From Treebank Conversion to Automatic Dependency Parsing for Vietnamese

Dat Quoc Nguyen[1], Dai Quoc Nguyen[1], Son Bao Pham[1], Phuong-Thai Nguyen[1], and Minh Le Nguyen[2]

[1] Faculty of Information Technology
University of Engineering and Technology
Vietnam National University, Hanoi
{datnq, dainq, sonpb, thainp}@vnu.edu.vn
[2] School of Information Science
Japan Advanced Institute of Science and Technology
nguyenml@jaist.ac.jp

Abstract This paper presents a new conversion method to automatically transform a constituent-based Vietnamese Treebank into dependency trees. On a dependency Treebank created according to our new approach, we examine two state-of-the-art dependency parsers: the MSTParser and the MaltParser. Experiments show that the MSTParser outperforms the MaltParser. To the best of our knowledge, we report the highest performances published to date in the task of dependency parsing for Vietnamese. Particularly, on gold standard POS tags, we get an unlabeled attachment score of 79.08% and a labeled attachment score of 71.66%.

1 Introduction

Dependency parsing is one of the major research topics in natural language processing (NLP) as dependency-based syntactic representations are useful for many NLP applications such as machine translation and information extraction [1]. This research field was boosted by the successes of the CoNLL shared tasks on multilingual dependency parsing [2,3] where raising current state-of-the-art approaches based on supervised data-driven machine learning. McDonald and Nivre [4] has determined two major categories of data-driven dependency parsing: graph-based approaches [5,6,7,8] and transition-based ones [9,10,11]. In addition, there are hybrid methods [12,13] to combine the graph-based and transition-based approaches. However, those methods require such large training corpora as dependency Treebanks which are very expensive: taking a lot of time and human effort to manually annotate the corpora.

In many languages like Vietnamese, there are no manually labeled dependency Treebanks available. Since constituent structure-based Treebanks, for instances the English Penn Treebank [14] and the Vietnamese Treebank [15], are dominant resources to develop natural language parsers, constituent-to-dependency conversion approaches must be applied to generate larger amounts of annotated dependency structure-based corpora. Johansson and Nugues [16] proposed an extended conversion procedure for English to overcome drawbacks of previous methods [9,17] on new versions of the English Penn Treebank. The Treebank transformation procedures in such other languages

as German, French, Spanish, Bulgarian, Chinese and Korean can be correspondingly found in [18], [19], [20], [21], [22] and [23].

Turning to Vietnamese, there are only two works [24,25] on dependency parsing. Hong et al. [24] described an approach for Vietnamese dependency parsing based on Lexicalized Tree-Adjoining Grammars (LTAG). A LTAG parser was trained on set of 8367 constituent trees in the Vietnamese Treebank. Then dependency relations were extracted from derivation trees returned by LTAG parsing. Evaluating on 441 sentences of 30-words length or less, the Hong et al. [24] 's method obtained an unlabeled attachment score[3] of 73.21% on automatically assigned part-of-speech (POS) tags.

Thi et al. [25] presented a constituent-to-dependency transformation method for Vietnamese. The method applied head-percolation rules constructed for Vietnamese as exhibited in [26] to find the head of each constituent phrase. However, it is not clearly how dependency labels are inferred since Thi et al. [25] just outlined that there was a function namely GetDependentLabel exploited to label dependency relations. On a Stanford format-based dependency Treebank of 10k sentences converted from the Vietnamese Treebank [15] according to their own procedure, Thi et al. [25] showed good experimental results on 10-fold cross validation evaluation scheme. They earned 73.03% and 66.35% computed for the unlabeled and labeled attachment scores given by the transition-based MaltParser toolkit [11] using gold standard POS tags.

The difference between our work and the two previous works on Vietnamese dependency parsing is that we make a better use of existing information in the Vietnamese Treebank. The previous works performed shallow processes as they do not take grammatical function tags into account as well as do not handle cases of coordination and empty category mappings. To sum up, the contributions of our study are:

• We propose a new constituent-to-dependency conversion method to automatically create a dependency Treebank from the input constituent-based Vietnamese Treebank. Specifically, in addition to modifications of head-percolation rules, we bring clear heuristics to label dependency relations employing grammatical function tags and other existing information in the Vietnamese Treebank. We also solve the cases of coordination and empty categories.

