An empirical study for Vietnamese dependency parsing

Dat Quoc Nguyen, Mark Dras and **Mark Johnson**
Department of Computing
Macquarie University, Australia
dat.nguyen@students.mq.edu.au, {mark.dras, mark.johnson}@mq.edu.au

Abstract

This paper presents an empirical comparison of different dependency parsers for Vietnamese, which has some unusual characteristics such as copula drop and verb serialization. Experimental results show that the neural network-based parsers perform significantly better than the traditional parsers. We report the highest parsing scores published to date for Vietnamese with the labeled attachment score (LAS) at 73.53% and the unlabeled attachment score (UAS) at 80.66%.

1 Introduction

Dependency parsing has become a key research topic in natural language processing in the last decade, boosted by the success of the CoNLL 2006 and 2007 shared tasks on multilingual dependency parsing (Buchholz and Marsi, 2006; Nivre et al., 2007a). McDonald and Nivre (2011) identify two types of approaches for dependency parsing: graph-based approaches (McDonald et al., 2005) and transition-based approaches (Nivre et al., 2007b). Most traditional graph- or transition-based dependency parsers (McDonald et al., 2005; Nivre et al., 2007b; Bohnet, 2010; Zhang and Nivre, 2011; Martins et al., 2013; Choi and McCallum, 2013) manually define a set of core and combined features associated with one-hot representations.

Recent work shows that neural network-based parsers obtain the state-of-the-art parsing results across many languages. Chen and Manning (2014), Weiss et al. (2015), Pei et al. (2015), and Andor et al. (2016) represent the core features with dense vector embeddings and then feed them as inputs to neural network-based classifiers, while Dyer et al. (2015), Kiperwasser and Goldberg (2016a), and Kiperwasser and Goldberg (2016b) propose novel neural network architectures to solve the feature-engineering problem.

Dependency parsing for Vietnamese has not been actively explored. One main reason is because there is no manually labeled dependency treebank available. Thi et al. (2013) and Nguyen et al. (2014b) propose constituent-to-dependency conversion approaches to automatically translate the manually built constituent treebank for Vietnamese (Nguyen et al., 2009) to dependency treebanks. The converted dependency treebanks are then used in later works on Vietnamese dependency parsing, including Vu-Manh et al. (2015), Le-Hong et al. (2015) and Nguyen and Nguyen (2015). All of the previous research works use either the MSTparser (McDonald et al., 2005) or the Maltparser (Nivre et al., 2007b) for their parsing experiments. Among them, Nguyen et al. (2014b) report the highest results with LAS at 71.66% and UAS at 79.08% obtained by MSTparser. However, MSTparser and Maltparser are no longer considered state-of-the-art parsers.

In this paper, we present an empirical study of Vietnamese dependency parsing. We make comparisons between neural network-based parsers and traditional parsers, and also between graph-based parsers and transition-based parsers. We show that the neural network-based parsers obtain significantly higher scores than the traditional parsers. Specifically, we report the highest up-to-date scores for Vietnamese with LAS at 73.53% and UAS at 80.66%. We also examine potential problems specific to parsing Vietnamese, and point out potential solutions for improving the parsing performance.

2 Experimental setup

Dataset: There are two Vietnamese dependency treebanks which are automatically converted from the manually-annotated Vietnamese constituent

Dat Quoc Nguyen, Mark Dras and Mark Johnson. 2016. An empirical study for Vietnamese dependency parsing. In *Proceedings of Australasian Language Technology Association Workshop*, pages 154–160.

Dep. labels		POS tags		Sent. length	
Type	Rate	Type	Rate	Length	Rate
adv	5.9	A	6.0	$1-10$	19.0
amod	2.4	C	3.7	$11-20$	35.4
conj	1.9	E	6.5	$21-30$	25.6
coord	1.9	M	3.6	$31-40$	12.2
dep	3.1	N	24.6	$41-50$	4.9
det	6.2	Nc	2.4	>50	2.9
dob	6.0	Np	4.2	_	_
loc	2.3	P	4.0	_	_
nmod	19.0	R	7.4	_	_
pob	5.6	V	19.4	_	_
punct	13.9	_	_	_	_
root	4.7	_	_	_	_
sub	6.8	_	_	_	_
tmp	2.2	_	_	_	_
vmod	14.8	_	_	_	_

