Multiword Expressions in Statistical Dependency Parsing

Gülşen Eryiğit **Tugay İlbay** **Ozan Arkan Can**
Department of Computer Engineering
Istanbul Technical University
Istanbul, 34469, Turkey
{gulsen.cebiroglu,ilbay, cano}@itu.edu.tr

Abstract

In this paper, we investigated the impact of extracting different types of multiword expressions (MWEs) in improving the accuracy of a data-driven dependency parser for a morphologically rich language (Turkish). We showed that in the training stage, the unification of MWEs of a certain type, namely compound verb and noun formations, has a negative effect on parsing accuracy by increasing the lexical sparsity. Our results gave a statistically significant improvement by using a variant of the treebank excluding this MWE type in the training stage. Our extrinsic evaluation of an ideal MWE recognizer (for only extracting MWEs of type named entities, duplications, numbers, dates and some predefined list of compound prepositions) showed that the preprocessing of the test data would improve the labeled parsing accuracy by 1.5%.

1 Introduction

Multiword expressions are compound word formations formed by two or more words. They generally represent a different meaning than the words which compose them. The importance of detecting multiword expressions in different NLP problems is emphasized by many researchers and is still a topic which is being investigated for different NLP layers (Kordoni et al., 2011). It is impossible to neglect the importance for machine translation where the translation of a MWE would be totally different than the translation of its constituents. But the effect of different MWE types on parsing accuracy is still an open research topic and needs to be analyzed in detail.

Hogan et al. (2011) recently give their preliminary results on detecting named-entities and reports no improvement in parsing accuracy for English. Nivre and Nilsson (2004) reports a 5% parsing accuracy increase for Swedish by using a depedency parser which uses a memory-based learner as its oracle. In this study, they only focus on multiword names and compound function words. Cafferkey (2008) reports very small (falling short of their initial expectations) but statistically significant improvements for a PCFG parser (on English). Korkontzelos and Manandhar (2010) reports an improvement of sentence accuracy 7.5% in shallow parsing by concentrating on MWE of types compound nominals, proper names and adjective noun constructions. The conflicting results and the difference in success ranges may be caused by many factors; such as the parsing paradigm, the language in focus, the MWE types used in the experiments and the evaluation metrics.

In this study, we are making a detailed investigation of extracting different MWE classes as a preprocessing step for a statistical dependency parser (MaltParser v1.5.1 Nivre et al. (2007)). We conducted our experiments on Turkish Dependency Treebanks (Oflazer et al., 2003; Eryiğit, 2007). We semi-automatically created different versions of the data by manually annotating and classifying MWEs. We made an in depth analysis of using the new treebank versions both on training and testing stages. We evaluated the parser's performance both on MWEs and the remaining parts of the sentences.

Proceedings of the 2nd Workshop on Statistical Parsing of Morphologically-Rich Languages (SPMRL 2011), pages 45–55,
Dublin, Ireland, October 6, 2011.

Our results showed that different MWE types have different impacts on the parsing accuracy. For Turkish, we showed that the preprocessing of compound verb and noun formations causes a considerable decrease in accuracy. We also demonstrated that a MWE extractor which finds the MWEs from the remaining types would make a significant improvement for parsing. For now, it is not possible to generalize the results for other languages and parsing paradigms. But we believe, we present a systematic approach for evaluating the scenario.

2 Motivation

Eryigit et al. (2008) in their article "Dependency Parsing of Turkish" points out to a decrease of nearly 4 percentage points when they test their parser on the raw data. Although they only look at this decrease from the point of the errors caused by parts-of-speech (POS) tagging, the decrease could actually be due to two reasons: 1. The errors caused by the automatic POS tagging, 2. The lack of MWE handling which exists in the gold standard. In this study, we will focus on to the second item and on the improvement that could be reached by handling MWEs. With this purpose, we are asking and answering the following questions during the remaining of the paper:

1. What is the success of available MWE extractors on detecting the MWEs manually annotated in the treebank?
2. When we analyze the "false positives"[1] of the MWE extractors, we see that most of them are actually valid MWEs. Should we as well manually annotate these in the treebanks?
3. When we decide to annotate these MWEs[2], the results of the automatic parsing become worse then the previous results with the original treebank. Should we concentrate on different MWE types?

The paper is structured as follows: Section 3 makes a short description of the Turkish language. Section 4 presents the configuration used in our experiments. Section 5 gives our experimental results. Section 6 gives our conclusion and comments for future works.

[1] the group of words which are tagged as MWEs by the automatic analyzers but not in the current treebanks (somehow missed by the human annotators).

