun-ConsecAdj rule pair follows it with 19.30 per cent. The AdvAdj-ConsecAdv rule pair has 10.18 per cent of the rule-encodings and the remaining 8.05 per cent rule-encodings is resulted from the CmpxPred rule.

In the second part of Table 6, the performance of the hybrid and baseline parsers on the tokens for which the last-suffix information is available and on the tokens which are assigned a rule-encoding is given. On 4,295 tokens with last-suffix information, the hybrid parser made approximately 3.5 points and 4.5 points improvement over the baseline parser in UAS and LAS, respectively. On 2,958 tokens with rule-encodings, the hybrid parser outperforms the baseline parser by almost

TABLE 6. Analysis of the proposed methods on the IMST-UD test set.

Total token count	Tokens with last-suffix	Tokens with rule-encodings				
			CmpxPred count	NounComp & PossConst count	AdvAdj & ConsecAdv count	AdjNoun & ConsecAdj count
8,199	4,295	2,958	250	1,836	301	571
	52.38%	36.07%	8.05%	62.07%	10.18%	19.30%
Performance of the parsers:	**on tokens with last-suffix (4,295 tokens)**	**on tokens with rule-encodings (2,958 tokens)**				
Performance of the hybrid parser:						
UAS:	73.04	78.23	72.40	78.27	82.40	78.46
LAS:	65.66	69.47	60.80	68.95	80.40	69.18
Performance of the baseline parser:						
UAS:	69.61	75.42	64.80	75.70	82.05	75.65
LAS:	61.14	66.83	55.20	66.67	78.73	66.19

3 points in both scores. Both performance differences being greater than the performance differences between the hybrid and baseline parsers on the whole test set (Table 3) signals the contribution of the two enhancements. It also suggests that increasing the coverage of the rules and morphology is likely to result in better parsing scores. When we compare the two parsers on individual rules, we see that the biggest effect is caused by the CmpxPred rule with adding almost 6 points to UAS and 5.5 points to LAS over the performance of the baseline. The AdjNoun-ConsecAdj rule pair improves the scores by 3 points, the effect of the NounComp-PossConst rule pair is 2.5 points in both scores. The AdvAdj-ConsecAdv has the lowest improvement rates with only a slight difference on UAS and almost 2 points increase in LAS.

d: ERROR ANALYSIS ON THE RULES

To measure the impact of each rule on the parsing decision more explicitly, we form the following two questions: (i) For a given rule, of all the words that are given the correct rule-encoding by that rule, how many of them are predicted correctly (true positive: TP) by the parser and how many are missed (false negative: FN)? (ii) For a given rule, of all the words that are given a wrong rule-encoding by that rule, how many of them are predicted falsely (false positive: FP) by the parser and how many are predicted correctly (true negative: TN) in spite of the misleading rule-encoding?

To answer these questions, we created confusion matrices from the outputs of the hybrid parser which employs only the rule enhancement (hybrid - rule) and the baseline parser. Fig. 10 depicts these matrices for the CmpxPred rule and NounComp-PossConst, AdvAdj-ConsecAdv, and AdjNoun-ConsecAdj rule-pairs. We observe that the existence of a correct rule-encoding helps the hybrid parser to make the right parsing decision (TP) and reduces the FN rates, (i.e., setting up the wrong dependency relation) when compared to the baseline parser. On the other hand, if the word is given a wrong rule-encoding, this time we observe that the hybrid parser has a bias towards this rule-encoding which results in a higher FP rate and a lower TN rate for the case of CmpxPred rule and the NounComp-PossConst rule pair. However, this negative effect is small compared to the positive effect of the rule-based enhancement on the hybrid parser. We observe that even if the rules are not 100% correct individually, our proposed approach of incorporating the knowledge obtained from them into the neural parser's learning has been successful in dependency parsing of Turkish.

We conclude that the proposed methods increase the parsing accuracy of Turkish which has insufficient amount of training data. The aim of our approach is to give the parser additional information in constructing the dependency relations when learning from the training data is inadequate, i.e. there is not sufficient data to learn specific relations. The results show that both the rule-based and morphology-based enhancements on the neural parser improve the parsing accuracy significantly. The experiments performed on Turkish suggest that the languages with rich morphology need language-specific treatments and remarkably benefit from the usage of the basic grammar rules as well as from the inclusion of morphological suffix information.

D. THE EFFECT OF TRAINING DATA SIZE

After the experiments made on the IMST-UD Treebank for measuring the performance of each proposed model, we made additional experiments in a larger setup to understand how the improvement gained by the hybrid approach changes with different amounts of training data.

The training set of the IMST-UD Treebank consists of 3,685 sentences. To be able to observe the effect of the gradual increase of the training data more accurately, we additionally used the BOUN Treebank [52], a newly introduced Turkish treebank annotated in UD style. Being the largest dependency treebank in Turkish, the BOUN Treebank includes a total of 9,761 manually annotated sentences (7,803 training, 979 development, and 979 test sentences) from various topics including biographical texts, national newspapers, instructional texts, popular culture articles, and essays.

