Modeling Input Uncertainty in Neural Network Dependency Parsing

Rob van der Goot
University of Groningen
r.van.der.goot@rug.nl

Gertjan van Noord
University of Groningen
g.j.m.van.noord@rug.nl

Abstract

Recently introduced neural network parsers allow for new approaches to circumvent data sparsity issues by modeling character level information and by exploiting raw data in a semi-supervised setting. Data sparsity is especially prevailing when transferring to non-standard domains. In this setting, lexical normalization has often been used in the past to circumvent data sparsity. In this paper, we investigate whether these new neural approaches provide similar functionality as lexical normalization, or whether they are complementary.

We provide experimental results which show that a separate normalization component improves performance of a neural network parser even if it has access to character level information as well as external word embeddings. Further improvements are obtained by a straightforward but novel approach in which the top-N best candidates provided by the normalization component are available to the parser.

1 Introduction

Recently, neural network dependency parsers (Chen and Manning, 2014; Dyer et al., 2015; Kiperwasser and Goldberg, 2016) obtained state-of-the-art performance for dependency parsing. These parsers incorporate character level information (de Lhoneux et al., 2017a; Ballesteros et al., 2015; Nguyen et al., 2017) and can more easily exploit raw text in a semi-supervised setup. These new methods are especially beneficial for words not occurring in the training data. In practice, such unseen words often are spelling mistakes, or alternative spellings of known words. In more classical parsing models, these unseen words were usually clustered using ad-hoc rules. For non-standard domains, the number of unseen words is much larger. To minimize the degradation in performance, lexical normalization is often used. Lexical normalization is the task of converting non-standard input to a more standard format. Previous work has shown that this is beneficial, in particular for parsing social media data (Foster, 2010; Zhang et al., 2013; van der Goot and van Noord, 2017b).

This leads to the question whether normalization is indeed no longer required for these modern character-based neural network parsers, or whether normalization is capable of solving problems beyond the scope of this type of neural network parsers.

Our main contributions are:

- We show that using normalization as preprocessing improves parser performance for non-standard language, even if pre-trained embeddings and character level information are used.
- We propose a novel technique to exploit the top-N candidates provided by the normalization component, and we show that this technique leads to a further increase in parser performance.
- A treebank containing non-standard language is created to evaluate the effect of normalization on parser performance. The treebank consists of 10,005 tokens annotated with lexical normalization and Universal Dependencies (Nivre et al., 2017). The treebank has been made publicly available.

2 Related Work

Early work on parser adaptation focused on relatively canonical domains, like biomedical data (McClosky and Charniak, 2008). More recently, there has been an increasing interest in parsing of the notoriously noisy domain of social media. A lot of previous work is orthogonal to our

Original word	new		pix		comming		tomoroe	
Cand. 1 (p1)	new	(0.95)	pix	(0.79)	coming	(0.57)	tomorrow	(0.54)
Cand. 2 (p2)	news	(0.03)	selfies	(0.08)	comming	(0.43)	tomoroe	(0.39)
Cand. 3 (p3)	knew	(0.01)	pictures	(0.06)	combing	(<0.01)	tomorrow's	(0.02)

Table 1: Output of the normalization model for the example sentence "new pix comming tomoroe" including candidate probabilities. Only the top-3 candidates are shown here.

approach, as it focuses on adaptation of the training data (Foster et al., 2011; Khan et al., 2013; Kong et al., 2014; Blodgett et al., 2018). In the remainder of this section we will shortly review work which evaluated the effect of normalization on dependency parsing.

Zhang et al. (2013) tune a normalization model for the parsing task, and show performance improvement on a silver treebank obtained from manually normalized data. Daiber and van der Goot (2016) use an existing normalization model as pre-processing for a graph-based dependency parser, and show a small but significant performance improvement. In the shared task of parsing the web (Petrov and McDonald, 2012) held at SANCL 2012, some teams used a simple rule-based normalization, but the effect on final performance remained untested.

Baldwin and Li (2015) examined the theoretical impact of different normalization actions on parsing performance. To this end they use manual normalization. They show that edits beyond the word level can also be crucial for parsing (e.g. insertion of copulas and subjects). However, these are difficult to obtain automatically.

