A Fast and Lightweight System for Multilingual Dependency Parsing

Tao Ji, Yuanbin Wu, Man Lan
Department of Computer Science and Technology
East China Normal University
10132130251@stu.ecnu.edu.cn
{ybwu, mlan}@cs.ecnu.edu.cn

Abstract

Following Kiperwasser and Goldberg (2016), we present a multilingual dependency parser with a bidirectional-LSTM (BiLSTM) feature extractor and a multi-layer perceptron (MLP) classifier. We trained our transition-based projective parser in UD version 2.0 datasets without any additional data. The parser is fast, lightweight and effective on big treebanks.

In the *CoNLL 2017 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*, the official results show that the macro-averaged LAS F1 score of our system *Mengest* is 61.33%.

1 Introduction

Developing tools that can process multiple languages has always been an important goal in NLP. Ten years ago, CoNLL 2006 (Buchholz and Marsi, 2006) and CoNLL 2007 (Nivre et al., 2007) Shared Task were a major milestone for multilingual dependency parsing. The *CoNLL 2017 UD Shared Task* (Zeman et al., 2017) is an extension of the tasks addressed in previous years. Unlike CoNLL 2006 and CoNLL 2007, the focus of the *CoNLL 2017 UD Shared Task* is learning syntactic dependency parsers on a universal syntactic annotation standard. This shared task requires participants to parse raw texts from different languages, which vary both in typology and training set size.

The *CoNLL 2017 UD Shared Task* provided universal dependencies description from LREC 2016 (Nivre et al., 2016), two datasets, which are UD version 2.0 datasets (Nivre et al., 2017b) and this task test datasets (Nivre et al., 2017a), two baseline models, which are UDPipe (Straka et al., 2016) and SyntaxNet (Weiss et al., 2015), and the evaluation platform TIRA (Potthast et al., 2014).

In this paper, We present our multilingual dependency parsing system *Mengest* for *CoNLL 2017 UD Shared Task*. The system contains a BiLSTM feature extractor for feature representation and a MLP classifier for the transition system. The inputs of our system are word form (lemma or stem, which depending on the particular treebank) and part of speech (POS) tags (coarse-grained and fine-grained) for each token. Based on this input, the system finds a governor for each token, and assigns a universal dependency relation label to each syntactic dependency. Our official submission obtains 61.33% macro-averaged LAS F1 score on all treebanks.

The rest of this paper is organized as follows. Section 2 discusses the transition-based model (Kiperwasser and Goldberg, 2016) and our implementation. Section 3 explains how our system deals with parallel sets and surprise languages. Finally, we present experimental and official results in Section 4.

2 System Description

We implement a transition-based projective parser following Kiperwasser and Goldberg (2016). The system consists of a BiLSTM feature extractor and an MLP classifier. We describe their model and our implementation in the following sections in detail.

2.1 Arc-Hybrid System

In this work, we use the arc-hybrid transition system (Kuhlmann et al., 2011). In the arc-hybrid system, a configuration $c = (\alpha, \beta, A)$ consists of a stack α, a buffer β, and a set of dependency arcs A. Given n words sentence $s = w_1, \cdots, w_n$, the initial configuration $c = (\emptyset, \{1, 2, \cdots, n, root\}, \emptyset)$ with an empty stack, an empty arc set, and a full buffer $\beta = 1, 2, \cdots, n, root$, where $root$ is the special root index. The terminal configuration set

Proceedings of the CoNLL 2017 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies, pages 237–242,
Vancouver, Canada, August 3-4, 2017.

contains configurations with an empty stack, an arc set and a buffer containing only $root$.