• We provide[4] a Vietnamese dependency Treebank namely VnDT containing 10200 sentences. The VnDT Treebank is formatted following 10-column data format as proposed by the CoNLL shared tasks on multilingual dependency parsing [2,3]. It is because most state-of-the-art dependency parsers such as the graph-based MSTParser [5] and the MaltParser refer to the CoNLL 10-column format as an input standard.

• We achieve highest performances published to date in the task of Vietnamese dependency parsing by examining the MSTParser and the MaltParser on our VnDT Treebank. In particular on the evaluation scheme of 10-fold cross validation, we gain the unlabeled attachment score (UAS) of 79.08% and the labeled attachment score (LAS) of 71.66% on gold standard POS tags. Besides, the highest parsing scores are 76.21% for UAS and 66.95% for LAS in terms of automatically assigned POS tags.

[3] Un-analyzable sentences and punctuations are not taken into account.

[4] Our Vietnamese dependency parsing resources including preprocessing tools, pre-trained models and the VnDT Treebank are available at http://vndp.sourceforge.net/

2 Our new Treebank conversion approach for Vietnamese

This section is to introduce our new procedure to automatically convert constituent trees to dependency trees for Vietnamese. We provide information about the Vietnamese constituent trees in section 2.1 and technique to transform the constituent trees to unlabeled dependency trees in section 2.2. We then describe how our method labels dependency links in the use of function tags in section 2.3 and heuristics to infer suitable labels in section 2.4. The processes of solving cases of coordination and empty category mappings are detailed in sections 2.5 and 2.6, respectively.

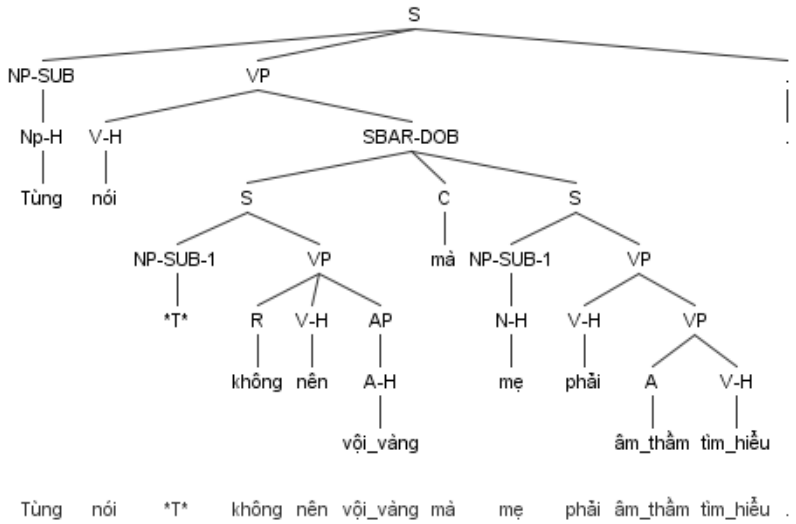

Figure 1. A tree in the Vietnamese Treebank "*Tùng nói* $^*T^*$ *không nên vội_vàng mà mẹ phải âm_thầm tìm_hiểu* ." (*Tung suggests that* $^*T^*$ *should not be hastily but mother must be silent to find out* .)

2.1 Constituent trees in the Vietnamese Treebank

The Vietnamese Treebank [15] has been produced as a part of the national project VLSP[5]. It contains about 10200 constituent trees (220k words) formatted similarly as those trees in the Penn Treebank [14]. In figure 1 illustrating a constituent tree of a Vietnamese sentence, the tree includes Vietnamese words at leaf nodes[6], and POS tag-level and phrase-level nodes as explained in table 1. Table 1 also gives information about grammatical function tags associated to POS and phrase-level tags at non-leaf nodes. These tags will be used in our transformation approach to label dependency relations.

[5] http://vlsp.vietlp.org:8080/

[6] Vietnamese is a monosyllabic language; hence, a word may consist of more than one token. Tokens in a word can be distinguished by underline character _. The $^*T^*$ at a leaf node in figure 1 means an empty category.