[2] described in the 2nd item

3 Turkish

Turkish is an agglutinative language with a very rich morphological structure. The dependencies between the words constructing a sentence is almost entirely head-final in the written text. The derivational and inflectional richness of the language results in shorter sentence lengths when compared to other languages (8.16 words/sentence[3]). In similar studies, the Turkish words are most of the time analyzed as a sequence of one or more inflectional groups (IGs). Each IG consists of either a stem or a derivational suffix plus all the inflectional suffixes belonging to that stem/derivational suffix. The head of a whole word is not just another word but a specific IG of another word. Figure 1 shows the IGs in a simple sentence: "küçük odadayım" (*I'm in the small room*). The word "odadayım" is formed from two IGs; a verb is derived from an inflected noun "odada" (*in the room*). In the example, the adjective "küçük" (*small*) should be connected to the first IG of the second word. It is the word "oda" (*room*) which is modified by the adjective, not the derived verb form "odadayım" (*I'm in the room*). So both the correct head word and the correct IG in the head word should be determined by the parser.

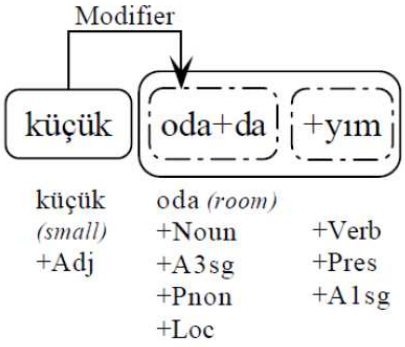

Figure 1: Word and dependency representations
A1sg = 1sg number/person agreement, A3sg = 3sg number/person agreement, Loc = Locative case, Pnon = No possessive agreement, Pres = Present Tense

These properties of the language makes it very

[3] where the number differs between 13.16-27.7 for other languages (Nivre et al., 2007)

hard for dependency parsing[4]. Buchholz and Marsi (2006) reports the Turkish Treebank having the highest number of different surface forms and the second with new lemma within 13 different languages. As a result, any change on lexical representation could end up with severe lexical sparsity problems.

4 Configuration

4.1 Parser

We are using MaltParser v1.5.1 (Nivre et al., 2007) which is a data-driven dependency parser whose success is reported to be very high across a variety of different languages (Nivre et al., 2006). For the repeatability of the results we used exactly the same feature representation and parser options from Eryigit et al. (2008) . The cited reference gives these options in details so we do not repeat them here again. The parser's current version uses a support vector machine (SVM) classifier for predicting the parser's actions. The usage of SVM in this area has been proven to be very successful. And we know that the parser's capability is very high at learning many syntactic structures especially the ones with shorter distances. In the following sections, we will see that the extraction of MWE types where the parser is already good at determining have a negative effect on parsing accuracy by increasing the lexical sparsity.

4.2 Data Sets

In our experiments, we are using the METU-Sabancı Turkish Treebank (Oflazer et al., 2003) which consists of 5635 sentences (in Conll format). The second column of Table 1 gives the statistics for the original treebank (Vo); there exist 2040 MWEs which are manually combined into single units. Figure 2-a gives the symbolic representation of MWEs in the original treebank. In the figure, w1 w2 w3 are the three constituents of a MWE and are collapsed into a single unit (by the use of underscores) which acts as a single word in the dependency structure. We created 3 new versions of the treebank:

1. Detached Version (Vd): is the version where the annotated MWEs are detached and a new dependency type "MWE" is created between the MWE constituents (Figure 2-b). The strategy that is adopted while creating these dependencies is to connect the last IG of the dependent to the first IG of the head except for compound functional words (explained later in this section). In the treebank, the gold-standard POS tag and inflectional features given for the MWE in focus is only valid for the last constituent of that MWE. After the detachment process, we need to assign the correct tags (token index, lemma, surface form, postag, inflectional features, dependency type and head index) to the new coming IGs (words are constructed from one or more IGs each having their own tags). In order to select the correct postags, we passed these words from a morphological analyzer (Oflazer, 1994) and then manually disambiguated the ones having more than one possible analyses (1265/2437 words). After this stage, we created the new tags, inserted them to the sentences and renumbered the token indices, incoming and outgoing dependencies within the remaining of the sentence.
2. Enlarged Version (Ve): We needed to create the following two versions of the treebank during the evaluation of the automatic MWE extractors that will be introduced in the following section. In order to create version Ve,

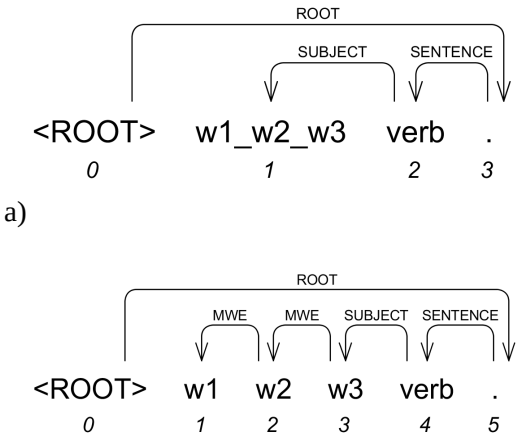

Figure 2: MWE representation in the Turkish Treebank *(picture drawn with MaltEval TreeViewer(Nilsson and Nivre, 2008))*

[4]Interested reader may refer to Eryigit et al. (2008) for detailed examples.