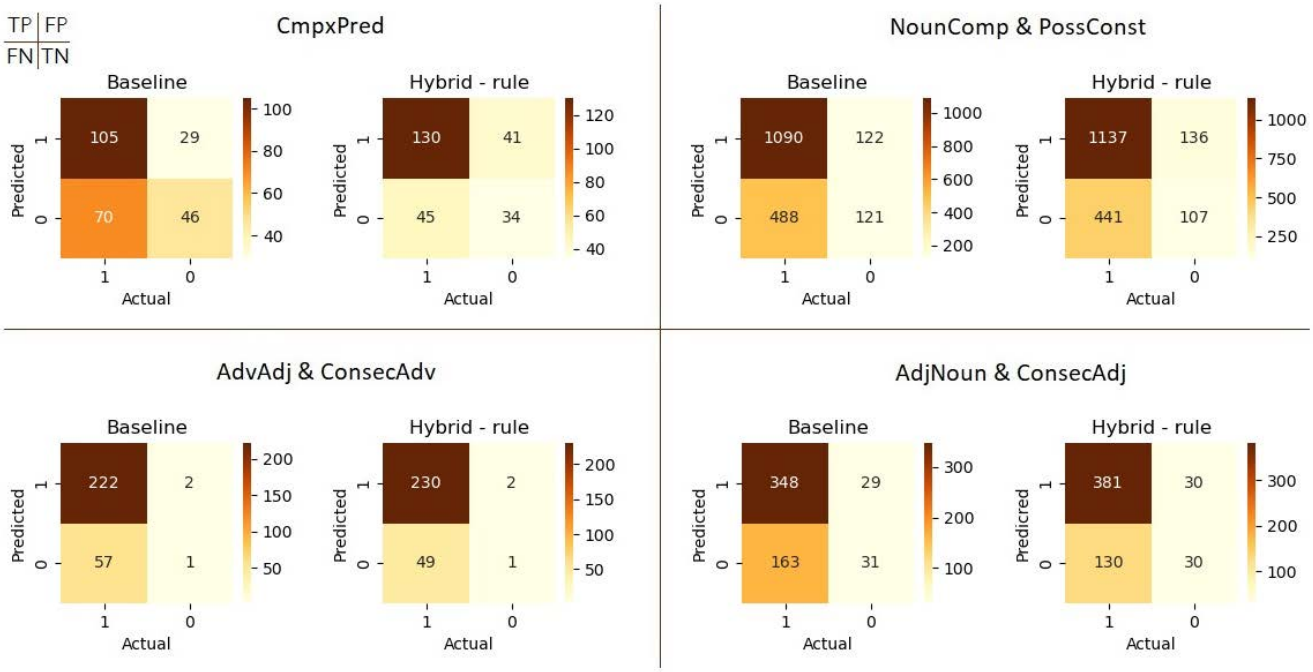


FIGURE 10. Confusion matrices for the case of detecting complex predicates (upper-left corner), noun compounds and possessive constructions (upper-right corner), adverb-adjective relations (lower-left corner), and adjective-noun relations (lower-right corner). Label 1 on axis *Actual* means the rule-encoding assigned is correct (e.g., the word is assigned the rule-encoding `cmp` by the CmpxPred rule and the word *is* a part of a complex predicate) and Label 0 on axis *Actual* means the rule-encoding assigned is wrong (e.g., the word is assigned the rule-encoding `cmp` but the word is *not* a part of a complex predicate). Label 1 on axis *Predicted* means the parser predicts the relation of the word in accordance with its rule-encoding (e.g., the word is assigned the rule-encoding `cmp` and the parser predicts it as a part of a complex predicate) and Label 0 on axis *Predicted* means the parser predicts the relation of the word by conflicting its rule-encoding (e.g., the word is assigned the rule-encoding `cmp` but the parser does *not* predict it as a part of a complex predicate).

The source texts were taken from the Turkish National Corpus (TNC) [58]. Decisions regarding the annotation of the BOUN Treebank were made in line with the recent efforts for unifying the Turkish UD treebanks through manual re-annotation [51]. In this context, the IMST-UD Treebank and Turkish PUD Treebank were also re-annotated manually [51], [59] and these re-annotated versions comply with the BOUN Treebank in terms of annotation decisions. Although the current version of the PUD Treebank (which is also the version used in our work) is this re-annotated version, the re-annotated version of the IMST-UD Treebank has not been validated. So, we use the original version of the IMST-UD Treebank in all our experiments. However, note that there are some major differences in the annotations of the current version of the IMST-UD Treebank and the BOUN Treebank. For more detailed information, see [59] and [52].