Table 1. Part-Of-Speech (POS) tags, phrase-level and function tags existing in the Vietnamese Treebank. In addition to LBKT and RBKT (left/right bracket) POS tags, there are such different punctuation types as **.,;:-''/** and **...**

POS tags			
N Noun	Nb Borrowed noun	R Adjunct	I Exclamation
Np Proper noun	V Verb	L Determiner	T Particle
Nc Classifier noun	Vb Borrowed verb	M Quantity	Y Abbreviation
Nu Unit noun	A Adjective	E Preposition	S Affix
Ny Abbreviated noun	P Pronoun	C Conjunction	X Un-definition/Other

Phrase-level tags				Function tags			
S	Sentence	UCP	Unlike coordinated phrase	H	Head	TMP	Temporal
SQ	Question	YP	Abbreviation phrase	SUB	Subject	LOC	Location
SBAR	Subordinate clause	WHNP	Wh-noun phrase	DOB	Direct object	DIR	Direction
NP	Noun phrase	WHAP	Wh-adjective phrase	IOB	Indirect object	MNR	Manner
VP	Verb phrase	WHRP	Wh-adjunct phrase	TPC	Topicalization	PRP	Purpose
AP	Adjective phrase	WHPP	Wh-prepositional phrase	PRD	Predicate	CND	Condition
RP	Adjunct phrase	WHVP	Wh-verb phrase	EXT	Extent	CNC	Concession
PP	Prepositional phrase	WHXP	Wh-undefined phrase	VOC	Vocatives	ADV	Adverbial
QP	Quantity phrase	XP	Un-definition/other phrase				
MDP	Modal phrase						

2.2 Head-percolation rules

Finding the head of each phrase is an essential task in order to generate dependency links. Similar to a common manner to find the head in a phrase structure [9,17], our method is based on a classical technique of exploiting head-percolation rules (head rules). Following [25], we employ the head rules built for Vietnamese as shown in [26].

There are around 2200 unindexed empty categories such as *(NP-SUB *T*)*, *(NP-SUB *E*)* and *(V-H *E*)* appearing in the Vietnamese Treebank. For those unindexed phrases, it is unable to retrieve the corresponding phrases in an empty category mapping process. It leaded to a removal of those phrases from the Vietnamese Treebank. Therefore, we made minor changes on some existing head rules to adapt to the modified Vietnamese Treebank. For example, we changed the rule for VP by adding SBAR, R, RP and PP.

In table 2 presenting the used head rules: the first column denotes the phrase types, the second one indicates a search direction (l or r expressing a looking for leftmost or rightmost constituent respectively), and the third is a left-driven priority list of phrase-level and POS tags to look for (**.*** meaning any tag while **;** be a delimiter between tags). For example, to determine the head of a S phrase node, we look from left to right for any its child node associated to H function tag. If no child node H is found, we find for any child node with a S tag, and so on.

Using the head rules, it is straightforward to create unlabeled dependency trees from constituent trees: (i) marking the head of each phrase structure utilizing its head rule, and (ii) making dependents on the head for other child nodes in the phrase. For instance, a dependency tree consisting of unlabeled relation links will be returned as demonstrated in figure 2 for the input constituent tree in figure 1.

Table 2. Head-percolation rules for Vietnamese

S	l	*-H;S;VP;AP;NP;.*
SBAR	l	*-H;SBAR;S;VP;AP;NP;.*
SQ	l	*-H;SQ;VP;AP;NP;WHPP;.*
NP	l	*-H;NP;Nc;Nu;Np;N;P;VP;.*
VP	l	*-H;VP;V;A;AP;N;NP;SBAR;S;R;RP;PP;.*
AP	l	*-H;AP;A;N;S;.*
RP	r	*-H;RP;R;T;NP;.*
PP	l	*-H;PP;E;VP;SBAR;AP;QP;.*
QP	l	*-H;QP;M;.*
XP	l	*-H;XP;X;.*
YP	l	*-H;YP;Y;.*
MDP	l	*-H;MDP;T;I;A;P;R;X;.*
WHNP	l	*-H;WHNP;NP;Nc;Nu;Np;N;P;.*
WHAP	l	*-H;WHAP;A;N;V;P;X;.*
WHRP	l	*-H;WHRP;P;E;T;X;.*
WHPP	l	*-H;WHPP;E;P;X;.*
WHXP	l	*-H;XP;X;.*
WHVP	l	*-H;V;.*
UCP	l	*-H;.*

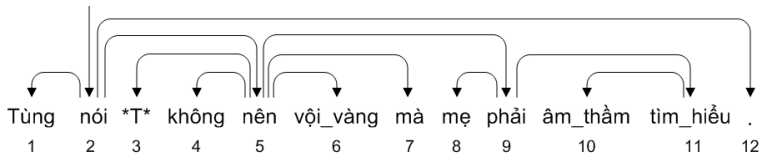

Figure 2. An unlabeled dependency tree converted from the tree in figure 1 in using the head rules for Vietnamese.