For each configuration $c = (\sigma|s_1|s_0, b_0|\beta, A)$, the arc-hybrid system has 3 kinds of transitions, $T = \{\text{SHIFT}, \text{LEFT}_l, \text{RIGHT}_l\}$:

$$\begin{aligned}
&\text{SHIFT}(c) = (\sigma|s_1|s_0|b_0, \beta, A) \\
&\quad s.t.\ |\beta| > 0 \\
&\text{LEFT}_l(c) = \big(\sigma|s_1, b_0|\beta, A \cup \{(b_0, s_0, l)\}\big) \\
&\quad s.t.\ |\beta| > 0, |\sigma| > 0 \\
&\text{RIGHT}_l(c) = \big(\sigma|s_1, b_0|\beta, A \cup \{(s_1, s_0, l)\}\big) \\
&\quad s.t.\ |\sigma| > 0, s_0 \neq root
\end{aligned}$$

The SHIFT transition moves the first item of the buffer (b_0) to the stack. The LEFT_l transition removes the first item on top of the stack (s_0) and attaches it as a modifier to b_0 with label l, adding the arc (b_0, s_0, l) to arc set A. The RIGHT_l transition removes s_0 from the stack and attaches it as a modifier to the next item on the stack (s_1), adding the arc (s_1, s_0, l) to arc set A.

We apply a classifier to determine the best action for a configuration. Following Chen and Manning (2014), we use a MLP with one hidden layer. The score of the transition $t \in T$ is defined as:

$$\begin{aligned}
&MLP_\theta\big(\phi(c)\big) = W^2 \cdot tanh\big(W^1 \cdot \phi(c) + b^1\big) + b^2 \\
&\text{SCORE}_\theta\big(\phi(c), t\big) = MLP_\theta\big(\phi(c)\big)[t]
\end{aligned}$$

where $\theta = \{W^1, W^2, b^1, b^2\}$ are the model parameters, $\phi(c)$ is the feature representation of the configuration c. $MLP_\theta\big(\phi(c)\big)[t]$ denotes an indexing operation taking the output element which is the class of transition t.

2.2 The Feature Representation

We consider two types of feature repersentations $\phi(c)$ of a configuration: *simple* and *extended*.

Simple: For an input sequence $s = w_1, \cdots, w_n$, we associate each word w_i with a vector x_i:

$$x_i = e(w_i) \circ e(p_i) \circ e(q_i)$$

where $e(w_i)$ is the embedding vector of word w_i, $e(p_i)$ is the embedding vector of POS tag p_i, $e(q_i)$ is the embedding vector of coarse-grained POS (CPOS) tag q_i. The embeddings $e(w_i), e(p_i), e(q_i)$ are randomly initialized (without pre-training) and jointly trained with the parsing model. Then, in order to encode context features, we use a 2-layer sentence level BiLSTM on top of $x_{1:n}$:

$$\begin{aligned}
\vec{h}_t &= LSTM(\vec{h}_{t-1}, x_i, \vec{\theta}) \\
\overleftarrow{h}_t &= LSTM(\overleftarrow{h}_{t+1}, x_i, \overleftarrow{\theta}) \\
v_i &= \vec{h}_i \circ \overleftarrow{h}_i
\end{aligned}$$

$\vec{\theta}$ are the model parameters of the forward hidden sequence $\vec{h}$. $\overleftarrow{\theta}$ are the model parameters of the backward hidden sequence $\overleftarrow{h}$. The vector v_i is our final vector representation of ith token in s, which has took into account both the entire history $\vec{h}_i$ and the entire future $\overleftarrow{h}_i$ by concatenating the matching Long Short-Term Memory Network (LSTM).

For $\phi(c)$, our simple feature function is the concatenated BiLSTM vectors of the top 3 items on the stack and the first item on the buffer. A configuration c is represented by:

$$\phi(c) = v_{s_2} \circ v_{s_1} \circ v_{s_0} \circ v_{b_0}$$

Extended: We add the feature vectors corresponding to the right-most and left-most modifiers of s_0, s_1 and s_2, as well as the left-most modifier of b_0, reaching a total of 11 BiLSTM vectors as extended feature representation. As we will see in experimental sections, using the extended set does indeed improves parsing accuracies.