For this second set of experiments, the training set of the IMST-UD Treebank was split into 7 batches of 500 sentences (the last batch includes 685 sentences), and the training set of the BOUN Treebank was split into 8 batches of 1000 sentences (the last batch includes 803 sentences). Then the training sets used in the experiment were created by first adding the 500-sentence batches of IMST-UD on top of each other one by one and then continuing the process with the 1000-sentence batches of the BOUN Treebank.

In each setup, the trained models were evaluated on four different test sets. These test sets are the test set of the IMST-UD Treebank, the test set of the BOUN Treebank, the Turkish PUD Treebank, and the combined set of these three sets.

Fig. 11 depicts the results of these experiments. The four plots in the first row show the UAS performance of the proposed hybrid model and the baseline model on the four test sets and the plots in the second row demonstrate the LAS performance of the models. The vertical dashed line in the plots shows the point where additional BOUN training sentences are beginning to be added on top of the training set of the IMST-UD Treebank.

From these performance curves in the figure, we observed the following:

- The proposed hybrid model outperforms the baseline model consistently across all the test sets at every step.
- Adding additional training sentences from the BOUN Treebank has a positive effect on the parsing performance on all test sets except the test set of the IMST-UD Treebank. The fluctuating performance change on the IMST-UD test set results from the annotation difference between the IMST-UD Treebank and the BOUN Treebank.
- The performance of the baseline model on the PUD test set does not always increase when there are only IMST-UD sentences in the training set. When we start to add sentences from the BOUN Treebank, the performance is first decreased and then started to increase

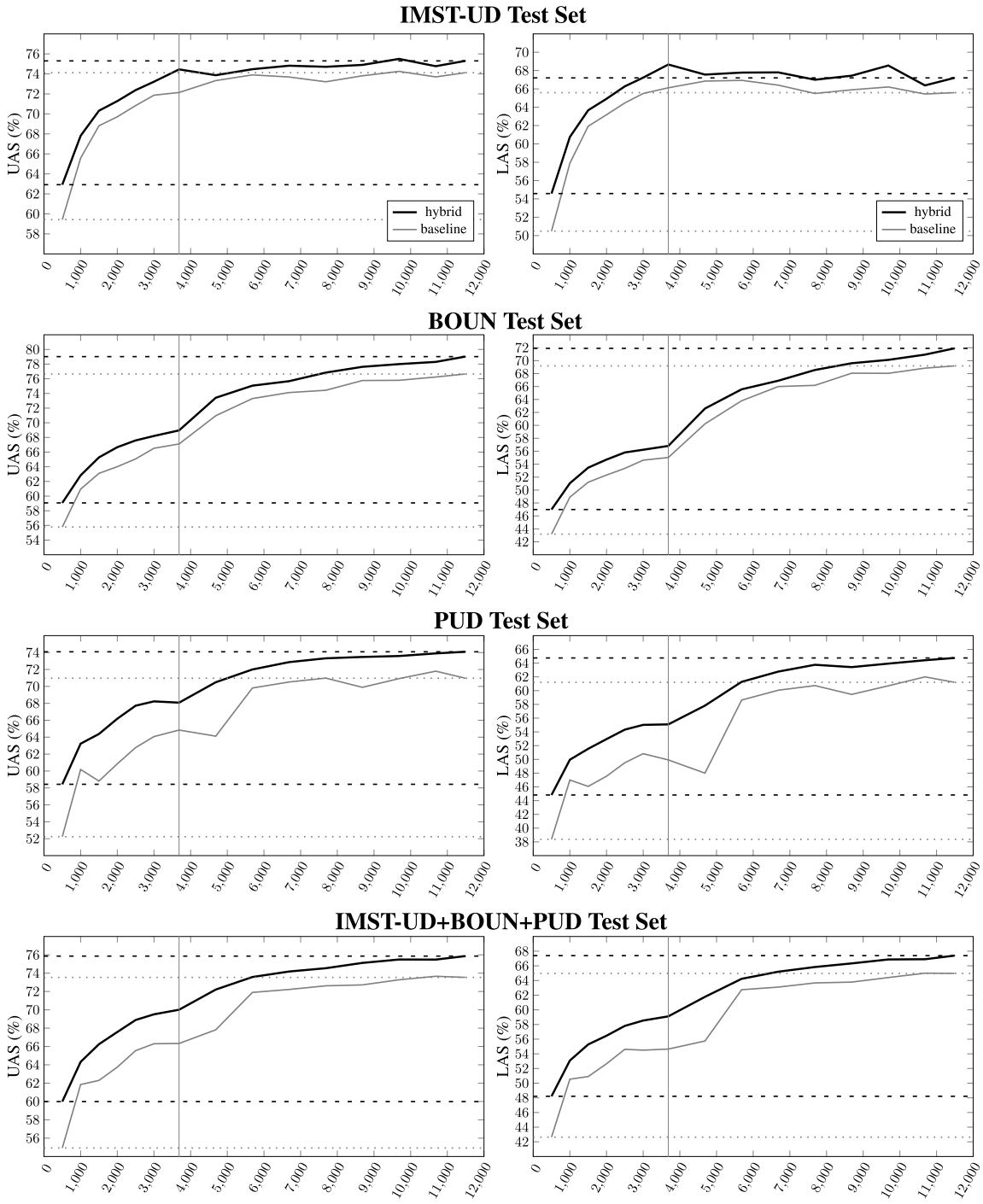


FIGURE 11. Effect of increasing the training data size on parsing performance of our hybrid model and the baseline model [13] in terms of UAS and LAS. The vertical line in the plots denotes the point where additional BOUN sentences are beginning to be added to the training data. The horizontal lines show the performance differences between two parsers at the beginning and end of this incremental process (the black horizontal dashed line is for the hybrid parser and the gray horizontal dotted line is for the baseline parser).