Version	Turkish Treebank *Vo*	*Vd*	*Ve*	*Ve-o*	Validation Set
# of sentences	5635	5635	5635	5635	300
# of tokens	65184	67803	63318	65887	4513
# of words inc. punctuations	53992	56424	52267	54642	3610
# of words exc. punctuations	43572	45999	41847	44217	3080
# of combined collocations	2040	0	3674	1697	89

Table 1: Data sets' statistics: Version (Vo - Version Original; Vd - Version detached; Ve - Version enlarged; Ve-o - Version enlarged excluding the combined collocations of Vo)

we first extracted a MWE list consisting the 30150 MWEs available in the Turkish Dictionary (TDK, 2011) and then automatically listed the entire treebank sentences where the lemmas of the co-occurring words could match[5] the lemmas of the MWE constituents in the list. We manually marked the sentences where the co-occurring words may be actually accepted as a MWE (but somehow missed during the construction of the treebanks) and then automatically combined these words into single units and renumbered the token indices and incoming and outgoing dependencies within the remaining of the sentence. By this process, 1697 new MWEs are added to the treebank (Table 1).

3. Version Ve-o: is the treebank version where only the MWEs coming from the dictionary are annotated over the detached version. In other words, the annotated MWEs in Ve-o is the relative complement of Vo in Ve.[6]

Table1 gives the statistics for all the versions and the validation set (Eryiğit, 2007). We see from the table that different versions change the number of tokens and dependencies that should be discovered by the parser; as an example Vd increases the dependency number that will be evaluated from 43572 to 45999. In Table 1, the difference between # of combined collocations 1697 and 3674-2040 are due to overlapping MWEs consisting 2 or more words.

In the Turkish Treebank, there exists a very small amount (54) of compound functional words which are combined into a single unit. These are mostly the conjunctions which has an extra -da/-de/-ki enclitics written on the right side of and separately from the conjunction they attach to. We see that since these head-initial dependencies do not obey the general head-final dependency tradition of the treebank, only this type of compound functional words are preferred to be combined into single units in the construction phase of the treebank. While creating the detached version, we only process these differently and detach them in a head-initial manner.[7] Figure 3 shows the detachment for the "ya da" (*or*) conjunction.

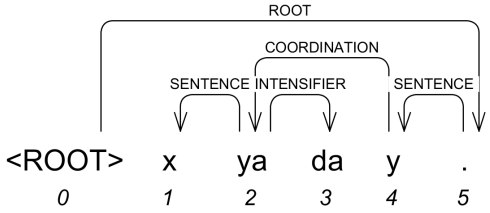

Figure 3: The detachment of Combined Functional Words

4.3 MWE Extractors

In order to understand the structure of the data in hand and the behavior of the current MWE extractors on it, we evaluated the success of two different MWE extractors. Table 2 gives the precision, recall and F values. All the experiments have been conducted on the detached version of the treebank and evaluated both on the original and enlarged versions.

[5]The lemmas of the words in the treebank are gold-standard. But in order to create the possible lemmas for the MWE constituents of the dictionary list, we used a morphological analyzer.

[6]Thus this version similarly to the version Vd, consists also a new dependency label "MWE".

[7]In our preliminary tests, we have detached these as regular MWEs and observed that the parser tends to find the actually correct dependencies but we penalize it unnecessarily.

The first MWE extractor is the rule based processor of Oflazer et al. (2004). For the second one, we developed a MWE checker by using the dictionary MWE list described in the previous section. Due to the inflectional structure of Turkish, MWE constituent words go under inflection as well. That's why we tried 3 different models while catching the MWEs in the treebank:

- Model 0: The co-occurring words in the treebank sentences are accepted as a MWE if and only if they have the exact surface forms with the MWE's constituents in the dictionary list.
- Model 1: Only the last constituent of the MWE is allowed to go under inflection (the ones at the beginning should have exactly the same surface form).
- Model 2: All of the constituents are allowed to be inflected.

