IMST: A Revisited Turkish Dependency Treebank

Umut Sulubacak∗, Tuğba Pamay†, Gülşen Eryiğit‡
Department of Computer Engineering,
Istanbul Technical University,
Istanbul, 34469, Turkey.
Email: [∗sulubacak, †pamay, ‡gulsen.cebiroglu]@itu.edu.tr

***Abstract*—In this paper, we present a critical analysis of the dependency annotation framework used in the METU-Sabancı Treebank (MST), and propose new annotation schemes that would alleviate the issues we have identified. Later, we describe our attempt at reannotating the treebank from the ground up using the proposed schemes, and then compare the consistencies of the two versions via cross-validation using a dependency parser. According to our experiments, the reannotated version of the original treebank, which we call the ITU-METU-Sabancı Treebank (IMST), demonstrates a labeled attachment score of 75.3% and an unlabeled attachment score of 83.7%, surpassing the corresponding scores of 65.9% and 76.0% for MST by a very large margin.**

I. Introduction

Despite the considerable interest in Turkish syntax, parsing performances have not seen a major improvement in a long time, as evidenced by several recent case studies [6], [11], [17], [18], [30], [35]. Many studies seem to be concentrating on specific computational or linguistic issues, fine-tuning certain aspects of their parsers and leaving the rest untouched. As a result, although many still demonstrate local improvements, they fail to make any pivotal progress. As certain issues remain in focus and others fall behind the spotlight, a considerable portion of the field remains uncharted.

It is likely that there are some issues outside the domain of well-researched cases that create a bottleneck for syntactic parsing. Considering that virtually all state-of-the-art parsers make use of supervised learning from human-annotated corpora, it is entirely possible that the issues stem from imperfections in the training corpora. The METU-Sabancı Turkish Treebank (MST) [29] has proved to be an invaluable resource over the years, and has been utilized by almost every Turkish dependency parser to date. However, its dependency grammar has come to be criticized on occasion from various standpoints, and it is known to contain a large amount of annotation inconsistencies, as also attested in some previous works [5], [12]. At present, there is no other available resource[1] for Turkish that would be equivalent or an alternative to MST. This further conceals any issues with the corpus that might otherwise emerge.

In the light of these considerations, it could be worthwhile to take a detour from specific case studies and directly tackle the corpus, which, decidedly, has ample room for improvement. The effort would also be promising in alleviating certain problems commonly attributed to the corpus, such as excessive parsing difficulty [4] and cross-parser instability [25]. Although engaging in a tedious investigation in order to recondition a corpus may not seem cost-effective, previous successful attempts for other prominent languages [2], [22], [27], [39] provide a strong motivation for the effort.

In this paper, we propose changes in certain dependency schemes, leading to an updated annotation framework for Turkish. We thereby aim to relieve some of the known difficulties in the current framework, as well as to reduce stress on human annotators and thus alleviate manual annotation errors. We also present the ITU-METU-Sabancı Treebank (IMST), a new version of MST reannotated from the ground up following this new framework. Later, we make empirical evaluations on our new treebank and report our results. The paper is structured as follows: Section 2 briefly outlines Turkish and the dependency formalism, Section 3 explains the problems and the proposed solutions, Section 4 introduces the new treebank, Section 5 describes the experiments, and finally, Section 6 presents the conclusion.

II. Turkish and the Dependency Formalism

Though the concept of dependencies has existed since some of the earliest recorded grammars [32], the modern *dependency grammar* is commonly attributed to Tesnière [37]. The formalism has seen a great deal of attention and extensive usage in computational linguistics in recent years. Essentially, a dependency grammar defines a set of practical rules on how to utilize dependencies to model the syntax of a sentence.

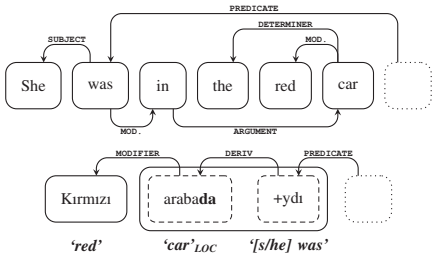


Fig. 1: An example dependency tree for a sentence in Turkish and English. Note that the definite article does not occur in the Turkish sentence, and the English dependency to the preposition 'in' is analogous to the Turkish locative suffix '-da'.

[1]There is the ITU Validation Set [14], [15], [20], but it is a fairly small corpus containing only 300 sentences, and is meant to be a validation or test set for supervised learners, therefore not suitable for training data-driven models.