again as more BOUN sentences are included in the training set. This might again stem from the conflicting annotation differences between IMST-UD and BOUN treebanks.

- Across the BOUN, IMST-UD, and PUD test sets, the performance gap between the hybrid and the baseline parsers is 4.3 points in UAS and 4.8 points in LAS on average when training data consists of 500 sentences. The size of this gap decreases to 2.2 points in UAS and 2.6 points in LAS when training size is 11,488 sentences.

E. GENERALIZATION TO OTHER LANGUAGES

Although this study focuses on the improvement of Turkish dependency parsing, the proposed models can be adapted to

other languages as well. Below we first state the adaptation of the rule-based component, which is followed by the adaptation of the morphology-based component.

1) ADAPTING THE RULE-BASED PARSING APPROACH TO A TARGET LANGUAGE

For the rule-based parser, the ConsecAdj, AdvVerb, AdjNoun, and NounVerb rules can be directly applied to any language as long as the underlying structures exist in the language. The ConsecAdv and AdvAdj rules use adverb lists to make decisions. They can be applied to any language by supplying the adverbs specified in the rule descriptions for that language.

The PossConst rule needs a small adaptation to work for other languages because it uses the genitive and possessive marks when constructing possessive compounds. The genitive and possessive marks should be changed with the corresponding ones of the language, if applicable.

For the CmpxPred rule which handles complex predicates, we use a dictionary of complex predicates and idioms for Turkish. If there is a similar structure in the language the model is adapted to, supplying such a dictionary to the CmpxPred rule will be sufficient. Similarly, providing the three lexicons that separate between noun compounds, possessive compounds, and reduplicated compounds for the target language will be sufficient to adapt the NounComp rule.

We stated above how the rules proposed in this study can be adapted to other languages. In addition to such rule adaptations, and more importantly, effective rules for any language can be found by making a manual error analysis on the parser outputs. The grammar rules that hold for the relations between verbs, nouns, adverbs, and adjectives in a sentence can easily be applied for that language in general. The rule applications can then be integrated with the dependency parser by following the proposed strategy explained in Section III-C.

2) ADAPTING THE MORPHOLOGY-BASED ENHANCEMENT APPROACH TO A TARGET LANGUAGE

To create our suffix-based models, we utilize a morphological analysis tool for Turkish. A similar approach can be followed for any target language with agglutinative morphology. All of the three morphology-based models can be adapted by extracting the derivational and inflectional affixes in the target language. Moreover, other suffix-based models (e.g. using particular suffixes rather than the last suffix) can be employed depending on the specifics of the language by using the strategy proposed in Section III-D.

V. CONCLUSION AND FUTURE WORK

In this paper, we introduced a new rule-based method and three morphology-based methods for improving the accuracy of a deep learning-based parser on Turkish dependency parsing. In the rule-based approach, decisions made by the rule-based system are integrated into the word representation model of the deep learning-based parser. In the morphology-based approach, we experimented with the morphemes of the words by investigating different methods to integrate the information extracted from the morphemes into the word vector representation model of the parser. We observed that the best method of utilizing morphological information in terms of the dependency parsing of Turkish is using the last suffix of a word. A combination of the rule-based and morphology-based approaches outperforms all of the other proposed models as well as the baseline system, suggesting that Turkish dependency parsing benefits both from the linguistic grammar rules and the additional morphological information extracted from the input words. The experimental results show that our enhancement methods are useful for purely deep learning-based parsers, especially when there is not sufficient amount of training data to learn the dependency relations. The results also indicate that the best performing model of the proposed approaches outperforms the state-of-the-art on parsing of the IMST-UD Treebank. Our code is available at the github repository https://github.com/sb-b/BOUN-PARS.

As future work, we plan to adapt our methods to other languages with restricted amount of annotated data. We believe that applying the proposed models to these languages will improve their parsing accuracies. Moreover, combining the proposed models with a transformers-based language model can give further improvements in dependency parsing of morphologically rich languages.