		Ofl.(2004)	Mod.0	Mod.1	Mod.2
Vo	P	57.27	27.54	35.00	28.43
	R	30.93	12.22	46.29	49.48
	F	40.17	16.93	39.86	36.11
Ve	P	75.09	90.15	84.93	73.52
	R	22.54	22.24	62.45	71.15
	F	34.68	35.68	71.97	72.32

Table 2: Performance of MWE Extractors: Ofl.(2004):the MWE processor of Oflazer et al. (2004);Mod.x:Model x

We see from Table 2 that the results of the evaluation by using the version Vo is very low than expected; the highest F score that could be obtained with Ofl.(2004) is 40.17 and the dictionary list is 39.86. When we look at the cause, we notice that many of the compound verb and noun formations in the Turkish Dictionary are not marked in the original treebank and that causes very low precision scores (Mod.0 27.54). In order to alleviate this problem and with the hope to achieve better results in parsing accuracy, we created the version Ve of the treebank described in the previous section. By evaluating with this new version of the treebank, we see that the precision values are increased drastically for all of the models; Ofl.(2004) from 57.27 $\rightarrow$ 75.09, Mod.0 from 27.54 $\rightarrow$ 90.15. The recalls are still low for many of the models; Mod.2 with the highest recall value. The next section will search the answer of the question what will happen to the parsing performance if we develop a perfect MWE recognizer with an F score of 100%.

5 Experiments

We have four sets of experiments. Before introducing them, we will first explain our evaluation strategy.

5.1 Evaluation Metrics

We exactly follow the evaluation metrics used in Eryigit et al. (2008): We use ten-fold cross-validation in the experiments on the treebank (except the experiments on the validation set). We report the results as mean scores of the ten-fold cross-validation, with standard error. The main evaluation metrics that we use are the unlabeled attachment score (AS_U) and labeled attachment score (AS_L), namely, the proportion of IGs that are attached to the correct head (with the correct label for AS_L). A correct attachment is one in which the dependent IG (the last IG in the dependent word) is not only attached to the correct head word but also to the correct IG within the head word. We also report the (unlabeled) word-to-word score (WW_U), which only measures whether a dependent word is connected to (some IG in) the correct head word. We will refer to this metric (WW_U) especially while evaluating the dependencies between MWEs' constituents since we are automatically creating these dependencies (thus automatically selecting the IG to which the dependent will be connected). Where relevant, we also test the statistical significance of the results.

5.2 Evaluation Type

In order to see the impact of different approaches, we are making 3 different evaluations:

1. overall; AS_U, AS_L, WW_U scores provided
2. the dependencies with "MWE" labels (appearing after the detachment of MWE units Figure 2-b): AS_U, precision, recall, WW_U scores provided
3. the dependencies excluding the ones with "MWE" labels (the surrounding structure in the sentence): AS_U, AS_L, WW_U scores provided

5.3 Experiment Set I

In this set of experiments we first repeated the results of Eryigit at al. (2008) with the new Maltparser version. And then, tested the trained model on the different treebank versions that we had prepared. The first two lines of Table 3 gives the results reported in Eryigit at al. (2008); the second line is their results on the raw data where the pos tagging has been done by using an automatic analyzer. The fourth line gives our results when we test our parser on the detached version of the treebank. We see that our results on this version are better than the results obtained with raw data (73.3 - 74.7 AS_U) since we are using gold standard pos tags in order to be able to focus on the errors caused by the lack of MWE annotation. So, if we compare the tests on Vo with Vd, we see that the parser's performance drops from 76.1 to 74.7 (-1.4) in AS_U, 83.0 to 81.8 (-1.2) in WW_U and 67.4 to 63.3 (-4.1) in AS_L. We provide two values for the labeled accuracy on Vd: Since there is not any dependency with "MWE" label in the training model trained with the original treebank, it is impossible for the parser to assign correct labels to this kind of dependencies. If we accept all the labels assigned to these dependencies as correct than we will obtain a labeled accuracy of 66.5 given after the slash in the AS_L cell. And in this case, the drop in labeled accuracy would be 0.9. Of course this is a very optimistic evaluation but the real labeled accuracy should be something in between these two if the parsing model have already seen this dependency type.

tested on	AS_U	AS_L	WW_U
Ery.(2008) Vo	76.0±0.2	67.0±0.3	82.7±0.5
Ery.(2008) Raw Data	73.3±0.3	63.2±0.3	80.6±0.7
Vo	76.1±0.2	67.4±0.3	83.0±0.2
Vd	74.7±0.2	63.3/66.5±0.3	81.8±0.2
Ve	75.5±0.2	66.7±0.3	82.5±0.2
Ve-o	74,0±0,2	62,4/65.7±0,3	81,1±0,1

Table 3: The parser's performance trained on the original treebank (Vo)

The fifth line of the Table 3 gives the results on the enlarged version of the treebank. We see that although the results are higher than Vd, they are not as good as Vo. From here, we understand that by collapsing some words into single units, we disappeared some of the dependencies where the parser was already very successful at finding; the average scores were getting higher by the success coming from this type of dependencies. As a final step in this set of experiments, we tested our parser on version Ve-o and saw that the results are worse than the results on the detached version (Vd): Collecting the new MWE components into single units gives worse results than doing no MWE processing at all. But is this just an illusion[8] or is there really a bad effect on the discovery of the remaining dependencies also? Actually the results provided here is not enough for answering this question. In order to see the exact picture we should examine the results more closely (Section 5.5).