2.3 Grammatical dependency labels

We exploit all grammatical function tags as dependency labels excluding the tag *H* as *H* is employed in head-percolation rules to determine the head of every phrase. Some of the function tags may be combined[7] together like *TMP-TPC*. However, as pointed out by Choi and Palmer [27], most statistical dependency parsers do not often detect joined tags precisely. Hence, our conversion procedure keeps only the first function one in the pair of combined tags. Taking *TMP-TPC* as an example, the tag *TMP* will be selected as the dependency label instead of the joined tag *TMP-TPC*.

2.4 Inferred labels

Most of dependency links (arcs) in converted trees have no label. In order to label those links, we use heuristic rules as detailed in our algorithm 1.

[7] There are about 200 joined-tags pairs in the Vietnamese Treebank.

Algorithm 1: Rules to label dependency arcs

Data: Let c be a word while C is the highest node for which c is the head of. And P is the parent of C with the word p be the head of P.
Result: A dependency label for the arc $c \longleftarrow p$
if *C is the root node* **then return** ROOT;
else if *C is X, XP or WHXP* **then**
  if *C has a function tag of non-H* **then return** X + the tag;// There are XADV, XLOC, XMDP, XTMP, XPRD and XMNR.
  else return X;
else if *C has a function tag of non-H* **then** // Section 2.3: 15 grammatical function tags as dependency labels
  return the function tag;
else if *c is a determiner* **then return** DET;
else if *c is a punctuation* **then return** PUNCT;
else if *P is VP, and C is E, R, RP or WHRP* **then return** ADV;
else if *P is VP or WHVP* **then return** VMOD;// Verb modifier
else if *P is AP or WHAP* **then return** AMOD;// Adjective modifier
else if *P is NP or WHNP* **then return** NMOD;// Noun modifier
else if *P is PP, and C is NP* **then return** POB;// Object of a preposition
else if *P is PP or WHPP* **then return** PMOD;// Prepositional modifier
else return *DEP*;// Default label

Figure 3 displays a dependency tree with labels for which it is transformed from the constituent tree in figure 1 in the use of the head rules and algorithm 1.

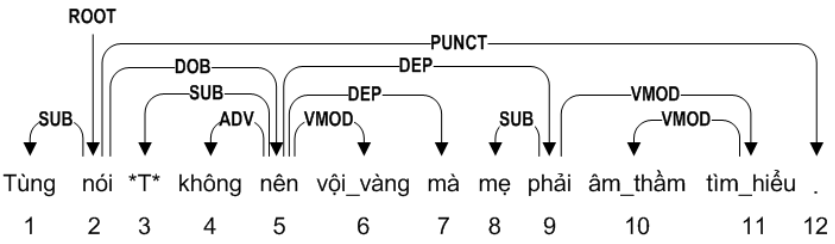

Figure 3. A labeled dependency tree transformed from the tree in figure 1 in utilizing the head rules and the algorithm 1.

2.5 Coordination

Our new procedure refers to a phrase as a coordinated one[8] if: (i) the phrase contains at least a child labeled with a C tag (i.e. conjunction), and (ii) the heads of both left and the right conjuncts (i.e. the left and right siblings of the C node) have the same

[8] The commas and semicolons are considered as separators within a coordinated phrase. Due to the Vietnamese Treebank annotation guideline: the UCP phrase always has at least a C-tag child. There are just 20 UCP-tagged phrases in the Vietnamese Treebank.

POS type. For instance in figure 1, we have the node $SBAR$ be a coordination as it has child node C corresponding to the word "mà$_{but}$", and the heads of two left and right conjuncts S are verbs "nên$_{should}$" and "phải$_{must}$".