REFERENCES

[1] C. Dyer, M. Ballesteros, W. Ling, A. Matthews, and N. A. Smith, "Transition-based dependency parsing with stack long short-term memory," in *Proc. 53rd Annu. Meeting Assoc. Comput. Linguistics*, Beijing, China, 2015, pp. 334–343.
[2] T. Dozat and C. D. Manning, "Deep biaffine attention for neural dependency parsing," in *Proc. 5th Int. Conf. Learn. Represent. (ICLR)*, Toulon, France, 2017, pp. 1–9.
[3] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," *J. Mach. Learn. Res.*, vol. 15, pp. 1929–1958, Jun. 2014.
[4] V. P. Sudha and R. Kowsalya, "A survey on deep learning techniques applications and challenges," *Int. J. Adv. Res. Sci. Eng.*, vol. 4, no. 3, pp. 311–317, Mar. 2015.
[5] E. Brill, "A simple rule-based part of speech tagger," in *Proc. 3rd Conf. App. Natural Lang. Process. (ANLP)*, Trento, Italy, 1992, pp. 152–155.
[6] S. Poria, E. Cambria, L.-W. Ku, C. Gui, and A. Gelbukh, "A rule-based approach to aspect extraction from product reviews," in *Proc. 2nd Workshop Natural Lang. Process. Social Media (SocialNLP)*, Dublin, Ireland, 2014, pp. 28–37.
[7] K. Oflazer, "Dependency parsing with an extended finite-state approach," *Comput. Linguistics*, vol. 29, no. 4, pp. 515–544, Dec. 2003.
[8] R. Sennrich, G. Schneider, M. Volk, and M. Warin, "A new hybrid dependency parser for German," in *Proc. GSCL*, Potsdam, Germany, 2009, pp. 115–124.
[9] L. Ramasamy and Z. Žabokrtský, "Tamil dependency parsing: Results using rule based and corpus based approaches," in *Proc. 12th Int. Conf. Intell. Text Process. Comput. Linguistics (CICLing)*, Berlin, Germany, 2011, pp. 82–95.
[10] I. Boguslavsky, L. Iomdin, V. Sizov, L. Tsinman, and V. Petrochenkov, "Rule-based dependency parser refined by empirical and corpus statistics," in *Proc. Int. Conf. Dependency Linguistics*, Barcelona, Spain, 2011, pp. 318–327.
[11] M. Korzeniowski and J. Mazurkiewicz, "Rule based dependency parser for Polish language," in *Proc. Int. Conf. Artif. Intell. Soft Comput.*, Zakopane, Poland, 2017, pp. 498–508.

[12] E. Kiperwasser and Y. Goldberg, "Simple and accurate dependency parsing using bidirectional LSTM feature representations," *Trans. Assoc. Comput. Linguistics*, vol. 4, pp. 313–327, Jul. 2016.

[13] T. Dozat, P. Qi, and C. D. Manning, "Stanford's graph-based neural dependency parser at the CoNLL 2017 shared task," in *Proc. CoNLL Shared Task, Multilingual Parsing Raw Text Universal Dependencies*, Vancouver, BC, Canada, 2017, pp. 20–30.

[14] D. Zeman, M. Popel, M. Straka, J. Hajic, J. Nivre, F. Ginter, J. Luotolahti, S. Pyysalo, and S. Petrov, "CoNLL 2018 shared task: Multilingual parsing from raw text to universal dependencies," in *Proc. CoNLL Shared Task, Multilingual Parsing Raw Text Universal Dependencies*, Vancouver, BC, Canada, 2017, pp. 1–19.

[15] J. Kanerva, F. Ginter, N. Miekka, A. Leino, and T. Salakoski, "Turku neural parser pipeline: An end-to-end system for the CoNLL 2018 shared task," in *Proc. CoNLL Shared Task, Multilingual Parsing Raw Text Universal Dependencies*, Brussels, Belgium, 2018, pp. 133–142.

[16] W. Che, Y. Liu, Y. Wang, B. Zheng, and T. Liu, "Towards better UD parsing: Deep contextualized word embeddings, ensemble, and treebank concatenation," in *Proc. CoNLL Shared Task, Multilingual Parsing Raw Text Universal Dependencies*. Brussels, Belgium, Oct. 2018, pp. 55–64.

[17] D. Zeman, J. Hajič, M. Popel, M. Potthast, M. Straka, F. Ginter, J. Nivre, and S. Petrov, "CoNLL 2018 shared task: Multilingual parsing from raw text to universal dependencies," in *Proc. CoNLL Shared Task, Multilingual Parsing Raw Text Universal Dependencies*, Brussels, Belgium, 2018, pp. 1–21.

[18] R. Tsarfaty, D. Seddah, Y. Goldberg, S. Kuebler, Y. Versley, M. Candito, J. Foster, I. Rehbein, and L. Tounsi, "Statistical parsing of morphologically rich languages (SPMRL) what, how and whither," in *Proc. 1st Workshop Stat. Parsing Morphol.-Rich Lang.*, Los Angeles, CA, USA, 2010, pp. 1–12.