5.4 Experiment Set II

In this set of experiments, we are looking at the results by using the new treebank versions in the training stage as well. Table 4 shows that in all of the test set combinations, the worst results are obtained by training with the enlarged treebank (Ve).

train.	test.	AS_U	AS_L	WW_U
Vo	Ve	75.5±0.2	66.7±0.3	82.5±0.2
	Vo	**76.1**±0.2	**67.4**±0.3	**83.0**±0.2
	Vd	**74.7**±0.2	**63.3**±0.3	**81.8**±0.2
Vd	Ve	75.3±0.2	65.9±0.3	82.4±0.2
	Vo	**76.0**±0.2	**66.7**±0.3	**82.9**±0.1
	Vd	**76.0**±0.2	**65.9**±0.3	**82.7**±0.2
Ve	Ve	75.3±0.2	66.7±0.3	82.3±0.2
	Vo	75,7±0.2	67,1±0.3	82,7±0.2
	Vd	74.3±0.2	63.0±0.3	81.4±0.2

Table 4: Parser's performance by training and testing with the different versions of the treebank

Table 4 shows that the results on the detached test set (Vd) are better when trained by Vd (76.0±0.2 AS_U) rather than the original treebank Vo (74.7±0.2 AS_U). This means that the parser is better at find-

[8] caused by the removed dependencies after the combination of the MWE constituents into single units in Ve-o: if the parser was already highly successful at finding these, the combination operation would certainly give the effect of a success decrease in the average.

ing the dependencies if it has samples from the same genre. But again by just looking the results this way, it is not possible to understand the situation entirely.

5.5 Experiment Set III

In order to be able to understand what is happening on the dependencies within MWEs and their surrounding structures, we are evaluating the results on 3 different ways described in Section 5.2. Table 5 gives the scores in all of thees three evaluation types; The first column block states each training and testing combination together with the number of combined MWEs and the total number of dependencies in the test sets. The second column block gives the overall results of the parser. The fourth column block lists the results obtained on the dependencies (with MWE label) occurring between the constituent words which appeared after the detachment of the MWEs annotated on the original treebank. The third column block gives the results obtained on the remaining dependencies (excluding the ones with MWE label). Example: The number of MWE labeled dependencies is 0 for the Vo and 2427 for Vd meaning that the detachment of 2040 combined MWEs (two or more words) on Vo resulted to 2427 dependencies. Thus, the average number of dependency per MWE is 1.19.

From the Table 5, we may now analyze the parser's performance in detail; We observe that the parser's performance (trained on the original treebank) drops significantly when it is tested on the detached version both on the overall results (76.1 $\rightarrow$ 74.7 AS_U) and excluding MWEs (76.1 $\rightarrow$ 75.9 AS_U, 83.0 $\rightarrow$ 82.8 WW_U the difference is small but statistically significant (McNemar $p<0.01$)). We see that the tests on Ve-o not only causes a decrease on the overall accuracy (74.7 $\rightarrow$ 74.0 AS_U) but also a decrease on the dependencies excluding MWEs (75.9 $\rightarrow$ 74.7 AS_U). Thus, we may say that the combination of MWEs listed in the dictionary has a harming effect on the determination of other syntactic structures as well. On the other side, we observe that the combination of MWEs annotated in the original treebank has a positive effect on the determination of other syntactic structures (from Vd to Vo the AS_U differs 75.9 $\rightarrow$ 76.1 but from Vd to Ve-o the AS_U differs 75.9 $\rightarrow$ 74.7).

These results bring the question: "What is the difference between the dictionary MWE list and the treebank MWE list?" In order to answer this question, we manually classified the MWEs in the Turkish treebank into six categories which are listed in Table 6. The second column in the table lists the number of MWEs in each categories, the third column lists the number of dependencies when we detach these MWEs and the fourth column gives the WW_U scores on the dependencies from these specific MWE categories. We only look at the WW_U scores since the IG-based links are created automatically and WW_U scores is considered to be more informative. The results are obtained with a parsing model trained with Version Vd. (it is obvious that a model trained with Vo won't be able to find most of these dependencies because of the lack of samples.)