Table 1: VnDT statistics by most frequent dependency and part-of-speech (POS) labels, and sentence length (i.e. number of words). "Rate" denotes the percentage occurrence in VnDT. Dependency labels: *adv* (adverbial), *amod* (adjectival modifier), *conj* (conjunct), *coord* (coordinating conjunction), *dep* (unspecified dependency), *det* (determiner), *dob* (direct object), *loc* (location), *nmod* (noun modifier), *pob* (object of a preposition), *punct* (punctuation), *sub* (subject), *tmp* (temporal), *vmod* (verb modifier). POS tags: A (Adjective), C (Conjunction), E (Preposition), M (Quantity), N (Noun), Nc (Classifier noun), Np (Proper noun), P (Pronoun), R (Adjunct), V (Verb).

treebank (Nguyen et al., 2009), using conversion approaches proposed by Thi et al. (2013) and Nguyen et al. (2014b). In Thi et al. (2013)'s conversion approach, it is not clear how the dependency labels are inferred; also, it ignores grammatical information encoded in grammatical function tags. In addition, Thi et al. (2013)'s approach is unable to handle cases of coordination and empty category mappings, which frequently appear in the Vietnamese constituent treebank. Nguyen et al. (2014b) later proposed a new conversion approach to handle those cases, with a better use of existing information in the Vietnamese constituent treebank. So we conduct experiments using VnDT, the high quality Vietnamese dependency treebank produced by Nguyen et al. (2014b). The VnDT treebank consists of 10,200 sentences (about 219K words). Table 1 gives some basic statistics of VnDT. We use the last 1020 sentences of VnDT for testing while the remaining sentences are used for training, resulting in an out-of-vocabulary rate of 3.3%.

Dependency parsers: We experiment with four parsers: the graph-based parsers BIST-bmstparser[1] (**BistG**) and MSTparser[2] (**MST**), and the transition-based parsers BIST-barchybrid[3] (**BistT**) and Maltparser[4] (**Malt**). The state-of-the-art BistG and BistT parsers (Kiperwasser and Goldberg, 2016b) employ a bidirectional LSTM RNN architecture (Schuster and Paliwal, 1997; Hochreiter and Schmidhuber, 1997) to automatically learn the feature representation. In contrast, the traditional parsers MST (McDonald et al., 2005) and Malt (Nivre et al., 2007b) use a set of predefined features. For training these parsers, we used the default settings.

Evaluation metrics: The metrics are the labeled attachment score (LAS), unlabeled attachment score (UAS) and label accuracy score (LS). LAS is the percentage of words which are correctly assigned both dependency arc and label while UAS is the percentage of words for which the dependency arc is assigned correctly, and LS is the percentage of words for which the dependency label is assigned correctly.

3 Main results

3.1 Overall accuracy

Table 2 compares the parsing results obtained by the four parsers. The first four rows report the scores with gold part-of-speech (POS) tags while the last four rows present the scores with automatically predicted POS tags.[5]

As expected the neural network-based parsers BistG and BistT perform significantly better than the traditional parsers MST and Malt.[6] Specifically, we find $2^{+}\%$ absolute improvements in LAS and UAS scores in both graph- and transition-based types. In most cases, there are no significant differences between the LAS and UAS scores of BistG and BistT, except LAS scored on gold POS

[1] https://github.com/elikip/bist-parser/tree/master/bmstparser

[2] http://www.seas.upenn.edu/~strctlrn/MSTParser/MSTParser.html

[3] https://github.com/elikip/bist-parser/tree/master/barchybrid

[4] http://www.maltparser.org

[5] We adapted the RDRPOSTagger toolkit (Nguyen et al., 2014a; Nguyen et al., 2016) to automatically assign POS tags to words in the test set with an accuracy of 94.58%.

[6] Using McNemar's test, the differences are statistically significant at $p < 0.001$.