In this work, as also in the majority of modern syntactic studies for Turkish, we adopt the dependency formalism. The formalism necessitates the representation of syntactic information with sets of directed binary relations (*dependencies*) between tokens (Fig. 1). Each dependency is defined between a governing token (the *head*) and a subordinate token that modifies it (the *dependent*), and represented by labeled arcs from the *head* to the *dependent*. The labels assigned to dependencies indicate the type of the relation, called the *dependency type*. For a recent discussion of the dependency formalism, the interested reader may refer to [23].

Turkish is a classical example of an agglutinative morphologically rich language incorporating a large number of productive derivational suffixes. For example, the suffix 'ydı' ('*[s/he] was*') in Fig. 1 is a third-person singular past copula attached to the stem 'araba' ('*car*'). As different portions of such derived words may correspond to several words in a weakly inflected language such as English, Turkish sentences often comprise relatively fewer, highly inflected words. In order to properly analyze the syntax of a Turkish sentence, words are divided from derivational boundaries into morphosyntactic units called *inflectional groups* (IGs). This formalism establishes tokens as the IGs comprising the sentence, rather than orthographic words. Words with multiple IGs are quite prevalent in Turkish—in fact, it is not unusual to find words with as many as four or five IGs. Having been practiced in many influential works [19], [18], [21], [28], their usage has become the de facto standard for parsing Turkish.

III. Problems and Proposed Solutions

In designing a dependency annotation framework, it is essential to have a clear definition of the dependency relations and the set of conventions on when to use which relation. Although dependency relations would be ideally expressive, exclusive, coherent and concise, there are often trade-offs between some of these properties. As such, it becomes a challenge to balance a grammar around them. Considering the drawbacks of the original MST, we reason that prioritizing clarity and aiming for a minimal dependency grammar makes the better sense in mitigating inconsistency and obscurity.

As foundations for our work, we carried out an in-depth manual error analysis on the original MST. Among the most frequent cases that we noted were inconsistently or erratically annotated linguistic constructions as well as standard annotation methods that mandated the usage of certain particles that were optional in informal language. The subsections below present our attempt to loosely categorize the questionable cases that we encountered. For each issue, we also provide example cases, discuss our own standpoints and finally describe our proposed annotation schemes.

In the process of settling on local annotation schemes, we investigated the corresponding methods followed in some other prominent frameworks [3], [8], [9], [10], [27], [38] and reviewed previous work in the subject [24], [33], [34]. Through all these, we laid strong foundations for our decisions.

Throughout the rest of the section, we regularly refer to our proposed annotation framework, though a description of the whole framework is not provided in this article. The full list of the proposed dependency types and their usages is provided within a separate annotation manual [36].

A. Semantic Incoherence

In the original framework, some dependency relations were used in a way that is contradictory to their semantic connotations. Such cases occurred especially in less prevalent secondary usages of common dependency types. Though it might have seemed counter-productive to handle such cases under exclusive dependency types or another encompassing type, we maintain that the incoherence is generally less favorable, as they would confuse the associations drawn by annotators. Even though this phenomenon was not very common, it occurred frequently enough to warrant notice.

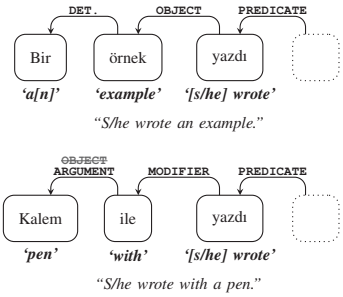


Fig. 2: The **OBJECT** relation used for the object of the main verb *(top)* and for an adpositional phrase argument *(bottom)*.

One example is that adpositional phrases were connected via the dependency label **OBJECT**. Although dependents of adpositional phrases are sometimes called adpositional objects, they are in fact arguments of the adpositional head and unrelated to sentence (or clausal) objects. Not only was it not immediately obvious that they should be regarded as objects, but also this annotation method confused parsers and made the prediction of objects difficult. We assign these the new dependency label **ARGUMENT** along with the rest of the phrasal arguments.

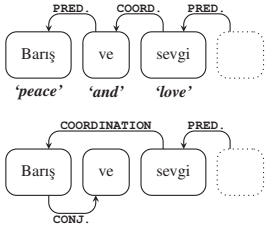


Fig. 3: An example showing the original *(top)* and the proposed *(bottom)* annotation scheme for coordination structures.