MWE type	#of MWEs	#of Dep.	WW_U
Named ent.	618	941	83.7
Num. exp.	98	123	82.1
Comp. func.	54	54	5.6
Dup.	206	206	66.5
Comp. vn.	1061	1103	93.0

Table 6: Parser's success on special MWE types: Named Entities (Named ent.), Numerical Expressions (Num.exp.), Compound function words (Comp.func.), Duplications (Dup), Compound verb and noun formations (Comp.vn.)

We see from the Table 6, the parser is very bad (5.6 WW_U) at determining the compound function words (which are very rare) (Figure 3) and duplications[9] (66.5 WW_U). These dependencies could actually be easily discovered by a rule-based extractor. The success on named entities (83.7) and number expressions (82.1) could be considered as good but one shouldn't forget that the training and testing data is from the same treebank and the sentences could actually not be considered as random. Thus on a totally unseen data, these results would be lower. But again we believe a rule-based extractor for numerical and date expressions could be developed with

[9] "These are partial or full duplications of the forms involved and can actually be viewed as morphological derivational processes mediated by reduplication across multiple tokens." Oflazer et al. (2004) Example: "uyur uyumaz" ((he) sleeps (he) doesn't sleep) the MWE meaning is "as soon as (he) sleep".

train on	test on	# of comb. MWEs	# of Dep.	Overall results			Results Excl. MWE type dependencies(# of Dep= 43572)			Results on MWE type dependencies				
				AS_U	AS_L	WW_U	AS_U	AS_L	WW_U	# of Dep.	AS_U	P	R	WW_U
Vo	Vo	2040	43572	76.1±0.2	67.4±0.3	83.0±0.2	n/c	n/c	n/c	0	n/a	n/a	n/a	n/a
	Vd	0	45999	74.7±0.2	63.3/66.5±0.3	81.8±0.2	75.9±0.2	67,2±0.3	82.8±0.1	2427	62.1	n/a	n/a	73.1
	Ve	3674	41847	75.5±0.2	66.7±0.3	82.5±0.2	n/c	n/c	n/c	0	n/a	n/a	n/a	n/a
	Ve-o	1697	44217	74,0±0,2	62,4/65.7±0,3	81,1±0,1	74.7±0.2	65.9±0.3	81.7±0.2	2369	60.7	n/a	n/a	71.9
	S1	976	44675	75,6±0,2	65,4/67.2±0,3	82,8±0,1	76.0±0.2	67.3±0.3	83.0±0.2	1103	72.1	n/a	n/a	91.4
	S2	770	44881	75,4±0,2	65,1/67.0±0,3	82,6±0,1	76.0±0.2	67.3±0.3	82.9±0.2	1309	67.2	n/a	n/a	84.2
	S3	716	44935	75,2±0,2	64,9/66.9±0,3	82,4±0,1	75.9±0.2	67.2±0.3	82.8±0.1	1363	64.6	n/a	n/a	81.0
Vd	Vo	2040	43572	76.0±0.2	66.7±0.3	82.9±0.1	n/c	n/c	n/c	0	n/a	n/a	n/a	n/a
	Vd	0	45999	76.0±0.2	65.9/68.0±0.3	82.7±0.2	76.0±0.2	66.6±0.3	82.8±0.2	2427	81.6	71.3	57.2	86.2
	Ve	3674	41847	75.3±0.2	65.9±0.3	82.4±0.2	n/c	n/c	n/c	0	n/a	n/a	n/a	n/a
	Ve-o	1697	44217	75.2±0.2	65.1/67.0±0.3	82.0±0.2	74.9±0.2	65.7±0.3	81.9±0.2	2369	79.9	73.4	55.4	84.6
	S1	976	44675	**76.0**±0.2	66.2/67.8±0.3	**82.9**±0.2	**76.0**±0.2	66.6±0.3	**82.9**±0.2	1103	**85.6**	51.9	53.3	**93.0**
	S2	770	44881	75.9±0.2	66.1/67.8±0.3	82.8±0.2	76.0±0.2	66.7±0.3	82.9±0.2	1309	82.1	55.7	52.2	88.9
	S3	716	44935	75.8±0.2	65.9/67.7±0.3	82.6±0.1	76.0±0.2	66.6±0.3	82.9±0.2	1363	79.0	54.0	50.1	85.5