	System	With punctuation						Without punctuation					
		Overall			Exact match			Overall			Exact match		
		LAS	UAS	LS	LAS	UAS	LS	LAS	UAS	LS	LAS	UAS	LS
Gold POS	BistG	**73.17**	**79.39**	**84.22**	**11.27**	**19.71**	15.20	**73.53**	80.66	**81.86**	**11.96**	**20.88**	15.20
	BistT	72.53	79.33	83.71	**11.27**	19.41	**16.18**	72.91	**80.73**	81.29	11.67	20.29	**16.18**
	MST	70.29	76.47	83.23	8.43	12.94	14.02	71.61	78.71	80.72	9.80	16.37	14.02
	Malt	69.10	74.91	81.72	9.22	14.80	13.92	70.39	77.08	79.33	9.71	17.16	13.92
Auto POS	BistG	**68.40**	76.28	**80.56**	9.12	16.18	11.76	**68.50**	77.55	**77.65**	9.71	**17.25**	11.76
	BistT	68.22	**76.56**	80.22	**9.80**	**16.27**	**13.24**	68.31	**77.91**	77.27	**10.00**	17.06	**13.24**
	MST	65.99	73.94	79.78	6.86	10.78	10.88	66.99	76.12	76.75	7.84	13.33	10.88
	Malt	64.94	72.32	78.43	7.35	12.25	10.20	65.88	74.36	75.56	7.55	14.02	10.20

Table 2: Parsing results. "Without punctuation" denotes parsing results where the punctuation and other symbols are excluded from evaluation. "Exact match" denotes the proportion of sentences whose predicted dependency trees are entirely correct.

tags (73.17% against 72.53%, and 73.53% against 72.91%).[7] Compared to the previous highest results (LAS at 71.66% and UAS at 79.08%) scored without punctuation on gold POS tags in Nguyen et al. (2014b), we obtain better scores (LAS at 73.53% and UAS at 80.66%) with BistG.

Next, Section 3.2 gives a detailed accuracy analysis on gold POS tags **without** punctuation, and Section 3.3 discusses the source of some errors and possible improvements.

3.2 Accuracy analysis

Sentence length: Figures 1 and 2 detail LAS and UAS scores by sentence length in bins of length 10. It is not surprising that all parsers produce better results for shorter sentences. For sentences shorter than 10 words, all LAS and UAS scores are around 80% and 85%, respectively. However, the scores drop by 10^{+}% for sentences longer than 50 words. The Malt parser obtains the lowest LAS and UAS scores across all sentence bins. BistG obtains the highest scores for sentences shorter than 20 words while BistT obtains highest scores for sentences longer than 40 words. BistG, BistT and MST perform similarly on 30-to-40-word sentences. For shorter sentences from 20 to 30 words, BistG and BistT produce similar results but higher than obtained by MST.

Dependency distance: Figures 3 and 4 show F_1 scores in terms of the distance from each dependent word to its head. Similar to English (Choi et al., 2015), we find better predictions for the left dependencies than for the right dependencies. Unlike in English where the lower scores are associated with longer distances, we find a different pattern when predicting the left dependencies

[7]The differences are statistically significant at $p < 0.02$.

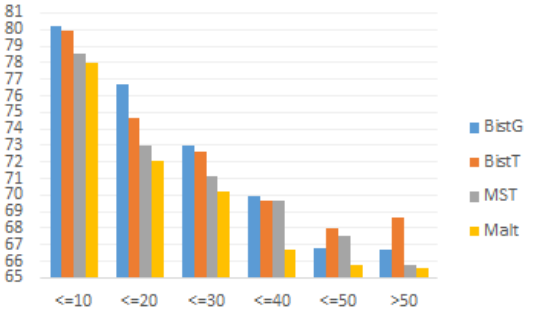

Figure 1: LAS by sentence length.

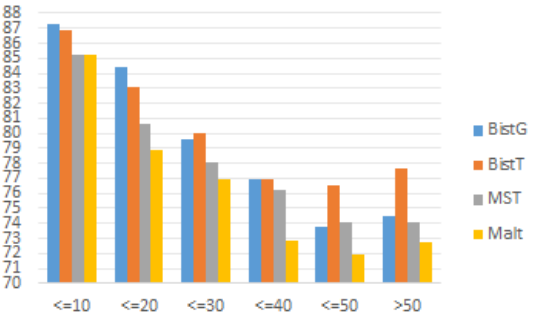

Figure 2: UAS by sentence length

in Vietnamese. In a distance bin of 3, 4 and 5 words with respect to the left dependencies, three over four parsers including BistG, BistT and Malt generally obtain better predictions for longer distances. Compared to English, Vietnamese is head-initial, so finding a difference with respect to left dependencies is not completely unexpected. In addition, for this distance bin, the transition-based parser does better than the graph-based parser in both neural net-based and traditional categories (i.e. BistT $>$ BistG and Malt $>$ MST). In both those categories, however, the graph-based parser does better than the transition-based parser for 5-word-longer distances (i.e. BistG $>$ BistT and MST $>$ Malt), while they produce similar results on dependency distances of 1 or 2 words.