Another case was in coordination structures, where the coordinating conjunction was connected to the succeeding token with the dependency label **COORDINATION**. This constitutes a counter-intuitive scenario which semantically implied that the token is in coordination with the conjunction itself,

whereas the tokens should be in coordination with each other, as also attested in [1]. We make it so that tokens are connected directly to the next token in coordination, while preserving the **COORDINATION** label. This approach has also been previously proved to improve parsing performances in [35], who applied automatic conversion routines to map coordination structures to different styles and compared local performances.

B. Hierarchy and Overlap

In the original framework, certain dependency relations fell within the scope of others. As the grammar did not enact a dependency hierarchy to exploit granularity in dependency types, this also had a negative effect. The immediate impact was on annotators, for whom it occasionally became arbitrary which dependency type to use. Parsing frameworks also suffered from increased entropy in prediction. Yet another impact was on evaluation, as such cases caused some sound dependency annotations to be considered incorrect because any label other than that which was in the gold-standard would be a mismatch.

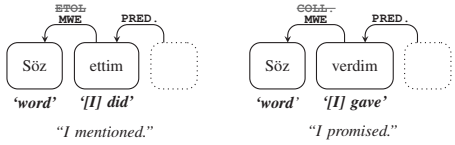
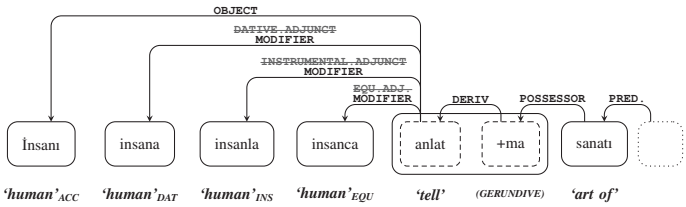


Fig. 4: Two similar idiomatic expressions indicated by the dependency relations **ETOL** and **COLLOCATION**.

An example to this (Fig. 4) is the sub-type **ETOL**, which comprised a group of multiword expressions incorporating certain auxiliary verbs, otherwise denoted by the label **COLLOCATION**. We eliminate such types altogether.



"The art of relating humans to humans, with humans, like humans."

Fig. 5: Nominal adjuncts serving as modifiers were mostly indicated by different **X.ADJUNCT** labels according to their cases.

There were also some cases where a dependency relation overlapped with another in usage, giving way to confusion. This was the most obvious between the label **MODIFIER** and the **X.ADJUNCT** labels for every noun declension (such as **DATIVE.ADJUNCT**), which are also effectively modifiers. For instance, while generic adjuncts that did not fall into a specific category would use a **MODIFIER** label and a regular nominal adjunct in the locative case would use the label **LOCATIVE.ADJUNCT**, certain other adjuncts, which were grammatically nouns in the locative case, would still be assigned a **MODIFIER** label due to semantic concerns. To address this complication, we preserve only the **MODIFIER** label (Fig. 5) and eliminate the **X.ADJUNCT** labels, which are at any rate reproducible using morphological information.

C. Ambiguous Annotation

For certain annotation schemes, the framework clearly defined what the head should be, but not the dependency relation (or vice versa). This encouraged arbitrary annotation, or else annotation conventions that were quite difficult for annotators to memorize, which impaired the annotation consistency. Although at times this would be due to linguistic relations not properly explained by any dependency label, this was mostly observed in cases of ambiguity, when a relation could be possibly explained by more than one label. For such cases, we introduce new dependency types where the involved dependencies would be common enough to represent a group.

An instance of this phenomenon was seen in phrasal arguments, which were not precisely covered under any dependency type, and were variously assigned **MODIFIER** or **OBJECT** labels. We introduce the new dependency label **ARGUMENT** for all cases where exactly one argument is syntactically required to modify a head, such as in adpositional phrases, in contrast to modifiers, of which a head could have more than one, or none at all.

D. Optional Annotation

In the original framework, only certain types of punctuation (usually conjunctive punctuation and terminal periods) had dependency types associated with them, and the rest were allowed to pass without any head (Fig. 6). These tokens were connected to an arbitrary head and assigned the label NOT-CONNECTED. This indicated that the dependency grammar essentially did not enforce dependencies for all tokens in a sentence, which is required by most dependency parsers, leading to complexity in evaluation. Furthermore, since NOT-CONNECTED was computationally considered to be a regular dependency type in parsing, learning performances were also indirectly affected. To address this issue, we introduce the new label **PUNCTUATION** and standardize the annotation scheme of all types of punctuation, as well as eliminating the support for optionality in the grammar. In this approach, all punctuation should be connected at all times with the **PUNCTUATION** relation to the last non-punctuation token occurring before it. Punctuation that begins a sentence should be connected to the sentence's root node instead (Fig. 2).