Note that all this previous work, except for Blodgett et al. (2018), is based on traditional feature-based dependency parsers, whereas we focus on neural network parsing.

3 Method

In this section we will first shortly review the two models we will combine: a lexical normalization model and a neural network parser. Then we describe how they can be combined.

3.1 Normalization

In this work we use an existing normalization model: MoNoise (van der Goot and van Noord, 2017a)[1]. This model is based on the observation that normalization requires a variety of different

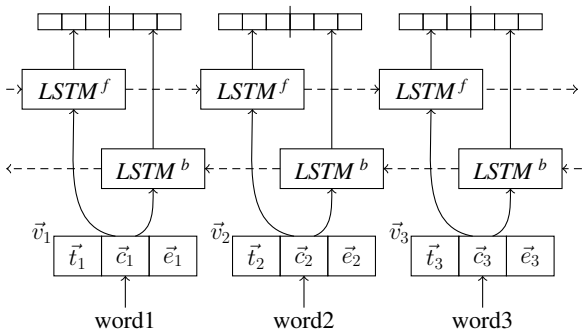

Figure 1: Overview of the conversion of input words to vectors which are used in the shift-reduce algorithm.

replacement actions. For these different actions, different modules are used to generate candidates, including: the Aspell spell checker[2], word embeddings and a lookup list generated from the training data. Features from these generation modules are complemented with N-gram features from canonical data and non-canonical data. A random forest classifier is used to score and rank the candidates. In this work, we use the top-N candidates and convert the confidence scores of the classifier to probabilities. An example of this output is shown in Table 1.

We train MoNoise on 2,577 tweets annotated with normalization by Li and Liu (2014), which only contains word-word replacements. In our inital experiments, we noted that the normalization model wrongfully normalized some words due to the different tokenization in the treebank (e.g. "ca n't"), because these do not occur in the normalization data. We manually created a list of exceptions, which are not considered for normalization process.

3.2 Neural Network Parser

As a starting point, we use the shift-reduce UUParser 2.0 (de Lhoneux et al., 2017b; Kiperwasser and Goldberg, 2016). This parser uses the Arc-Hybrid Transition system (Kuhlmann et al., 2011). Words are first converted to continu-

[1] https://bitbucket.org/robvanderg/monoise

[2] www.aspell.net

ous vectors, which are then processed through a Bidirectional Long-Short Term Memory network (BiLSTM) (Graves and Schmidhuber, 2005) before they are passed on to the parsing algorithm. The decision whether to shift, reduce or swap is made by a multi-layer perceptron with one hidden layer. The BiLSTM is trained jointly with the parsing objective, so that the vectors are optimized for the parsing task.

Figure 1 shows an overview of how the input words are converted to vectors which are used in the shift-reduce algorithm. We denote the vector used as input to the BiLSTM for word i by $\vec{v}_i$. This vector is a concatenation of three vectors which are derived from the input word. $\vec{t}_i$ is optimized on the training data, $\vec{c}_i$ is the result of a separate BiLSTM ran over the characters of word i and $\vec{e}_i$ is the external vector; it is obtained from external embeddings which are trained on huge amounts of raw texts. In this work we use the same word embeddings as used by the normalization model (van der Goot and van Noord, 2017a), which are trained on 760,744,676 tweets using word2vec (Mikolov et al., 2013).

3.3 Adaptation Strategy

Notation We use $\vec{w}_0 ... \vec{w}_n$ to represent the vectors of the original words of a sentence. The vectors of the normalization candidates are represented by $\vec{n}_{ij}$, where i is the index of the original word in the sentence, and j is the rank of the candidate. The corresponding probability as given by the normalization model is p_{ij}. We use $\vec{g}_i$ for the vector of the manual normalization of word i

Our baseline setup is to simply use the vector of the original word:

$$\text{ORIG:} \quad \vec{v}_i = \vec{w}_i$$

The most straightforward use of normalization is to use the best normalization sequence as input to the parser. In our setup, this means that we use the vector of the best normalization candidate for each position:

$$\text{NORM:} \quad \vec{v}_i = \vec{n}_{i0}$$

To give more information to the parser, we will exploit the top-n candidates of the normalization model. The vectors of the top-N candidates are merged using linear interpolation:

$$\text{INTEGRATED:} \quad \vec{v}_i = \sum_{j=0}^{n} p_{ij} * \vec{n}_{ij}$$

An interesting property of this integration approach is that it does not influence the size of the search space, so the effect on complexity of the parsing algorithm is negligible. The only extra runtime compared to ORIG originates from running the normalization model.