Table 5: Overall Parsing Results (on and outside MWEs) with different treebank versions:
n/c:no change with the previous results on the left column block (overall results); n/a:not available

high success. The best performance (93.0) of the parser is on the MWE of types compound verb and noun formations. We see that this is the class with the highest number of MWEs. And when we look at the dependencies on the dictionary list, we see that almost all of the MWEs are from this class; i.e. compound verb and noun formations. So actually by creating the enlarged version (Ve) of the treebank, we added 1697 more MWEs into this category where the parser is already very good at discovering. So what is going wrong when we combine these MWEs into single units in the preprocessing step? Instead of having an accuracy of 93% with automatic detection, we would have an accuracy of 100% with our deterministic approach. The problem is that when we do this we actually increase the lexical sparsity. For example, the verb "etmek" (to do) in Turkish may produce many MWEs such as "ikaz etmek" (to warn), "buyur etmek" (to welcome), "fark etmek" (to notice) and so on. And this is a very frequent verb in Turkish; it occurs 388 times in the current Turkish Treebank where in 340 of them it formed a MWE (already annotated in the treebank). With the addition of the dictionary list, 27 more MWEs constructed with "etmek" is added to the treebank. When we combine the MWEs constructed with this verb into single units we are actually splitting its frequency to many rarely occurring MWEs.

The next question is "If the addition of more MWEs of type compound verb and noun formations caused an accuracy decrease in parsing performance, could we obtain better or similar results by detaching the MWEs from this type and leaving the others as in the original?" If the answer is yes, then we could develop type specific MWE extractors according to the results. To answer our new question, we tested on 3 new versions of the treebank which consists the following subsets of MWEs:

- Subset 1 (S1)- Vo excluding MWEs of type compound verb and noun formations (meaning that only this type of MWEs are detached into their constituents and the others are left combined.)
- Subset 2 (S2)- S1 excluding MWEs of type duplications.
- Subset 3 (S3)- S2 excluding MWEs type compound function words.

Table 5 gives the results for the subsets as well. We see from the table that S1 has 1103 dependencies with MWE label (of type compound verb and noun formations). The unlabeled results of the tests on S1 are better than the tests on Vd, Ve, Ve-o and very close to the results on Vo. This means that, a MWE extractor concentrating only on the MWEs of type named entities, numerical expressions, duplications and some compound function words already annotated in the treebank could obtain similar results to the scores on the original treebank. But we see that we still have a problem with labeled accuracies. The addition of the new label increased the complexity for the parser and caused false positive assignments on dependencies from other types.

5.6 Experiment Set IV

To alleviate the problem observed in the labeled accuracy, instead of assigning the new "MWE" label to the detached MWEs, we developped a rule based dependency label chooser which assigns an appropriate label to these dependencies obeying the Turkish Treebank annotation approach. We have 16 rules similar to the following one:
if DEPENDENCY LABEL eq 'MWE'{
if HEAD's POSTAG eq 'Verb'
&& DEPENDENT's POSTAG eq 'Adverb'
then change MWE→ MODIFIER
}
We changed the MWE labels in S1 and Vd by using this rule based dependency label chooser. Table 7 gives the results at the end of this operation.

train.	test.	AS_U	AS_L	WW_U
Vo	Vo	76.1±0.2	67.4±0.3	83.0±0.2
	Vd*	74.7±0.2	66.1±0.2	81.8±0.2
S1*	S1*	76.1±0.2	67.6±0.3	82.9±0.2
	Vd*	75.3±0.2	66.7±0.2	81.9±0.2

Table 7: Parsing results with MWE labels replaced by the label chooser

The results are as we expected. In the training stage, if we use our new version of the treebank (S1*) (where we detached the MWEs of type compound noun and verb formations) instead of the original one, the results on raw data (Vd*) (2nd and 4th lines of Table 7) became significantly better (AS_U and AS_L difference is statistically signif-

icant with McNemar ($p < 0.01$)). We also validated this outcome by testing on the validation set. Table 8 gives the results on both the original version of the validation set (ITU-Vo) and the detached version (ITU-Vd*). Although the small number of available MWEs (only 89), we observe an improvement by using the model trained with S1*. For the first two lines the improvement is rather small but statistically significant on labeled accuracy with Mcnemar($p<0.05$).

train.	test.	AS_U	AS_L	WW_U
Vo	ITU-Vo	80.42	72.47	84.55
S1*	ITU-Vo	81.01	73.25	84.68
Vo	ITU-Vd*	80.02	72.21	84.13
S1*	ITU-Vd*	80.65	72.94	84.32

Table 8: Results on Validation Set
(The trained model is on the data of 9 cross validation fold; the training set size is the same with other experiments)

Another outcome that could be observed from Table 7 (by comparing the first and third lines of results) is that a MWE extractor for only MWEs of types named entities, duplications, numbers, dates and some predefined list of compound prepositions would be enough for obtaining the results of the gold-standard treebank (there is no statistically significant results between these two lines). As a final comment, we may conclude that the preprocessing of the test data would improve the results by nearly 1.5 in IG-based evaluations (74.7→76.1, 66.1→67.6) and 1.1 (81.8→82.9) in word-based evaluation (2nd and 3rd lines of Table 7) if we also train with S1* instead of the original treebank. One should remember that there are in total 1324 fewer dependencies (Table 5) in S1 compared to Vd. These dependencies are expected to be discovered by the MWE extractor.