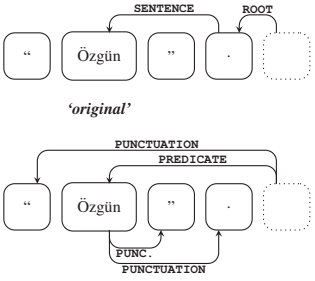
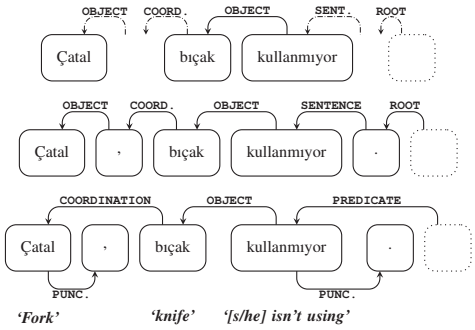


Fig. 6: Certain kinds of punctuation that were allowed to pass without a head *(top)* are now covered by the new dependency type **PUNCTUATION** *(bottom)*.

E. Reliance on Omissible Tokens

Some annotation schemes required certain tokens to occur in a specific position within the sentence, and could not be properly applied when they were omitted. This prevented regular annotation in case of omission, and caused uncertainty as to how to alternatively mark the relation, which led to annotation inconsistencies. For instance, coordination structures were annotated with a dependency from the first constituent to the coordinating conjunction and another dependency from the coordinating conjunction to the next constituent, which made proper annotation impossible when the coordinating conjunction was omitted. Adverse cases are not uncommon in non-canonical language, most notably web jargon, where some common function words are frequently dropped in favor of brevity. Examples are encountered even in well-typed sentences, caused by less conventional, idiomatic or archaic usages. Therefore, the issue warranted addressing.



"S/he doesn't use a knife and a fork."

Fig. 7: Reliance on omissible tokens in the original annotation framework
The sentences show a case where annotation is impossible *(top)*, except by the addition of conjunctive and terminal punctuation *(middle)*. The scheme we propose *(bottom)* is not affected by this.

Reliance was perhaps the most noticeable in terminal periods (Fig. 7), which were essential in marking the main predicate of the sentence. The annotation required the predicate to be connected to the terminal period with the label **SENTENCE**. This scheme left no option for legitimately omitting periods, as practiced very frequently in non-canonical language. To address this, we make it so that predicates are connected directly to the sentence root with the renamed dependency label **PREDICATE**, making terminal periods properly optional.

IV. The ITU-METU-Sabanci Treebank

In order to have an indication of the impact of our proposed schemes and provide future studies with a new and fresh training corpus, we annotated the entire METU-Sabancı Treebank from the ground up. We call this reannotated corpus the ITU-METU-Sabancı Treebank (IMST). The annotation of IMST was carried out in parallel with the ITU Web Treebank [31], an original corpus of user-generated web data that was released earlier. This section provides details about IMST[2].

[2]The treebank underwent some minor revisions before its release and is at version 1.3 at the time of this publication. The latest version will be made available for research purposes on `http://tools.nlp.itu.edu.tr/`.

For the new corpus, we used an updated version of our ITU Annotation Tool [13]. Five annotators were employed, one linguist and four computer scientists with considerable experience in NLP research. Our annotators were well-versed in Turkish morphology and syntax, and underwent two weeks of supervised training in the new annotation framework before starting on the annotation. Dependency annotation was made on gold-standard tokens with pre-allocated morphological analyses[3], and was completed within a span of two months.

Although the annotation was started with two annotators for each sentence, our annotators eventually had to work individually on their exclusive shares of the data due to budgetary constraints. As a consequence, it was not possible to measure inter-annotator agreement. Nonetheless, after the initial annotation, sentences from both corpora were carefully inspected for inconsistent annotation, and a correction phase followed for two weeks, which led to the final version.