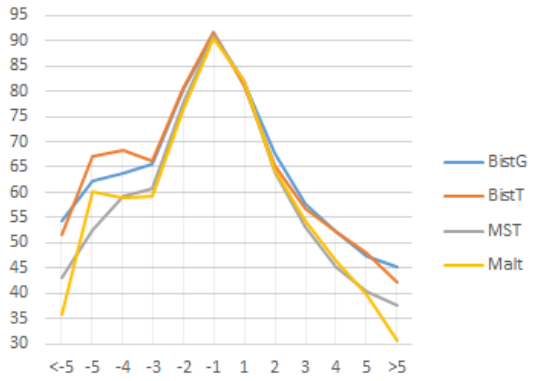

Figure 3: F_1 scores by dependency distance for labeled attachment

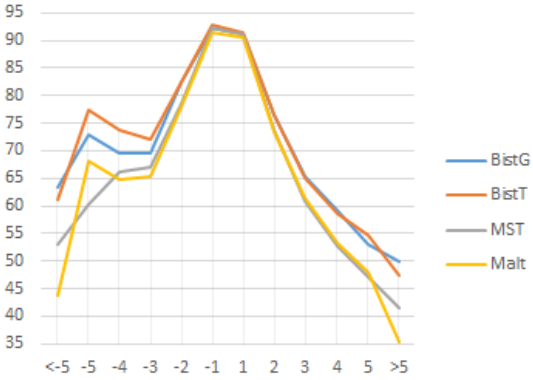

Figure 4: F_1 scores by dependency distance for unlabeled attachment

Because the dependency distance of 3, 4 or 5 occurs quite frequently in long sentences, so the results here are consistent with the results shown in Figures 1 and 2 where BistT obtains the highest scores for long sentences.

Dependency labels: Table 3 presents LAS scores for the most frequent dependency labels. The labels with higher than 90% accuracy are *adv*, *det* and *pob* in which surprisingly MST obtains the best results on all these labels and even on both *conj* and *dob* labels. BistT obtains the best scores on the two most frequent labels *nmod* and *vmod*, and also on the *loc* label. BistG performs best on the remaining labels. Biggest ranges ($> 10\%$) of obtained scores across parsers associate to labels *coord*, *dep*, *sub* and *tmp*.

Table 3 also shows that the label with the lowest LAS scores ($< 50\%$) across all parsers is *dep* which is a very general label. Those with LAS scores ranging from 50% to about 60% are *coord*, *loc*, *tmp* and *vmod* in which *coord*, *loc* and *tmp* are among the least frequent labels, while *vmod* is the second most frequent label.

Type	BistG	BistT	MST	Malt	Avg.
adv	92.09	**92.40**	**92.40**	92.33	92.31
amod	**77.30**	73.89	76.11	73.21	75.13
conj	74.82	73.11	**78 .00**	71.64	74.39
coord	**57.49**	49.52	46.14	52.66	51.45
dep	**47.83**	46 .00	32.54	42.08	42.11
det	94.15	94.30	**95.27**	94.52	94.56
dob	73.01	70.81	**78.62**	76.35	74.70
loc	52.54	**53.86**	51.43	50.77	52.15
nmod	79.34	**79.51**	78.10	76.67	78.41
pob	94.35	95.27	**96.18**	95.85	95.41
root	**85.69**	82.55	82.06	74.41	81.18
sub	**73.34**	72.61	66.49	62.67	68.78
tmp	**60.68**	57.05	44.66	41.45	50.96
vmod	61.51	**62.02**	60.79	60.23	61.14

Table 3: LAS by most frequent dependency labels. "Avg." denotes the averaged score of four parsers.

POS	LAS				UAS			
	BistG	BistT	MST	Malt	BistG	BistT	MST	Malt
A	68.32	**70.31**	69.89	66.83	73.01	**75.50**	74.86	70.88
C	**55.90**	50.00	44.94	50.00	**61.33**	56.87	50.60	54.94
E	**55.47**	53.87	50.91	49.96	**72.27**	71.54	68.86	64.45
M	92.11	91.05	**93.03**	91.18	93.42	93.16	**94.21**	91.71
N	**74.37**	73.58	73.77	71.30	**83.95**	83.86	82.58	80.48
Nc	69.86	**72.02**	68.49	67.12	78.47	**79.26**	76.13	74.17
Np	84.69	84.47	**84.80**	82.84	**88.49**	**88.49**	88.06	86.43
P	79.34	**80.16**	79.69	77.23	85.45	**85.92**	84.62	82.39
R	91.94	**93.08**	92.42	92.60	92.90	**93.87**	92.96	93.27
V	**68.01**	66.49	63.78	63.13	**75.05**	74.95	71.83	70.49

Table 4: Results by most frequent POS tags.