2.3 Training Details

The training objective is to make the score of correct transitions always above the scores of incorrect transitions. We use a margin-based criteria. Assume T_{gold} is the set of gold transitions at the current configuration c. At each time stamp, the objective function tries to maximize the margin between T_{gold} and $T - T_{gold}$. The hinge loss of a configuration c is defined as:

$$\begin{aligned}
Loss_\theta(c) = \big(1 - &\max_{t_o \in T_{gold}} \text{SCORE}_\theta(\phi(c), t_o) \\
&+ \max_{t_p \in (T - T_{gold})} \text{SCORE}_\theta(\phi(c), t_p)\big)_+
\end{aligned}$$

Our system use the backpropagation algorithm to calculate the gradients of the entire network (including the MLP and the BiLSTM).

Since our parser can only deal with projective dependency trees, we exclude all training examples with non-projective dependencies. This approach undoubtedly downgrades the performance of our system, we plan to use pseudo-projective approach to improve it in the future work.

3 Multilingual Dependency Parsing

There are 81 treebanks in the *CoNLL 2017 UD Shared Task*, including 55 big treebanks, 14 PUD treebanks (additional parallel test sets), 8 small treebanks and 4 surprise language treebanks. For each language treebank of UD version 2.0 training sets, we train a parser only using its monolingual training set (no cross-lingual features). In total, we trained 61 models, 55 on big treebanks and 6 on small treebanks[1]. Our system reads the CoNLL-U files predicted by UDPipe, and uses morphology (lemmas, UPOS, XPOS) predicted by UDPipe.

3.1 Dealing with Parallel Test Sets

There are 14 additional parallel test sets. Our system simply selects one trained model when we encounter a parallel test set where multiple training treebanks exist. For example, although we don't have English-PUD training set but we have English, English-LinES and English-ParTUT training set. So we only use the model trained on English training set to predict English-PUD test set.

3.2 Dealing with Surprise Languages

There are 4 surprise languages in the *CoNLL 2017 UD Shared Task*. Our system simply use the model trained on English to predict 4 surprise languages, without looking at the input words.

4 Results

We trained our system based on a MacBook Air with a Intel Core i5 1.6 GHz CPU and 4G memory. We used the official TIRA (Potthast et al., 2014) to evaluate the system. We used Dynet neural network library to build our system (Neubig et al., 2017).

The hyper-parameters of the final system used for all the reported experiments are detailed in Table 1.

4.1 Token Representation

We compare two constructions of x_i:

- lemma and POS tag ($w_i \circ p_i$).
- lemma, POS tag and CPOS tag ($w_i \circ p_i \circ q_i$).

The performance of different token representations on 4 example languages are given in Table 2. The results show that the CPOS tag improves the LAS measure between 0.5% and 0.72%.

[1] In UD version 2.0 datasets, Kazakh and Uyghur only contain development set, no training set.

Word embedding dimension	100
POS tag embedding dimension	25
CPOS tag embedding dimension	10
Label embedding dimension	25
Hidden units in MLP	100
BiLSTM layers	2
BiLSTM hidden layer dimensions	125
BiLSTM output layer dimensions	125
α (for word dropout)	0.25
Learning rate	0.1
Optimization algorithm	Adam

Table 1: Hyper-parameter values used in shared task.

Language	$w_i \circ p_i$	$w_i \circ p_i \circ q_i$
Bulgarian(bg)	83.78	84.28
Catalan(ca)	85.67	86.26
German(de)	70.77	71.49
English(en)	75.91	76.42

Table 2: The LAS score of two different token representations on the 4 treebanks: Bulgarian(bg), Catalan(ca), German(de), English(en).

4.2 BiLSTM Feature Representation

Performances of simple feature representation and extended feature representation are given in Table 3. The results show that the extended feature representation slightly increases the performance of our system. while the simple feature representation can significantly speed up the system.

	Simple Feature			Extended Feature		
	LAS	train(sec)	test(sec)	LAS	train(sec)	test(sec)
bg	84.24	205.6	22.4	84.28	287.9	29.1
ca	85.74	663.5	37.2	86.26	878.5	48.8
de	71.15	416.8	29.8	71.49	733.2	37.4
en	76.18	375.6	24.5	76.42	524.8	31.1

Table 3: Comparison of Simple and Extended feature representations, we report LAS score, offline training time, and TIRA testing time.