A. Deep Dependencies

Another detail to mention about the annotation is that we set out to indicate *deep* (or *unbounded*) dependencies in IMST. *Deep* dependencies are secondary dependencies of tokens to other logical heads, often with different dependency relations, in addition to their regular *surface* dependencies. The annotation of these dependencies violates the restriction of each constituent having a single head and thereby makes a corpus incompatible with most syntactic parsers without preprocessing. However, deep dependencies are favored often because they function as cues for semantic parsers designed to determine the semantic roles of verbal arguments in a sentence. In IMST, we regularly draw deep dependencies as substitutes for coreference links from zero pronouns as well as to mark shared modifiers for tokens in coordination.

B. Corpus Statistics

For a proper comparison between MST and IMST, we provide a particular selection of comparative statistics before describing our syntactic accuracy tests. Table I displays sentence, token and dependency counts for either corpora. Table II shows the distribution of dependencies by dependency relation.

TABLE I: Comparative sentence, token and dependency statistics.

	METU-Sabanci Treebank	ITU-METU-Sabanci Treebank
# Sentences	**5,635**	**5,635**
# Words	**56,424**	**56,424**
# Tokens (IG)	**67,403**	**63,089**
# Single-headed Tokens	**67,403** (100.0%)	**60,688** (96.2%)
# Multi-headed Tokens	—	**2,401** (3.8%)
# Dependencies (excl. Deriv)	**56,424**	**59,425**
# Dependencies (incl. Deriv)	**67,403**	**66,090**
# Projective Dependencies	**66,145** (98.1%)	**64,663** (97.8%)
# Non-projective Dependencies	**1,258** (1.9%)	**1,427** (2.2%)

V. Evaluation

This section presents the statistical analysis we performed on MST and IMST. Section V-A contains preliminary information about our parsing and evaluation systems. Section V-B shows the test outcome and a brief discussion of the results.

[3]The morphological tags were inherited from a version of MST following a revised morphological annotation framework established in [7], [16].

TABLE II: Distribution of the dependency relation labels.

	METU-Sabanci Treebank	ITU-METU-Sabanci Treebank
Ablative.Adjunct	**523** (0.8%)	—
Apposition	**202** (0.3%)	**91** (0.1%)
Argument	—	**1,805** (2.7%)
Conjunction	—	**1,360** (2.1%)
Classifier	**2,050** (3.0%)	—
Collocation	**73** (0.1%)	—
Coordination	**2,476** (3.7%)	**3,078** (4.7%)
Dative.Adjunct	**1,361** (2.0%)	—
Deriv	**10,979** (16.3%)	**6,665** (10.1%)
Determiner	**1,952** (2.9%)	**2,180** (3.3%)
Equ.Adjunct	**16** (0.0%)	—
Etol	**10** (0.0%)	—
Focus.Particle	**23** (0.0%)	—
Instrumental.Adjunct	**271** (0.4%)	—
Intensifier	**903** (1.3%)	**1,070** (1.6%)
Locative.Adjunct	**1,142** (1.7%)	—
Modifier	**11,690** (17.3%)	**15,516** (23.5%)
MWE	**2,432** (3.6%)	**3,552** (5.4%)
Negative.Particle	**160** (0.2%)	—
Object	**8,338** (12.4%)	**5,094** (7.7%)
Possessor	**1,516** (2.2%)	**4,070** (6.2%)
Predicate	—	**5,741** (8.7%)
Punctuation	—	**10,375** (15.7%)
Question.Particle	**289** (0.4%)	—
Relativizer	**85** (0.1%)	**129** (0.2%)
Root	**5,644** (8.4%)	—
S.Modifier	**597** (0.9%)	—
Sentence	**7,261** (10.8%)	—
Subject	**4,481** (6.6%)	**5,174** (7.8%)
Vocative	**241** (0.4%)	**190** (0.3%)
(disconnected tokens)	**2,688** (4.0%)	—

A. Preliminaries

In our test, we used the same MaltParser [26] configuration as in [17] so that the results would be properly comparable. In further accordance with the cited work, non-projective sentences were eliminated from all training sets, which is shown to cause a significant performance boost [17], [18]. The dependencies with the relation **Deriv** (denoting intra-word relations between morphosyntactic units) were excluded in evaluation, as they are considered trivial. In the literature, punctuation is either wholly excluded from evaluation (as in e.g. [4]) or included (as in e.g. [25]). We follow the latter approach and evaluate the dependencies of punctuation. Since the inherited parsing framework does not support learning from dependents annotated with multiple heads, we discard all deep dependencies from IMST before running the test.