POS tags: In Table 4 we analyze the results by the POS tag of the dependent. BistG achieves the highest results on the two most frequent POS tags *N* and *V* and also on *C* and *E*. BistT achieves the highest scores on the remaining POS tags except *M* for which MST produces the highest score.

3.3 Discussions

Linguistic aspects: One surprising characteristic of the results is the poor performance of verb-related dependencies: *vmod* accuracy is low, as are scores associated with the second most frequent POS tag *V* (Verb). For the latter, we find significantly lower scores for verbs in Vietnamese (around 65% as shown in Table 4) against scores for verbs (about $80^{+}\%$) obtained by MST and Malt parsers on 13 other languages reported in McDonald and Nivre (2011), and also much worse performance in terms of rank relative to other POS.

This may be related to syntactic characteristics of Vietnamese (Thompson, 1987). First, Vietnamese is described as a copula-drop language. Consider *Cô Hà có nhà đẹp* "Miss Hà has a beautiful house", where the attributive adjective *đẹp*

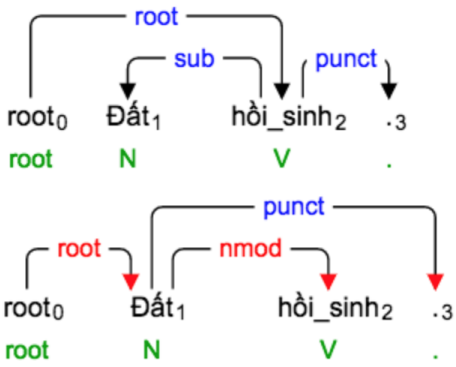

Figure 5: An example of a Vietnamese sentence with copula-drop. The first parsed tree is the gold one while the second parsed tree is the same output produced by all parsers. This sentence is translated to English as "the land$_{\#1}$ is revived$_{\#2}$.", in which the copula "is" is dropped in Vietnamese. The subscripts in the English-translated sentence refer to alignments with the word indexes in the Vietnamese sentence.

"beautiful" postmodifies the noun *nhà* "house". Adjectives can also be predicative, where they are conventionally labelled *V* (Verb), and a copula is absent: with Vietnamese's SVO word order, this is also *nhà đẹp* "the house is beautiful." Figure 5 presents an example from the treebank: all four parsers produce the incorrect structure, which is what would be expected for the attributive adjectival use in an NP. This construction is quite common in Vietnamese.

Second, Vietnamese permits verb serialization, as in Figure 6: *giật_mình* "accuses" should be a *vmod* dependent of *có* "excuses"; such a construction is analogous to the more familiar *nmod* in other languages. Verb dependencies in Vietnamese might thus be less predictable than in other languages, with a more varied distribution of dependents.

Other aspects: Generally, one reason for low overall scores on Vietnamese dependency parsing when compared to the scores obtained on the other languages (McDonald and Nivre, 2011) is probably because of the complex structures of many long sentences in the VnDT treebank (e.g. 45% of the sentences in VnDT consist of more than 20 words). So we can only obtain 60% and 50% for left and right dependency distances larger than 5 as shown in Figure 4, respectively, while for English both left and right dependencies with distances larger than 5 have greater than 70% accuracy (Choi et al., 2015).

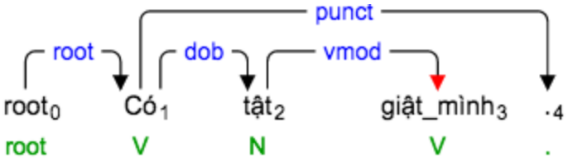

Figure 6: An example of a Vietnamese sentence with verb serialization (and pronoun-dropping), parsed by BistT. The gold parsed tree is when the indexed-3 word is attached to the indexed-1 word by *vmod*, instead of the indexed-2 word. For this sentence, BistG, MST and Malt attach the indexed-3 word to be dependent to the indexed-2 word by the label *nmod*. This sentence is translated to English as "He who excuses$_{\#1}$ himself$_{\#2}$, accuses$_{\#3}$ himself ."