There are several ways to represent coordinations in dependency structure [28,29]. Because we aim to generate the corpus of dependency trees in the CoNLL 10-column format, we follow the CoNLL dependency approach [2,3] to use dependency labels COORD and CONJ. Our method treats each preceding conjunct or conjunction to be the head of its following conjunct or conjunction. Figure 4 shows a dependency tree with a coordination example associated to the tree in figure 1.

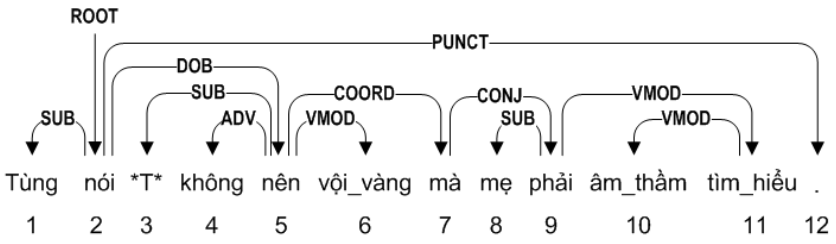

Figure 4. A coordination example.

2.6 Empty categories

There are two types of empty categories in the Vietnamese Treebank including $^*T^*$ (trace) and $^*E^*$ (expletive). Because all $^*E^*$ phrases and some of $^*T^*$ phrases have no associated indexes to map, thus, we removed those ones as mentioned in section 2.2. Turning to the remaining $^*T^*$ indexed empty categories , our approach to map those $^*T^*$ phrases is similar to the conversion method for English as described in [16]: relinking the heads of the phrases which are referred to by the corresponding indexes.

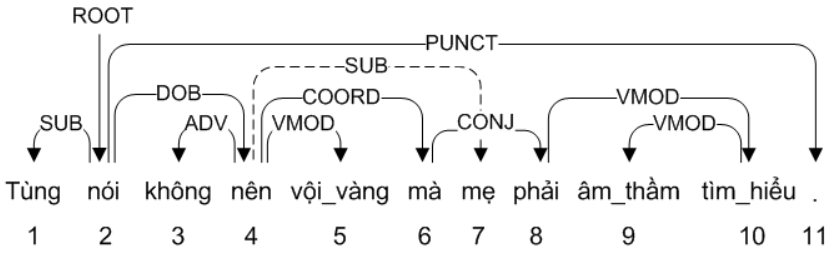

Figure 5. The final output dependency tree.

For example, from the trees in figures 1 and 4, we relink the head "mẹ$_{mother}$" (#8) of the phrase (NP-SUB-1 (N-H mẹ)) to be a dependent of the word "nên$_{should}$" (#5) which the $^*T^*$ (#3) depends on. The dependency label for the $^*T^*$ (#3) will be the dependency label for the word "mẹ$_{mother}$" (#8) in this empty category mapping process. The final

output dependency tree converted from the constituent tree in figure 1 according to our new transformation method is illustrated in figure 5 with the corresponding CoNLL 10-column format displayed in table 3. The relinking process creates some *non-projective* dependency trees (consisting of crossing links). Figure 5 presents an example of a non-projective tree.

Table 3. CoNLL format associated to the tree in figure 5.

1	Tùng	_	N	Np	_	2	SUB	_	_
2	nói	_	V	V	_	0	ROOT	_	_
3	không	_	R	R	_	4	ADV	_	_
4	nên	_	V	V	_	2	DOB	_	_
5	vội_vàng	_	A	A	_	4	VMOD	_	_
6	mà	_	C	C	_	4	COORD	_	_
7	mẹ	_	N	N	_	4	SUB	_	_
8	phải	_	V	V	_	6	CONJ	_	_
9	âm_thầm	_	A	A	_	10	VMOD	_	_
10	tìm_hiểu	_	V	V	_	8	VMOD	_	_
11	.	_	.	.	_	2	PUNCT	_	_

3 Experiments on dependency parsing for Vietnamese

3.1 Experimental setup

Data set. We conducted experiments of Vietnamese dependency parsing on our VnDT Treebank consisting of 10200 trees (219k words). The VnDT Treebank is automatically converted from the input Vietnamese Treebank [15] based on our new conversion approach. The VnDT schema contains 33 dependency labels as mentioned in the previous section. Table 4 shows the distributions of the labels in the VnDT Treebank. The proportion of non-projective trees in VnDT is 4.49% while it is 80% accounted for the percentage of sentences of 30-words length or less.