Finally, we include a theoretical upperbound of the effect of normalization, which uses manually annotated normalization:

$$\text{GOLD:} \quad \vec{v}_i = \vec{g}_i$$

4 Data

To test the effect of normalization, we need a treebank containing non-standard language, preferably with a corresponding training treebank from a more standard domain. Since the existing treebanks are not noisy enough (Foster et al., 2011; Kaljahi et al., 2015)[3] or do not have a corresponding training treebank in the same annotation format (Kong et al., 2014; Daiber and van der Goot, 2016) we annotate a small treebank for development and testing purposes[4]. We choose to use the Universal Dependencies 2.1 annotation format (Nivre et al., 2017), since the annotation efforts on the the English Web Treebank (Silveira et al., 2014) provide suitable training data. This treebank already contains web specific phenomena like URL's, E-Mail addresses and emoticons, so we do not have to create special annotation guidelines and the parser can learn these phenomena from the training data.

Our treebank consists of tweets, taken from Li and Liu (2015). The tweets in this dataset originate from two sources: the LexNorm corpus (Han and Baldwin, 2011), which was originally annotated with normalization, and a corpus originally annotated with POS tags (Owoputi et al., 2013). Li and Liu (2015) complemented this annotation for both datasets, so that they both have a normalization layer and a POS layer. To avoid overfitting on a specific filtering or time-frame we use the data collected by Owoputi et al. (2013) as development data and LexNorm as test data. We only keep the tweets which are still available on Twitter, resulting in a dataset of 305 development and

[3] Kaljahi et al. (2015) only normalize 3.6%, and we manually normalized the development data from Foster et al. (2011), were even less words were in need of normalization.

[4] It should be noted that two other suitable Twitter treebanks in the UD format where created in parallel to our treebank (Liu et al., 2018; Blodgett et al., 2018), which were released after submission of this paper.

327 test tweets (10,005 tokens in total). It should be noted that these corpora were filtered to contain domain-specific phenomena and non-standard language, and thus provide an ideal testbed for our experiments but are not representative of the whole Twitter domain.

Tokenization and normalization are first re-annotated, because the Universal Dependencies format requires treebank specific tokenization. To avoid parser bias, dependency relations are annotated from scratch. For more details on annotation decisions for domain-specific structures, we refer to the appendix.

MoNoise reaches 90% accuracy on the word level for the normalization task for our development data. In this dataset, 18% of all words are in need of normalization, so a baseline which simply copies the original words would reach an accuracy of 82%. The most common mistakes made by MoNoise are due to treebank specific normalizations, like 'na' $\mapsto$ go. However, these also occur in the training treebank, so normalization is not crucial.

5 Evaluation

In this section, we first use the development data to compare the effect of the different normalization settings with the use of character level information and external embeddings. Secondly, we confirm our main results on the test set. Thirdly, we test if our model is sensitive to over-normalization on standard data. Finally, we perform some analysis to examine why normalization is beneficial. All scores reported in this section are obtained using the CoNLL 2017 evaluation script (Zeman et al., 2017). In Section 5.1 the results are the average over ten runs, using a different seed for the BiLSTM and the shuffling of the training data. In the remainder of this section, the best model is used to simplify interpretation. The parser is trained using default settings (de Lhoneux et al., 2017b).

In our initial experiments, it became apparent that the parser often considered a username mention or retweet in the beginning of the tweet as root, resulting in a propagation of errors. Because we want to exclude any influences from this simple construction, we added an heuristic to our parser which exclude usernames and retweets in the beginning of a tweet, and connects them to the root after parsing. We use this heuristic in all experiments.