[19] D. Zeman and Z. Žabokrtský, "Improving parsing accuracy by combining diverse dependency parsers," in *Proc. 9th Int. Workshop Parsing Technol.*, Vancouver, BC, Canada, 2005, pp. 171–178.

[20] E. Ruppert, J. Klesy, M. Riedl, and C. Biemann, "Rule-based dependency parse collapsing and propagation for German and English," in *Proc. GSCL*, 2015, pp. 58–66.

[21] B. R. Ambati, S. Husain, J. Nivre, and R. Sangal, "On the role of morphosyntactic features in Hindi dependency parsing," in *Proc. 1st Workshop Stat. Parsing Morphol.-Rich Lang.*, Los Angeles, CA, USA, 2010, pp. 94–102.

[22] Y. Goldberg and M. Elhadad, "Easy-first dependency parsing of Modern Hebrew," in *Proc. 1st Workshop Stat. Parsing Morphol.-Rich Lang.*, Los Angeles, CA, USA, 2010, pp. 103–107.

[23] Y. Marton, N. Habash, and O. Rambow, "Improving Arabic dependency parsing with lexical and inflectional morphological features," in *Proc. 1st Workshop Stat. Parsing Morphol.-Rich Lang.*, Los Angeles, CA, USA, 2010, pp. 13–21.

[24] N. Habash and O. Rambow, "Arabic tokenization, part-of-speech tagging and morphological disambiguation in one fell swoop," in *Proc. 43rd Annu. Meeting Assoc. Comput. Linguistics (ACL)*, Ann Arbor, MI, USA, 2005, pp. 573–580.

[25] J. Nivre, J. Hall, J. Nilsson, A. Chanev, G. Eryiğit, S. Kübler, S. Marinov, and E. Marsi, "MaltParser: A language-independent system for data-driven dependency parsing," *Natural Lang. Eng.*, vol. 13, no. 2, pp. 95–135, Jun. 2007.

[26] C. Vania, A. Grivas, and A. Lopez, "What do character-level models learn about morphology? The case of dependency parsing," in *Proc. Conf. Empirical Methods Natural Lang. Process.*, Brussels, Belgium, 2018, pp. 2573–2583.

[27] Ş. B. Özateş, A. Özgür, T. Güngür, and B. Öztürk, "A morphology-based representation model for LSTM-based dependency parsing of agglutinative languages," in *Proc. CoNLL Shared Task, Multi. Parsing Raw Text Universal Dependencies*, Brussels, Belgium, 2018, pp. 238–247.

[28] M. Dehouck and P. Denis, "Phylogenic multi-lingual dependency parsing," in *Proc. Conf. North*, Minneapolis, MN, USA, 2019, pp. 192–203.

[29] J. Nivre, M.-C. de Marneffe, F. Ginter, Y. Goldberg, J. Hajič, C. Manning, R. McDonald, S. Petrov, S. Pyysalo, N. Silveira, R. Tsarfaty, and D. Zeman, "Universal dependencies v1: A multilingual treebank collection," in *Proc. 10th Int. Conf. Lang. Res. Eval. (LREC)*, Portorož, Slovenia, 2016, pp. 1659–1666.

[30] N. Silveira, T. Dozat, M.-C. de Marneffe, S. Bowman, M. Connor, J. Bauer, and C. Manning, "A gold standard dependency corpus for English," in *Proc. 9th Int. Conf. Lang. Resour. Eval. (LREC)*, Reykjavik, Iceland, 2014, pp. 2897–2904.

[31] U. Sulubacak, M. Gokörmak, F. Tyers, Ç. Çöltekin, J. Nivre, and G. Eryiğit, "Universal dependencies for Turkish," in *Proc. COLING*, Osaka, Japan, 2016, pp. 3444–3454.

[32] U. Sulubacak and G. Eryiğit, "Implementing universal dependency, morphology, and multiword expression annotation standards for Turkish language processing," *Turkish J. Elect. Eng. Comput. Sci.*, vol. 26, no. 3, pp. 1662–1672, May 2018.

[33] C. C. Çöltekin, "A grammar-book treebank of Turkish," in *Proc. 14th Workshop Treebanks Linguistic Theories (TLT)*, Warsaw, Poland, 2015, pp. 35–49.

[34] A. Göksel and C. Kerslake, *Turkish: A Comprehensive Grammar*. London, U.K.: Routledge, 2005. [Online]. Available: https://books.google.de/books?id=7fXCKZmee8QC

[35] G. Eryiğit and K. Oflazer, "Statistical dependency parsing for Turkish," in *Proc. 11th Conf. Eur. Chap. Assoc. Comput. Linguistics*, Trento, Italy, 2006, pp. 89–96.

[36] K. Oflazer, B. Say, D. Z. Hakkani-Tür, and G. Tür, "Building a Turkish treebank," in *Treebanks*, A. Abeillé, Ed. Dordrecht, The Netherlands: Springer, 2003, pp. 261–277.