The metrics used in evaluation are the conventional IG-based labeled and unlabeled attachment scores. The unlabeled attachment score (UAS) considers a prediction to be accurate if the head token alone was correctly predicted, while the labeled attachment score (LAS) additionally requires a correct prediction of the dependency relation. Between the two, a high LAS is more difficult to attain and more valuable, so we take the LAS as our primary criterion in performance comparison. We also provide standard error values, and use McNemar's Test for measuring statistical significance where needed.

B. Experimental Results

Parsing performances obtained by applying ten-fold cross-validation on IMST are shown side-by-side with the corresponding scores for MST in Table III.

TABLE III: Cross-validation scores and standard error values.

	LAS	UAS
METU-Sabanci Treebank	$65.9\% \pm 0.3\%$	$76.0\% \pm 0.2\%$
ITU-METU-Sabanci Treebank	$75.3\% \pm 0.2\%$	$83.7\% \pm 0.2\%$

As shown in Table I, the number of words and evaluated dependencies (excluding **Deriv**) is exactly the same between the two corpora. The slight difference between the dependency counts as seen in Table I is due to the updated morphological analysis framework mentioned earlier in Section IV and the entailed difference in derivational boundaries. The changes in IG bounding should only affect the performance of morphological analysis and have a negligible effect on parsing.

Comparing the current LAS of 75.3% for IMST with the corresponding score of 65.9%[4] for MST shows that we manage an increase of nearly 10 percentage points. The UAS seems to have improved in a similar way, increasing to 83.7% for IMST and passing the score of 76.0% for MST by a large margin.

VI. Conclusion

In this article, we initially described the annotation schemes we designed based on the dependency grammar of the METU-Sabancı Treebank (MST). Our new annotation framework incorporates only 16 dependency relation labels in contrast to the 24 labels of the baseline, but features generally clearer and more intuitive dependency types with reduced overlap between each other, hopefully relieving the difficulty of manual annotation without suffering any loss in expressiveness.

Afterwards, we presented the ITU-METU-Sabancı Treebank (IMST) as a reannotated version of MST that followed our revised annotation framework. We additionally marked deep dependencies in IMST to pave the way for future semantic role labeling studies. We substantiate the theoretical advantages of our proposed annotation schemes through a parsing experiment in compliance with the parsing framework used in the study for the original MST that still remains the state of the art. Our experiment yielded a labeled attachment score of 75.3% for IMST, surpassing the best score of 65.9% attained so far on MST by a very large margin.

Finally, considering the outcome of our work, we believe it would be safe to say that we succeeded in making pivotal progress by working directly on the training set. We show that improving the quality of data, although an open-ended endeavor, has a considerable effect on parsing performances, and will hopefully pave the way for corpus studies for Turkish.

Acknowledgement

This study is part of a research project entitled "*Parsing Web 2.0 Sentences*" subsidized by the Turkish Scientific and Technological Research Council under grant number 112E276 and associated with the ICT COST Action IC1207. We hereby offer our sincere gratitude to our volunteering annotators Dilara Torunoğlu-Selamet and Ayşenur Genç, as well as our colleagues Can Özbey, Kübra Adalı and Gözde Gül İşgüder who offered additional help with the annotation.

[4]The MST score was evaluated excluding punctuation, in accordance with the conventions at the time. As we reproduced the baseline scores on the original MST, we found the difference between models including and excluding punctuation to be statistically insignificant ($p < 0.01$). Conversely, excluding punctuation in evaluating IMST resulted in a drop in LAS from 75.3% to 70.0%, indicating that the contribution of the new punctuation annotation is far from wholly accounting for the increase in parsing performance.

REFERENCES

[1] B. R. Ambati, S. Reddy, and A. Kilgarriff, "Word sketches for Turkish," in *Proceedings of the 8th International Conference on Language Resources and Evaluation (LREC)*, 2012, pp. 2945–2950.

[2] E. Bejček, J. Panevová, J. Popelka, P. Straňák, M. Ševčíková, J. Štěpánek, and Z. Žabokrtský, "Prague Dependency Treebank 2.5–a revisited version of PDT 2.0," in *Proceedings of the 24th International Conference on Computational Linguistics (COLING)*, 2012, pp. 231–246.

[3] A. Böhmová, J. Hajič, E. Hajičová, and B. Hladká, "The Prague Dependency Treebank," in *Treebanks*. Springer, 2003, pp. 103–127.