Table 4. Distributions of dependency labels (in %): $X.^*$ means any dependency label starting with X while $.^*OB$ denotes any label ending by OB including DOB, IOB, and POB. *O.F.Tags* refers to other grammatical function tags as dependency labels.

ROOT	4.66	DET	6.19
VMOD	14.82	PUNCT	13.95
NMOD	19.01	DEP	3.13
AMOD	2.35	$X.^*$	0.28
PMOD	0.24	$.^*OB$	11.89
COORD	1.88	ADV	5.93
CONJ	1.86	SUB	6.78
O.F.Tags	7.03		

Evaluation scheme. Results are evaluated on 10-fold cross validation scheme. We randomly separate the VnDT Treebank into 10 folds, giving one fold size of 1020 sentences. This evaluation procedure is repeated 10 times where each fold is used as the test set, and 9 remaining folds are merged as the training set. All our experimental results are reported as the average performances over the test folds.

Dependency parsing toolkits. We trained and evaluated two state-of-the-art dependency parsers: the graph-based MSTParser[9] [5,6] and the transition-based MaltParser [11]. As the MaltParser only produces projective dependency trees, we utilized the MaltOptimizer [30] - an optimization tool designed for the MaltParser to provide suitable feature model and parameters, and to select best parsing algorithms including non-projective ones.

Metrics. Accuracy metrics[10] [3] are the unlabeled attachment score (UAS) and the labeled attachment score (LAS). UAS: The percentage of words that are correctly assigned the head (or no head if the word is a root). LAS: The percentage of words that are correctly assigned the head and dependency label (or no head if the word is a root).

3.2 Accuracy results

Accuracies on gold standard POS tags. Table 5 gives accuracies gained by the two parsers on gold standard POS tags, where the results computed on the set of short sentences (30-words length or less) and on the remaining longer sentences are also provided. It is clearly that the MSTParser surpasses the MaltParser. On UAS scores, the MSTParser obtains an accuracy of 79.08% which is 1.71% higher than the performance at 77.37% produced by the MaltParser. For LAS scores, the MSTParser does better with the result of 71.66% than the MaltParser returning a score of 70.49%.

Table 5. Accuracy results (%) on gold standard POS tags.

	MST		Malt	
Length	UAS	LAS	UAS	LAS
<= 30 words	80.89	73.48	79.28	72.38
> 30 words	76.19	68.74	74.31	67.47
All	**79.08**	**71.66**	*77.37*	*70.49*

Because we have different set of dependency labels in comparison to the Thi et al. [25]'s dependency Treebank, it is not suitable in order to directly compare our method with the Thi et al. [25]'s one. However, on the same 10-fold cross validation scheme with the similar sizes of dependency corpora which are both transformed from the same Vietnamese Treebank, we reach higher performances of $4\%^{+}$ improvements given by the MaltParser for which the Thi et al. [25]'s approach achieved the UAS of 73.03% and the LAS of 66.35%.

[9] The MSTParser is used with default parameter settings associated to "decode-type" of "non-proj".

[10] Accuracy results are calculated without scoring on punctuations.

Accuracies on automatically POS tagging. We also carried out the experiments on automatically assigned POS tags. We adapted the RDRPOSTagger toolkit[11] [31,32] to perform Vietnamese POS tagging with an accuracy result of 94.61% on the set of raw word-segmented sentences extracted from the VnDT Treebank.

The UAS and LAS results are presented in table 6. The highest scores are returned by the MSTParser with the UAS score of 76.21% and the LAS score of 66.95%.

Table 6. Accuracies (%) on automatically assigned POS tags.

	MST		Malt	
Length	UAS	LAS	UAS	LAS
<= 30 words	77.85	68.60	76.30	67.52
> 30 words	73.59	64.31	71.66	62.97
All	**76.21**	**66.95**	*74.52*	*65.77*

Turning to the short sentences, we obtain the greatest UAS accuracy of 77.85% which is 4.64% higher than the UAS result of 73.21% reported by the Hong et al. [24]'s method evaluated on the 441 sentences of 30-words length or less. Though it is not on the same evaluation scheme, we show very promising results in the task of Vietnamese dependency parsing.