[37] G. Eryiğit, J. Nivre, and K. Oflazer, "Dependency parsing of Turkish," *Comput. Linguistics*, vol. 34, no. 3, pp. 357–389, Sep. 2008.

[38] G. Eryiğit, T. I. Lbay, and O. A. Can, "Multiword expressions in statistical dependency parsing," in *Proc. 2nd Workshop Stat. Parsing Morphol. Rich Lang.*, Dublin, Ireland, 2011, pp. 45–55.

[39] J. Hall, J. Nilsson, J. Nivre, G. Eryiğit, B. Megyesi, M. Nilsson, and M. Saers, "Single malt or blended? A study in multilingual parser optimization," in *Proc. Conf. Empirical Methods Natural Lang. Process. Comput. Natural Lang. Learn. (EMNLP-CoNLL)*, Prague, Czech Republic, 2007, pp. 933–939.

[40] I. D. El-Kahlout, A. A. Akın, and E. Yılmaz, "Initial explorations in two-phase Turkish dependency parsing by incorporating constituents," in *Proc. 1st Joint Workshop Stat. Parsing Morphol. Rich Lang. Syntactic Anal. Non-Canonical Lang.*, Dublin, Ireland, 2014, pp. 82–89.

[41] W. Seeker and Ö. Çetinoğlu, "A graph-based lattice dependency parser for joint morphological segmentation and syntactic analysis," *Trans. Assoc. Comput. Linguistics*, vol. 3, pp. 359–373, Dec. 2015.

[42] U. Sulubacak, G. Eryiğit, and T. Pamay, "IMST: A revisited Turkish dependency treebank," in *Proc. TurCLing*, Konya, Turkey, 2016, pp. 1–7.

[43] B. Öztürk, "Case, referentiality and phrase structure," Ph.D. dissertation, Dept. Linguistics, HU, Cambridge, MA, USA, 2004.

[44] C. Keskin, *Subject Agreement-Dependency of Accusative Case in Turkish or Jump-Starting Grammatical Machinery*. Amsterdam, The Netherlands: LOT, 2009. [Online]. Available: https://www.lotpublications.nl/Documents/229_fulltext.pdf

[45] F. Akkuş, "Light verb constructions in Turkish: A case for DP predication and blocking," in *Proc. Workshop Formal Altaic Linguistics*, Ithaca, NY, USA, 2013, pp. 133–145.

[46] E. Solak, "Accusative licensing of nouns in Turkish," in *Proc. Workshop Turkic Lang. Contact Turkic*, Toronto, ON, Canada, 2021, p. 5052.

[47] C. Ramisch, S. Cordeiro, A. Savary, V. Vincze, V. Mititelu, A. Bhatia, M. Buljan, M. Candito, P. Gantar, V. Giouli, and T. Güngör, "Edition 1.1 of the PARSEME shared task on automatic identification of verbal multiword expressions," in *Proc. Joint Workshop Linguistic Annotation, Multiword Expressions Construct.*, Santa Fe, NM, USA, 2018, pp. 222–240.

[48] C. S. H. Akalın, R. Toparlı, and B. T. Aksu, *Atasözleri ve Deyimler Sözlüğü*. Ankara, Turkey: Türk Dil Kurumu, 2009.

[49] E. Erguvanlı Taylan, *The Function Word Order Turkish Grammar*, vol. 106. Oakland, CA, USA: Univ. California Press, 1984, [Online]. Available: https://books.google.de/books?id=-Cjl4uqiYEYC

[50] H. Sak, T. Güngör, and M. Saraçlar, "Resources for Turkish morphological processing," *Lang. Resour. Eval.*, vol. 45, no. 2, pp. 249–261, May 2011.

[51] U. Türk, F. Atmaca, Ş. B. Özateş, A. Köksal, B. O. Basaran, T. Gungor, and A. Özgür, "Turkish treebanking: Unifying and constructing efforts," in *Proc. 13th Linguistic Annotation Workshop*, Florence, Italy, 2019, pp. 166–177.

[52] U. Türk, F. Atmaca, Ş. B. Özateş, G. Berk, S. T. Bedir, A. Köksal, B. Ö. Başaran, T. Güngör, and A. Özgür, "Resources for Turkish dependency parsing: Introducing the BOUN treebank and the BoAT annotation tool," *Lang. Resour. Eval.*, vol. 56, no. 1, pp. 259–307, Mar. 2022.

[53] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *Proc. 3rd Int. Conf. Learn. Represent. (ICLR)*, Y. Bengio and Y. LeCun, Eds., San Diego, CA, USA, 2015, pp. 1–15.