4.3 Overall Performances

In our final submitted system to the shared task, we used lemmas, POS tags and CPOS tags in token representation and selected extended feature representation.

The macro-average LAS of the 55 big treebanks is 68.37% and the results for each language are

Language	LAS(max)	Language	LAS(max)	Language	LAS(max)	Language	LAS(max)
ar	65.65(72.90)	bg	84.28(89.81)	ca	86.26(90.70)	cs	83.85(90.17)
cs_cac	83.22(90.43)	cs_cltt	68.42(85.82)	cu	48.95(76.84)	da	72.78(82.97)
de	71.49(80.71)	el	78.72(87.38)	en	76.42(82.23)	en_lines	72.66(82.09)
en_partut	73.74(84.46)	es	75.41(87.29)	es_ancora	78.64(89.99)	et	55.40(71.65)
eu	62.89(81.44)	fa	61.43(86.31)	fi	69.86(85.64)	fi_ftb	75.13(86.81)
fr	80.07(85.51)	fr_sequoia	79.00(87.31)	gl	79.28(83.23)	got	57.02(71.36)
grc	49.30(73.19)	grc_proiel	60.61(75.28)	he	58.10(68.16)	hi	86.76(91.59)
hr	76.59(85.25)	hu	57.85(77.56)	id	74.40(79.19)	it	86.14(90.68)
ja	73.00(91.13)	ko	63.21(82.49)	la_ittb	74.37(87.02)	la_proiel	54.07(71.55)
lv	59.50(74.01)	nl	68.84(80.48)	nl_lassysmall	71.53(87.71)	no_bokmaal	75.96(89.88)
no_nynorsk	70.97(88.81)	pl	67.63(90.32)	pt	62.85(87.65)	pt_br	79.71(91.36)
ro	64.38(85.92)	ru	56.56(83.65)	ru_syntagrus	82.42(92.60)	sk	60.48(86.04)
sl	61.28(91.51)	sv	61.43(85.87)	sv_lines	61.09(82.89)	tr	49.11(62.79)
ur	61.77(82.28)	vi	31.67(47.51)	zh	58.03(68.56)		

Table 4: The LAS F1 score of our system and best system on the 55 big treebanks: ar, bg, ca, cs, cs_cac, cs_cltt, cu, da, de, el, en, en_lines, en_partut, es, es_ancora, et, eu, fa, fi, fi_ftb, fr, fr_sequoia, gl, got, grc, grc_proiel, he, hi, hr, hu, id, it, ja, ko, la_ittb, la_proiel, lv, nl, nl_lassysmall, no_bokmaal, no_nynorsk, pl, pt, pt_br, ro, ru, ru_syntagrus, sk, sl, sv, sv_lines, tr, ur, vi, zh.

Language	LAS(max)	Language	LAS(max)	Language	LAS(max)	Language	LAS(max)
fr_partut	72.40(88.13)	ga	55.07(70.06)	gl_treegal	61.17(74.34)	kk	_(29.22)
la	38.00(63.37)	sl_sst	23.77(59.07)	ug	_(43.51)	uk	20.61(75.33)

Table 5: The LAS F1 score of our system and best system on the 8 small treebanks: fr_partut, ga, gl_treegal, kk, la, sl_sst, ug, uk.

Language	LAS(max)	Language	LAS(max)	Language	LAS(max)	Language	LAS(max)
ar_pud	43.70(49.94)	cs_pud	80.44(84.42)	de_pud	69.13(74.86)	en_pud	79.02(85.51)
es_pud	72.61(81.05)	fi_pud	71.77(88.47)	fr_pud	73.92(78.81)	hi_pud	51.07(54.49)
it_pud	83.79(88.14)	ja_pud	76.66(83.75)	pt_pud	59.32(78.48)	ru_pud	52.73(75.71)
sv_pud	54.83(78.49)	tr_pud	22.52(38.22)				

Table 6: The LAS F1 score of our system and best system on the 14 PUD treebanks (additional parallel test sets): ar_pud, cs_pud, de_pud, en_pud, es_pud, fi_pud, fr_pud, hi_pud, it_pud, ja_pud, pt_pud, ru_pud, sv_pud, tr_pud.