Oracle	With punct.			Without punct.		
	LAS	UAS	LS	LAS	UAS	LS
Tree	79.20	85.22	88.38	79.33	86.24	86.66
Arc	85.98	90.50	92.67	85.96	91.14	91.57

Table 5: Upper bound of ensemble performance.

One simple approach to improve parsing performance for Vietnamese is to separately use the graph-based parser BistG for short sentences and the transition-based parser BistT for longer sentences. Another approach is to use system combination (Nivre and McDonald, 2008; Zhang and Clark, 2008), e.g. building ensemble systems (Sagae and Tsujii, 2007; Surdeanu and Manning, 2010; Haffari et al., 2011). Table 5 presents an upper bound of oracle ensemble performance, using the DEPENDABLE toolkit (Choi et al., 2015). DEPENDABLE assumes that either the best tree or the best arc can be determined by an oracle.

4 Conclusions

We have presented an empirical comparison for Vietnamese dependency parsing. Experimental results on the Vietnamese dependency treebank VnDT (Nguyen et al., 2014b) show that the neural network-based parsers (Kiperwasser and Goldberg, 2016b) obtain significantly higher scores than the traditional parsers (McDonald et al., 2005; Nivre et al., 2007b). More specifically, in each graph- or transition-based type, we find a 2% absolute improvement of the neural network-based parser over the traditional one.

We report the highest performance up to date for Vietnamese dependency parsing with LAS at 73.53% and UAS at 80.66%.

Acknowledgments

The first author is supported by an International Postgraduate Research Scholarship and a NICTA NRPA Top-Up Scholarship.

References

Daniel Andor, Chris Alberti, David Weiss, Aliaksei Severyn, Alessandro Presta, Kuzman Ganchev, Slav Petrov, and Michael Collins. 2016. Globally Normalized Transition-Based Neural Networks. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*, pages 2442–2452.

Bernd Bohnet. 2010. Very high accuracy and fast dependency parsing is not a contradiction. In *Proceedings of the 23rd International Conference on Computational Linguistics*, pages 89–97.

Sabine Buchholz and Erwin Marsi. 2006. CoNLL-X shared task on multilingual dependency parsing. In *Proceedings of the Tenth Conference on Computational Natural Language Learning, CoNLL-X*, pages 149–164.

Danqi Chen and Christopher Manning. 2014. A fast and accurate dependency parser using neural networks. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing*, pages 740–750.

Jinho D. Choi and Andrew McCallum. 2013. Transition-based dependency parsing with selectional branching. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics*, pages 1052–1062.

Jinho D. Choi, Joel Tetreault, and Amanda Stent. 2015. It Depends: Dependency Parser Comparison Using A Web-based Evaluation Tool. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing*, pages 387–396.

Chris Dyer, Miguel Ballesteros, Wang Ling, Austin Matthews, and Noah A. Smith. 2015. Transition-Based Dependency Parsing with Stack Long Short-Term Memory. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing*, pages 334–343.

Gholamreza Haffari, Marzieh Razavi, and Anoop Sarkar. 2011. An ensemble model that combines syntactic and semantic clustering for discriminative dependency parsing. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 710–714.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural Computation*, 9(8):1735–1780.

Eliyahu Kiperwasser and Yoav Goldberg. 2016a. Easy-first dependency parsing with hierarchical tree lstms. *Transactions of the Association for Computational Linguistics*, 4:445–461.

Eliyahu Kiperwasser and Yoav Goldberg. 2016b. Simple and Accurate Dependency Parsing Using Bidirectional LSTM Feature Representations. *Transactions of the Association for Computational Linguistics*, 4:313–327.

Phuong Le-Hong, Thi-Minh-Huyen Nguyen, Thi-Luong Nguyen, and My-Linh Ha. 2015. Fast dependency parsing using distributed word representations. In *Proceedings of the PAKDD 2015 Workshops*, pages 261–272.

Andre Martins, Miguel Almeida, and Noah A. Smith. 2013. Turning on the Turbo: Fast Third-Order Non-Projective Turbo Parsers. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 617–622.