[54] F. Ginter, J. Hajič, J. Luotolahti, M. Straka, and D. Zeman. (2017). *CoNLL 2017 Shared Task—Automatically Annotated Raw Texts and Word Embeddings*. [Online]. Available: http://hdl.handle.net/11234/1-1989
[55] D. Kondratyuk and M. Straka, "75 languages, 1 model: Parsing universal dependencies universally," in *Proc. Conf. Empirical Methods Natural Lang. Process. 9th Int. Joint Conf. Natural Lang. Process. (EMNLP-IJCNLP)*, Hong Kong, 2019, pp. 2779–2795.
[56] E. W. Noreen, *Computer-Intensive Methods for Testing Hypotheses*. New York, NY, USA: Wiley, 1989.
[57] S. Riezler and J. T. Maxwell, "On some pitfalls in automatic evaluation and significance testing for MT," in *Proc. ACL Workshop Intrinsic Extrinsic Eval. Measures Mach. Transl. Summarization*, Ann Arbor, MI, USA, 2005, pp. 57–64.
[58] Y. Aksan, M. Aksan, A. Koltuksuz, T. Sezer, U. Mersinli, U. U. Demirhan, H. Yılmazer, G. Atasoy, S. Öz, I. Yıldız, and O. Kurtoğlu, "Construction of the Turkish national corpus (TNC)," in *Proc. 8th Int. Conf. Lang. Res. Eval. (LREC)*, İstanbul, Turkey, 2012, pp. 3223–3227.
[59] U. Türk, F. Atmaca, Ş. B. Özateş, B. Öztürk Başaran, T. Güngör, and A. Özgür, "Improving the annotations in the Turkish universal dependency treebank," in *Proc. 3rd Workshop Universal Dependencies (UDW, SyntaxFest)*, Paris, France, 2019, pp. 108–115.

ŞAZİYE BETÜL ÖZATEŞ received the B.S. and M.S. degrees in computer engineering from Boğaziçi University, and the Ph.D. degree from Boğaziçi University, under the guidance of Prof. Arzucan Özgür and Prof. Tunga Güngör.

From 2020 to 2022, she was a Researcher with the Institute of Natural Language Processing, University of Stuttgart. She is a member of the Text Analytics and Bioinformatics Laboratory (TABILAB). She has been supported by the TÜBİTAK BİDEB 2211 Scholarship during her M.S. and Ph.D. studies. Her current research interest includes automatic processing of low-resource natural languages with the help of deep learning methods.

ARZUCAN ÖZGÜR received the B.S. and M.S. degrees in computer engineering from Boğaziçi University, and the Ph.D. degree in computer science and engineering from the University of Michigan, in 2010.

She is currently an Associate Professor at the Computer Engineering Department, Boğaziçi University. Her research interests include natural language processing and bioinformatics. Her recent focus has been on developing methods for processing and understanding textual data in natural languages or in the sequences of biomolecules.

Dr. Özgür was a recipient of the European Commission Marie Curie Career Integration Grant (2012–2016) for research on contextual text mining from the biomedical scientific literature, the Science Academy Young Scientist Award (BAGEP 2016, Turkey), and the Turkish Science Academy Outstanding Your Scientist Award (TÜBA-GEBİP 2019, Turkey).

TUNGA GÜNGÖR received the M.S. and Ph.D. degrees from the Department of Computer Engineering, Boğaziçi University.

He was a Visiting Professor at the Center for Language and Speech Technologies and Applications, Universitat Politecnica de Catalunya, Barcelona, Spain, from 2011 to 2012. He is currently a Senior Lecturer and a Researcher with the Department of Computer Engineering, Boğaziçi University. He is a member of the Artificial Intelligent Laboratory and the Text Analytics and Bioinformatics Laboratory, Department of Computer Engineering. He teaches undergraduate and graduate level courses on the topics of artificial intelligence, natural language processing, machine translation, and algorithm analysis. He participated as the Project Leader in projects about developing an adaptive question answering system for primary and secondary education students, developing concept mining methods for document analysis, developing a hand-written recognition system using a large lexicon, morphology-based language modeling for speech recognition, and developing structure-preserving and query-biased automated summarization methods. The projects were funded by the Turkish Scientific and Technological Research Council of Turkey and the national funds. He has published about 90 scientific articles and participated in several research projects and conference organizations. His research interests include natural language processing, machine translation, machine learning, and pattern recognition.

BALKIZ ÖZTÜRK BAŞARAN received the B.A. degree in translation studies and the M.A. degree in linguistics from Boğaziçi University, İstanbul, and the Ph.D. degree in linguistics from Harvard University, in 2004.

She is currently a Professor of linguistics at Boğaziçi University. She has published extensively on argument structure, case, relative clauses, pseudo-incorporation, NP-structure, null-arguments in journals and edited volumes. She is the author of *Case, Referentiality and Phrase Structure* (John Benjamins, 2005) and has co-edited the volumes *Exploring the Turkish Linguistic Landscape* (John Benjamins, 2016), *Pazar Laz* (Lincom, 2011), and *Morphological Complexities Within and Across Boundaries* (John Benjamins, 2020). Her main research interests include interfaces of syntax, morphology and lexicon specifically focusing on Altaic (Turkish, Uyghur, and Mongolian), and South Caucasian languages (Laz and Georgian).