[4] S. Buchholz and E. Marsi, "CoNLL-X Shared Task on multilingual dependency parsing," in *Proceedings of the 10th Conference on Computational Natural Language Learning (CoNLL)*. Association for Computational Linguistics, 2006, pp. 149–164.

[5] R. Çakıcı, "Wide-coverage parsing for Turkish," Ph.D. dissertation, The University of Edinburgh, 2008.

[6] O. Çetinoğlu and J. Kuhn, "Towards joint morphological analysis and dependency parsing of Turkish," in *Proceedings of the 2nd International Conference on Dependency Linguistics (DepLing)*. Prague, Czech Republic: Charles University in Prague, Matfyzpress, Prague, Czech Republic, August 2013, pp. 23–32.

[7] M. Şahin, U. Sulubacak, and G. Eryiğit, "Redefinition of Turkish morphology using flag diacritics," in *Proceedings of The 10th Symposium on Natural Language Processing (SNLP)*, Phuket, Thailand, October 2013.

[8] D. Csendes, J. Csirik, T. Gyimóthy, and A. Kocsor, "The Szeged Treebank," in *Text, Speech and Dialogue*. Springer, 2005, pp. 123–131.

[9] M.-C. De Marneffe, M. Connor, N. Silveira, S. R. Bowman, T. Dozat, and C. D. Manning, "More constructions, more genres: Extending Stanford Dependencies," in *Proceedings of the 2nd International Conference on Dependency Linguistics (DepLing)*. Prague, Czech Republic: Charles University in Prague, Matfyzpress, Prague, Czech Republic, August 2013, pp. 187–196.

[10] M.-C. De Marneffe and C. D. Manning, "The Stanford Typed Dependencies representation," in *Proceedings of the Workshop on Cross-Framework and Cross-Domain Parser Evaluation (COLING)*. Association for Computational Linguistics, 2008, pp. 1–8.

[11] I. Durgar El-Kahlout, A. A. Akın, and E. Yılmaz, "Initial explorations in two-phase Turkish dependency parsing by incorporating constituents," in *Proceedings of the 1st Joint Workshop on Statistical Parsing of Morphologically Rich Languages (SPMRL) and Syntactic Analysis of Non-Canonical Languages (SANCL)*. Dublin, Ireland: Dublin City University, August 2014, pp. 82–89.

[12] G. Eryiğit, "Dependency parsing of Turkish," Ph.D. dissertation, Istanbul Technical University, 2006.

[13] ——, "ITU Treebank Annotation Tool," in *Proceedings of the ACL Workshop on Linguistic Annotation (LAW)*, Prague, 24-30 June 2007.

[14] ——, "ITU Validation Set for METU-Sabancı Turkish Treebank," March 2007. [Online]. Available: http://web.itu.edu.tr/gulsenc/papers/validationset.pdf

[15] ——, "The impact of automatic morphological analysis & disambiguation on dependency parsing of Turkish," in *Proceedings of the 8th International Conference on Language Resources and Evaluation (LREC)*, Istanbul, Turkey, 23-25 May 2012.

[16] ——, "ITU Turkish NLP Web Service," in *Proceedings of the Demonstrations at the 14th Conference of the European Chapter of the Association for Computational Linguistics (EACL)*. Gothenburg, Sweden: Association for Computational Linguistics, April 2014.

[17] G. Eryiğit, T. Ilbay, and O. A. Can, "Multiword expressions in statistical dependency parsing," in *Proceedings of the 2nd Workshop on Statistical Parsing of Morphologically Rich Languages (SPMRL)*. Dublin, Ireland: Association for Computational Linguistics, October 2011.

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

[19] G. Eryiğit and K. Oflazer, "Statistical dependency parsing of Turkish," in *Proceedings of the 11th Conference of the European Chapter of the Association for Computational Linguistics (EACL)*, Trento, April 2006, pp. 89–96.

[20] G. Eryiğit and T. Pamay, "ITU Validation Set," *Türkiye Bilişim Vakfı Bilgisayar Bilimleri ve Mühendisliği Dergisi*, vol. 7, no. 1, 2014.

[21] D. Z. Hakkani-Tür, K. Oflazer, and G. Tür, "Statistical morphological disambiguation for agglutinative languages," *Computers and the Humanities*, vol. 36, no. 4, pp. 381–410, 2002.

[22] K. Haverinen, J. Nyblom, T. Viljanen, V. Laippala, S. Kohonen, A. Missilä, S. Ojala, T. Salakoski, and F. Ginter, "Building the essential resources for Finnish: the Turku Dependency Treebank," *Language Resources and Evaluation*, pp. 1–39, 2013.