Language	LAS(max)	Language	LAS(max)	Language	LAS(max)	Language	LAS(max)
bxr	12.44(32.24)	hsb	14.19(61.70)	kmr	8.62(47.53)	sme	10.00(48.96)

Table 7: The LAS F1 score of our system and best system on the 4 surprise language treebanks: bxr, hsb, kmr, sme.

shown in Table 4. The macro-average LAS of the 8 small treebanks is 33.88% and the results for each language are shown in Table 5. The macro-average LAS of the 14 PUD treebanks is 63.68% and the results for each language are shown in Table 6. The macro-average LAS of the 4 surprise language treebanks is 11.31% and the results for each language are shown in Table 7. The macro-averaged LAS F1 score of our system on all treebanks is 61.33%.

4.4 Computational Efficiencies

The parser is fast. Offline training time is about 300 words/sec. Prediction time on the official TIRA is about 400 words/sec without asking for more resources.

Memory requirements are lower than 512M for each language.

5 Conclusions

In this paper, we present a fast and lightweight multilingual dependency parsing system for the *CoNLL 2017 UD Shared Task*, which composed of a BiLSTMs feature extractor and a MLP classifier. Our system only uses UD version 2.0 datasets (without any additional data). The parser makes a good ranking at some of the big treebanks. The results suggests that the simple BiLSTM extractor is a reasonable baseline for multilingual dependency parsing. We will continue to improve our system and add cross-lingual techniques in our future work.

Acknowledgments

We would like to thank the *CoNLL 2017 UD Shared Task* organizers (Jan Hajič, Daniel Zeman, Joakim Nivre, Filip Ginter, Slav Petrov, Milan Straka , Martin Popel, Eduard Bejček, Martin Potthast et al.).

This research is supported by NSFC(61402175).

References

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 2006, New York City, USA, June 8-9, 2006*. pages 149–164. http://aclweb.org/anthology/W/W06/W06-2920.pdf.

Danqi Chen and Christopher D Manning. 2014. A fast and accurate dependency parser using neural networks. In *Empirical Methods in Natural Language Processing (EMNLP)*.

Eliyahu Kiperwasser and Yoav Goldberg. 2016. Simple and accurate dependency parsing using bidirectional LSTM feature representations. *TACL* 4:313–327. https://transacl.org/ojs/index.php/tacl/article/view/885.

Marco Kuhlmann, Carlos Gómez-Rodríguez, and Giorgio Satta. 2011. Dynamic programming algorithms for transition-based dependency parsers. In *The 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies, Proceedings of the Conference, 19-24 June, 2011, Portland, Oregon, USA*. pages 673–682. http://www.aclweb.org/anthology/P11-1068.

Graham Neubig, Chris Dyer, Yoav Goldberg, Austin Matthews, Waleed Ammar, Antonios Anastasopoulos, Miguel Ballesteros, David Chiang, Daniel Clothiaux, Trevor Cohn, Kevin Duh, Manaal Faruqui, Cynthia Gan, Dan Garrette, Yangfeng Ji, Lingpeng Kong, Adhiguna Kuncoro, Gaurav Kumar, Chaitanya Malaviya, Paul Michel, Yusuke Oda, Matthew Richardson, Naomi Saphra, Swabha Swayamdipta, and Pengcheng Yin. 2017. Dynet: The dynamic neural network toolkit. *arXiv preprint arXiv:1701.03980* .

Joakim Nivre, Željko Agić, Lars Ahrenberg, et al. 2017a. Universal dependencies 2.0 CoNLL 2017 shared task development and test data. LINDAT/CLARIN digital library at the Institute of Formal and Applied Linguistics, Charles University. http://hdl.handle.net/11234/1-2184.