4 Conclusion

In this paper, we describe a new constituent-to-dependency conversion approach to automatically transform the Vietnamese Treebank [15] to dependency trees. Our procedure brings a better use of existing information in the Vietnamese Treebank.

We provide our Vietnamese dependency Treebank VnDT of 10200 sentences formatted following the CoNLL 10-column standard, and examine two state-of-the-art parsers the MSTParser and the MaltParser on the VnDT Treebank. Experiments point out that the MSTParser performs better than the MaltParser in Vietnamese dependency parsing task. To the best of our knowledge, we earn highest accuracy results published to date. For gold standard POS tags, we get the UAS score of 79.08% and the LAS score of 71.66%. On automatically assigned POS labels, the scores are 76.21% and 66.95% for the UAS and the LAS, respectively.

Acknowledgment. This work is partially supported by the Research Grant from Vietnam National University, Hanoi No. QG.14.04.

References

1. Kübler, S., McDonald, R., Nivre, J.: Dependency Parsing. Synthesis Lectures on Human Language Technologies, Morgan & cLaypool publishers (2009)

[11] http://rdrpostagger.sourceforge.net/

2. Buchholz, S., Marsi, E.: CoNLL-X shared task on multilingual dependency parsing. In: Proceedings of the Tenth Conference on Computational Natural Language Learning, CoNLL-X. (2006) 149–164
3. Nivre, J., Hall, J., Kübler, S., McDonald, R., Nilsson, J., Riedel, S., Yuret, D.: The CoNLL 2007 Shared Task on Dependency Parsing. In: Proceedings of the CoNLL Shared Task Session of EMNLP-CoNLL 2007. (2007) 915–932
4. McDonald, R., Nivre, J.: Characterizing the Errors of Data-Driven Dependency Parsing Models. In: Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL). Number June (2007) 122–131
5. McDonald, R., Pereira, F., Ribarov, K., Hajič, J.: Non-projective Dependency Parsing Using Spanning Tree Algorithms. In: Proceedings of the Conference on Human Language Technology and Empirical Methods in Natural Language Processing. HLT '05 (2005) 523–530
6. McDonald, R., Lerman, K., Pereira, F.: Multilingual Dependency Analysis with a Two-stage Discriminative Parser. In: Proceedings of the Tenth Conference on Computational Natural Language Learning. CoNLL-X '06 (2006) 216–220
7. Nakagawa, T.: Multilingual Dependency Parsing Using Global Features. In: Proceedings of the CoNLL Shared Task Session of EMNLP-CoNLL 2007. (2007) 952–956
8. Koo, T., Collins, M.: Efficient Third-order Dependency Parsers. In: Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics. ACL '10 (2010) 1–11
9. Yamada, H., Matsumoto, Y.: Statistical dependency analysis with support vector machines. In: Proceedings of the 8th International Workshop of Parsing Technologies, IWPT'03. (2003)
10. Nilsson, J., Nivre, J., Hall, J.: Graph Transformations in Data-Driven Dependency Parsing. In: Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics. (July 2006) 257–264
11. Nivre, J., Hall, J., Nilsson, J., Chanev, A., Eryigit, G., Kübler, S., Marinov, S., Marsi, E.: MaltParser: A language-independent system for data-driven dependency parsing. Natural Language Engineering **13**(January) (January 2007) 1
12. Nivre, J., McDonald, R.: Integrating Graph-Based and Transition-Based Dependency Parsers. In: Proceedings of ACL-08: HLT. (June 2008) 950–958
13. Zhang, Y., Clark, S.: A Tale of Two Parsers: Investigating and Combining Graph-based and Transition-based Dependency Parsing. In: Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing, Honolulu, Hawaii (October 2008) 562–571
14. Marcus, M.P., Marcinkiewicz, M.A., Santorini, B.: Building a large annotated corpus of English: the penn treebank. Comput. Linguist. **19**(2) (June 1993) 313–330
15. Nguyen, P.T., Vu, X.L., Nguyen, T.M.H., Nguyen, V.H., Le, H.P.: Building a Large Syntactically-Annotated Corpus of Vietnamese. In: Proceedings of the Third Linguistic Annotation Workshop. (August 2009) 182–185
16. Johansson, R., Nugues, P.: Extended Constituent-to-dependency Conversion for English. In: Proceedings of 16th Nordic Conference of Computational Linguistics, NODALIDA'07, Tartu, Estonia (2007) 105–112
17. Collins, M.: Three generative, lexicalised models for statistical parsing. In: Proceedings of the 35th Annual Meeting of the Association for Computational Linguistics, ACL'97. (1997) 16–23
18. Seeker, W., Kuhn, J.: Making Ellipses Explicit in Dependency Conversion for a German Treebank. In: Proceedings of the 8th International Conference on Language Resources and Evaluation, LREC'12. (2012) 3132–3139
19. Candito, M., Crabbé, B., Denis, P.: Statistical French dependency parsing: treebank conversion and first results. In: Proceedings of the 7th International Conference on Language Resources and Evaluation, LREC'10. (2010)