[23] S. Kübler, R. McDonald, and J. Nivre, *Dependency Parsing*, ser. Synthesis Lectures on Human Language Technologies. Morgan & Claypool Publishers, 2009.

[24] R. McDonald, J. Nivre, Y. Quirmbach-Brundage, Y. Goldberg, D. Das, K. Ganchev, K. Hall, S. Petrov, H. Zhang, O. Täckström, C. Bedini, N. Bertomeu Castelló, and J. Lee, "Universal dependency annotation for multilingual parsing," in *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (ACL)*. Sofia, Bulgaria: Association for Computational Linguistics, August 2013, pp. 92–97.

[25] J. Nilsson, S. Riedel, and D. Yüret, "The CoNLL 2007 Shared Task on dependency parsing," in *Proceedings of the CoNLL Shared Task Session of the Joint Conference on Empirical Methods in Natural Language Processing (EMNLP) and Computational Natural Language Learning (CoNLL)*. Association for Computational Linguistics, 2007, pp. 915–932.

[26] 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 Language Engineering*, vol. 13, pp. 95–135, 6 2007.

[27] J. Nivre, J. Nilsson, and J. Hall, "Talbanken05: A Swedish treebank with phrase structure and dependency annotation," in *Proceedings of the 5th International Conference on Language Resources and Evaluation (LREC)*, 2006, pp. 1392–1395.

[28] K. Oflazer, "Dependency parsing with an extended finite-state approach," *Computational Linguistics*, vol. 29, no. 4, pp. 515–544, 2003.

[29] K. Oflazer, B. Say, D. Z. Hakkani-Tür, and G. Tür, "Building a Turkish treebank," in *Treebanks*. Springer, 2003, pp. 261–277.

[30] Özlem Çetinoğlu, "Turkish Treebank as a gold standard for morphological disambiguation and its influence on parsing," in *Proceedings of the 9th International Conference on Language Resources and Evaluation (LREC)*. Reykjavík, Iceland: European Language Resources Association (ELRA), May 2014.

[31] T. Pamay, U. Sulubacak, D. Torunoğlu-Selamet, and G. Eryiğit, "The annotation process of the ITU Web Treebank," in *Proceedings of the 9th Linguistic Annotation Workshop (LAW)*, Denver, CO, USA, 5 June 2015.

[32] W. K. Percival, "Reflections on the history of dependency notions in linguistics," *Historiographia Linguistica*, vol. 17, no. 1-2, pp. 29–47, 1990.

[33] M. Popel, D. Mareček, J. Štěpánek, D. Zeman, and Z. Žabokrtský, "Coordination structures in dependency treebanks," in *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (ACL)*. Sofia, Bulgaria: Association for Computational Linguistics, August 2013, pp. 517–527.

[34] N. Schneider, B. O'Connor, N. Saphra, D. Bamman, M. Faruqui, N. A. Smith, C. Dyer, and J. Baldridge, "A framework for (under)specifying dependency syntax without overloading annotators," in *Proceedings of the 7th Linguistic Annotation Workshop and Interoperability with Discourse (LAW VII & ID)*. Sofia, Bulgaria: Association for Computational Linguistics, August 2013, pp. 51–60.

[35] U. Sulubacak and G. Eryiğit, "Representation of morphosyntactic units and coordination structures in the Turkish dependency treebank," in *Proceedings of the 4th Workshop on Statistical Parsing of Morphologically-Rich Languages (SPMRL)*. Seattle, Washington, USA: Association for Computational Linguistics, October 2013, pp. 129–134.

[36] ——, *ITU Treebank Annotation Guide*, March 2016, available at http://tools.nlp.itu.edu.tr, version 2.7.

[37] L. Tesnière, *Éléments de Syntaxe Structurale*. Éditions Klinksieck, 1959.

[38] L. Van der Beek, G. Bouma, R. Malouf, and G. Van Noord, "The Alpino Dependency Treebank," *Language and Computers*, vol. 45, no. 1, pp. 8–22, 2002.

[39] V. Vincze, V. Varga, K. I. Simkó, J. Zsibrita, A. Nagy, R. Farkas, and J. Csirik, "Szeged Corpus 2.5: Morphological modifications in a manually POS-tagged Hungarian corpus." in *Proceedings of the 9th International Conference on Language Resources and Evaluation (LREC)*, 2014, pp. 1074–1078.