Joakim Nivre, Marie-Catherine de Marneffe, Filip Ginter, Yoav Goldberg, Jan Hajič, Christopher Manning, Ryan McDonald, Slav Petrov, Sampo Pyysalo, Natalia Silveira, Reut Tsarfaty, and Daniel Zeman. 2016. Universal Dependencies v1: A multilingual treebank collection. In *Proceedings of the 10th International Conference on Language Resources and Evaluation (LREC 2016)*. European Language Resources Association, Portoro, Slovenia, pages 1659–1666.

Joakim Nivre, Johan Hall, Sandra Kübler, Ryan T. McDonald, Jens Nilsson, Sebastian Riedel, and Deniz Yuret. 2007. The conll 2007 shared task on dependency parsing. In *EMNLP-CoNLL 2007, Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning, June 28-30, 2007, Prague, Czech Republic*. pages 915–932. http://www.aclweb.org/anthology/D07-1096.

Joakim Nivre et al. 2017b. Universal Dependencies 2.0. LINDAT/CLARIN digital library at the Institute of Formal and Applied Linguistics, Charles University, Prague, http://hdl.handle.net/

11234/1-1983. http://hdl.handle.net/11234/1-1983.

Martin Potthast, Tim Gollub, Francisco Rangel, Paolo Rosso, Efstathios Stamatatos, and Benno Stein. 2014. Improving the reproducibility of PAN's shared tasks: Plagiarism detection, author identification, and author profiling. In Evangelos Kanoulas, Mihai Lupu, Paul Clough, Mark Sanderson, Mark Hall, Allan Hanbury, and Elaine Toms, editors, *Information Access Evaluation meets Multilinguality, Multimodality, and Visualization. 5th International Conference of the CLEF Initiative (CLEF 14)*. Springer, Berlin Heidelberg New York, pages 268–299. https://doi.org/10.1007/978-3-319-11382-1_22.

Milan Straka, Jan Hajič, and Jana Straková. 2016. UDPipe: trainable pipeline for processing CoNLL-U files performing tokenization, morphological analysis, POS tagging and parsing. In *Proceedings of the 10th International Conference on Language Resources and Evaluation (LREC 2016)*. European Language Resources Association, Portoro, Slovenia.

David Weiss, Chris Alberti, Michael Collins, and Slav Petrov. 2015. Structured training for neural network transition-based parsing. *CoRR* abs/1506.06158. http://arxiv.org/abs/1506.06158.

Daniel Zeman, Martin Popel, Milan Straka, Jan Hajič, Joakim Nivre, Filip Ginter, Juhani Luotolahti, Sampo Pyysalo, Slav Petrov, Martin Potthast, Francis Tyers, Elena Badmaeva, Memduh Gökırmak, Anna Nedoluzhko, Silvie Cinková, Jan Hajič jr., Jaroslava Hlaváčová, Václava Kettnerová, Zdeňka Urešová, Jenna Kanerva, Stina Ojala, Anna Missilä, Christopher Manning, Sebastian Schuster, Siva Reddy, Dima Taji, Nizar Habash, Herman Leung, Marie-Catherine de Marneffe, Manuela Sanguinetti, Maria Simi, Hiroshi Kanayama, Valeria de Paiva, Kira Droganova, Hěctor Martínez Alonso, Hans Uszkoreit, Vivien Macketanz, Aljoscha Burchardt, Kim Harris, Katrin Marheinecke, Georg Rehm, Tolga Kayadelen, Mohammed Attia, Ali Elkahky, Zhuoran Yu, Emily Pitler, Saran Lertpradit, Michael Mandl, Jesse Kirchner, Hector Fernandez Alcalde, Jana Strnadova, Esha Banerjee, Ruli Manurung, Antonio Stella, Atsuko Shimada, Sookyoung Kwak, Gustavo Mendonça, Tatiana Lando, Rattima Nitisaroj, and Josie Li. 2017. CoNLL 2017 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies. In *Proceedings of the CoNLL 2017 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*. Association for Computational Linguistics.