6 Conclusion

In this paper, we made a detailed analysis on multiword expression extraction on parsing accuracy of a statistical dependency parser. Our results showed that different MWE types have different impacts on the parser's performance. During the experiments, for the representation of MWEs in parsing data, we used a highly adopted strategy and combined the MWEs' constituents into single units. But we observed that when this approach is applied to the MWE types that could already be determined by the parser with a high success, the overall performance is decreased instead of increasing. The reason is mostly due to the lexical sparsity caused by the representation of the MWEs (as a single unit).

Although the development of a high accuracy MWE extractor was out of scope of this paper, during the analysis of different MWE types, we observed that most of them (which helped to increase parsing performance) could be easily found by creating rule-based MWE extractors. As the future work, we plan to develop such an extractor and evaluate the real parsing performance by using it. Another research topic will certainly be to investigate different MWE representations (others than the combination strategy).

Acknowledgments

The authors want to thank Mehmet Altıparmak for his help during the creation of the new treebank versions.

References

Sabine Buchholz and Erwin Marsi. 2006. Conll-X shared task on multilingual dependency parsing. In *Proceedings of the 10th Conference on Computational Natural Language Learning*, pages 149–164, New York, NY. Association for Computational Linguistics.

Conor Cafferkey. 2008. Exploiting multi-word units in statistical parsing and generation. Master's thesis, Dublin City University, Ireland.

Gülşen Eryiğit. 2007. ITU validation set for Metu-Sabancı Turkish treebank. March.

Gülşen Eryiğit, Joakim Nivre, and Kemal Oflazer. 2008. Dependency parsing of Turkish. *Computational Linguistics*, 34(3):357–389.

Deirdre Hogan, Jennifer Foster, and Josef van Genabith. 2011. Decreasing lexical data sparsity in statistical syntactic parsing - experiments with named entities. In *Proceedings of the Workshop on Multiword Expressions: from Parsing and Generation to the Real World*, pages 14–19, Portland, Oregon, USA, June. Association for Computational Linguistics.

Valia Kordoni, Carlos Ramisch, and Aline Villavicencio, editors. 2011. *Proceedings of the Workshop on Multiword Expressions: from Parsing and Generation to*

the Real World. Association for Computational Linguistics, Portland, Oregon, USA, June.

Ioannis Korkontzelos and Suresh Manandhar. 2010. Can recognising multiword expressions improve shallow parsing? In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 636–644, Los Angeles, California, June. Association for Computational Linguistics.

Jens Nilsson and Joakim Nivre. 2008. Malteval: An evaluation and visualization tool for dependency parsing. In *In Proceedings of the Sixth International Language Resources and Evaluation. LREC*.

Joakim Nivre and Jens Nilsson. 2004. Multiword units in syntactic parsing. In *MEMURA 2004 - Methodologies and Evaluation of Multiword Units in Real-World Applications, Workshop at LREC 2004*, pages 39–46, Lisbon, Portugal, May. In Dias, G., Lopes, J. G. P. and Vintar, S. (eds.).

Joakim Nivre, Johan Hall, Jens Nilsson, Gülşen Eryiğit, and Stetoslav Marinov. 2006. Labeled pseudo-projective dependency parsing with support vector machines. In *Proceedings of the 10th Conference on Computational Natural Language Learning*, pages 221–225, New York, NY.

Joakim Nivre, Johan Hall, Jens Nilsson, Atanas Chanev, Gülşen Eryiğit, Sandra Kübler, Stetoslav Marinov, and Erwin Marsi. 2007. Maltparser: A language-independent system for data-driven dependency parsing. *Natural Language Engineering Journal*, 13(2):99–135.

Kemal Oflazer, Bilge Say, Dilek Z. Hakkani-Tür, and Gökhan Tür. 2003. Building a Turkish treebank. In A. Abeillé, editor, *Treebanks: Building and Using Parsed Corpora*, pages 261–277. Kluwer, London.

Kemal Oflazer, Özlem Çetinoğlu, and Bilge Say. 2004. Integrating morphology with multi-word expression processing in Turkish. In Takaaki Tanaka, Aline Villavicencio, Francis Bond, and Anna Korhonen, editors, *Second ACL Workshop on Multiword Expressions: Integrating Processing*, pages 64–71, Barcelona, Spain, July. Association for Computational Linguistics.

Kemal Oflazer. 1994. Two-level description of Turkish morphology. *Literary and Linguistic Computing*, 9(2):137–148.

TDK. 2011. Turkish Language Association Turkish dictionary. `http://www.tdk.gov.tr`.