20. Gelbukh, A., Calvo, H., Torres, S.: Transforming a constituency treebank into a dependency treebank. In: Proceedings of XXI Conference of the Spanish Society for Natural Language Processing, SEPLN'05. Volume 35. (2005) 145–152
21. Marinov, S., Nivre, J.: A data-driven dependency parser for Bulgarian. In: Proceedings of the Fourth Workshop on Treebanks and Linguistic Theories. (2005)
22. Ma, X., Zhang, X., Zhao, H., Lu, B.L.: Dependency Parser for Chinese Constituent Parsing. In: Joint Conference on Chinese Language Processing. (2010) 1–6
23. Choi, J.D., Palmer, M.: Statistical dependency parsing in Korean: from corpus generation to automatic parsing. In: Proceedings of the Second Workshop on Statistical Parsing of Morphologically Rich Languages. (2011) 1–11
24. Hong, P.L., Nguyen, T.M.H., Roussanaly, A.: Vietnamese Parsing with an Automatically Extracted Tree-Adjoining Grammar. In: Proceedings of the 9th IEEE RIVF International Conference on Computing & Communication Technologies, Research, Innovation, and Vision for the Future, Ieee (February 2012) 1–6
25. Thi, L.N., My, L.H., Viet, H.N., Minh, H.N.T., Hong, P.L.: Building a Treebank for Vietnamese Dependency Parsing. In: Proceedings of the 10th IEEE RIVF International Conference on Computing & Communication Technologies, Research, Innovation, and Vision for the Future. (2013)
26. Le-Hong, P., Nguyen, T.M.H., Nguyen, P.T., Roussanaly, A.: Automated extraction of tree adjoining grammars from a treebank for Vietnamese. In: Proceedings of The Tenth International Workshop on Tree Adjoining Grammars and Related Formalisms. (2010)
27. Choi, J.D., Palmer, M.: Robust constituent-to-dependency conversion for English. In: Proceedings of 9th Treebanks and Linguistic Theories Workshop. (2010) 55–66
28. Marneffe, M.c.D., Manning, C.D.: The Stanford typed dependencies representation. In: Proceedings of the Coling 2008 workshop on Cross-Framework and Cross-Domain Parser Evaluation. Number August (2008) 1–8
29. Čmejrek, M., Cu\vr\'in, J., Havelka, J.: Prague Czech-English Dependency Treebank: Any Hopes for a Common Annotation Scheme? In: HLT-NAACL 2004 Workshop: Frontiers in Corpus Annotation. (May 2004) 47–54
30. Ballesteros, M., Nivre, J.: MaltOptimizer : A System for MaltParser Optimization. In: Proceedings of the 8th International Conference on Language Resources and Evaluation, LREC '12. Number 2006 (2012) 2757–2763
31. Nguyen, D.Q., Nguyen, D.Q., Pham, S.B., Pham, D.D.: Ripple Down Rules for Part-of-Speech Tagging. In: Proceedings of the 12th international conference on Computational linguistics and intelligent text processing - Volume Part I. CICLing'11 (2011) 190–201
32. Nguyen, D.Q., Nguyen, D.Q., Pham, D.D., Pham, S.B.: RDRPOSTagger: A Ripple Down Rules-based Part-Of-Speech Tagger. In: Proc. of the Demonstrations at the 14th Conference of the European Chapter of the Association for Computational Linguistics. EACL'14 (2014)