Ryan McDonald and Joakim Nivre. 2011. Analyzing and integrating dependency parsers. *Computational Linguistics*, 37(1):197–230.

Ryan McDonald, Koby Crammer, and Fernando Pereira. 2005. Online large-margin training of dependency parsers. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, pages 91–98.

Kiet Van Nguyen and Ngan Luu-Thuy Nguyen. 2015. Error Analysis for Vietnamese Dependency Parsing. In *Proceedings of the 2015 Seventh International Conference on Knowledge and Systems Engineering*, pages 79–84.

Phuong Thai Nguyen, Xuan Luong Vu, Thi Minh Huyen Nguyen, Van Hiep Nguyen, and Hong Phuong Le. 2009. Building a Large Syntactically-Annotated Corpus of Vietnamese. In *Proceedings of the Third Linguistic Annotation Workshop*, pages 182–185.

Dat Quoc Nguyen, Dai Quoc Nguyen, Dang Duc Pham, and Son Bao Pham. 2014a. RDRPOSTagger: A Ripple Down Rules-based Part-Of-Speech Tagger. In *Proceedings of the Demonstrations at the 14th Conference of the European Chapter of the Association for Computational Linguistics*, pages 17–20.

Dat Quoc Nguyen, Dai Quoc Nguyen, Son Bao Pham, Phuong-Thai Nguyen, and Minh Le Nguyen. 2014b. From Treebank Conversion to Automatic Dependency Parsing for Vietnamese. In *Proceedings of 19th International Conference on Application of Natural Language to Information Systems*, pages 196–207.

Dat Quoc Nguyen, Dai Quoc Nguyen, Dang Duc Pham, and Son Bao Pham. 2016. A robust transformation-based learning approach using ripple down rules for part-of-speech tagging. *AI Communications*, 29(3):409–422.

Joakim Nivre and Ryan McDonald. 2008. Integrating Graph-Based and Transition-Based Dependency Parsers. In *Proceedings of ACL-08: HLT*, pages 950–958.

Joakim Nivre, Johan Hall, Sandra Kübler, Ryan McDonald, Jens Nilsson, Sebastian Riedel, and Deniz Yuret. 2007a. The CoNLL 2007 Shared Task on Dependency Parsing. In *Proceedings of the CoNLL Shared Task Session of EMNLP-CoNLL 2007*, pages 915–932.

Joakim Nivre, Johan Hall, Jens Nilsson, Atanas Chanev, Gülsen Eryigit, Sandra Kübler, Svetoslav Marinov, and Erwin Marsi. 2007b. MaltParser: A language-independent system for data-driven dependency parsing. *Natural Language Engineering*, 13(January):1.

Wenzhe Pei, Tao Ge, and Baobao Chang. 2015. An effective neural network model for graph-based dependency parsing. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing*, pages 313–322.

Kenji Sagae and Jun'ichi Tsujii. 2007. Dependency parsing and domain adaptation with LR models and parser ensembles. In *Proceedings of the CoNLL Shared Task Session of EMNLP-CoNLL 2007*, pages 1044–1050.

M. Schuster and K.K. Paliwal. 1997. Bidirectional recurrent neural networks. *IEEE Transactions on Signal Processing*, 45(11):2673–2681.

Mihai Surdeanu and Christopher D. Manning. 2010. Ensemble models for dependency parsing: Cheap and good? In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 649–652.

Luong Nguyen Thi, Linh Ha My, Hung Nguyen Viet, Huyen Nguyen Thi Minh, and Phuong Le Hong. 2013. Building a treebank for Vietnamese dependency parsing. In *Proceedings of 2013 IEEE RIVF International Conference on Computing and Communication Technologies, Research, Innovation, and Vision for the Future*, pages 147–151.

Laurence C. Thompson. 1987. *A Vietnamese Reference Grammar*. University of Hawaii Press.

Cam Vu-Manh, Anh Tuan Luong, and Phuong Le-Hong. 2015. Improving vietnamese dependency parsing using distributed word representations. In *Proceedings of the Sixth International Symposium on Information and Communication Technology*, pages 54–60.

David Weiss, Chris Alberti, Michael Collins, and Slav Petrov. 2015. Structured training for neural network transition-based parsing. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing*, pages 323–333.

Yue Zhang and Stephen Clark. 2008. 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*, pages 562–571.

Yue Zhang and Joakim Nivre. 2011. Transition-based dependency parsing with rich non-local features. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 188–193.