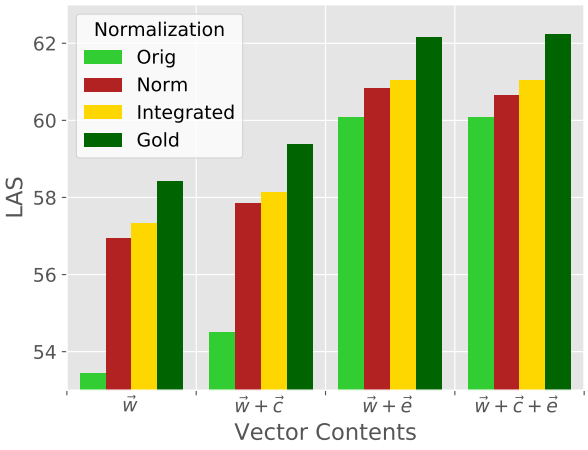

Figure 2: The effect of normalization on LAS for the different parsing models on the development data.

5.1 Normalization Strategies

The results of the different parser and normalization settings on the development data are plotted in Figure 2. Using external embeddings ($\vec{e}$) results in a much bigger performance improvement compared to using character level information ($\vec{c}$). Adding character level embeddings on top of external embeddings only leads to a very minor improvement. This can partly be explained by the coverage of 98.4% of the embeddings on the development data.

In the settings without external embeddings, the direct use of normalization (NORM) results in a improvement of approximately 3 LAS points. However, when external embeddings are included the improvement becomes more than twice as small, indicating that the approaches target some common issues, but are also complementary to each other. When external embeddings and normalization are already used, the character level embeddings slightly harm performance. Integration of the normalization (INTEGRATED) consistently results in a slightly higher LAS compared to direct normalization. Interestingly, gold normalization still performs substantially better compared to automatic normalization.

5.2 Test Data

Table 2 shows the results of the parser with external embeddings and character embeddings (using the best seed from the development data), for the different normalization strategies on the test data. These results confirm the observations on the development data: normalization helps on top of ex-

Model	UAS	LAS
ORIG	69.63	59.64
NORM	70.51	61.76*
INTEGRATED	70.62	62.30*
GOLD	70.71	62.33

Table 2: LAS scores for the Twitter test data. *Statistically significant compared to the previous row at $P < 0.05$ using a paired t-test.

ternal embeddings, and integrating normalization results in an even higher score. In contrast to the development data, the integrated approach almost reaches the theoretical upper bound of gold normalization on the test data. However, since this is only the case on the test data, not too strong conclusions can be drawn from this result. The performance difference between the datasets is probably partly due to the differences in filtering[5]. Interestingly, integrating normalization is especially beneficial for the LAS, meaning that it is most useful for choosing the type of relation.

5.3 Robustness

As stated in Section 4, our development and test data is filtered to be very non-standard. However, it is undesirable to have a parser that performs bad on more standard texts. Hence, we also tested performance on the English Web Treebank development set. This dataset also consists of data from the web, however, it contains much less words in need of normalization; MoNoise normalizes less than 0.5% of all words. We compared the performance using no normalization (ORIG) versus our INTEGRATED approach, which showed a very minor performance improvement from 81.42 to 81.43 LAS. This is a direct effect of the normalization model giving high probabilities to the original words on this more canonical data.

5.4 Analysis

To gain insights into which constructions are parsed better when using normalization, we compared the predictions of the vanilla parser with our NORM and INTEGRATED methods on the development data. Starting with NORM, the first observation is that the incoming arcs of the words which are normalized are responsible for 44.1% of all improvements, whereas the outgoing arcs are responsible for 17.6% of al improvements. So, the direct context of the normalized words is responsible for only 61.7% of all improvements. Considering the type of syntactic constructions for which parsing improved, it is hard to identify trends, because the improvements are based on the output of the normalization model, which normalizes a wide variety of words. One clearly influential effect of using normalization, was that the parser improved upon finding the root. When multiple unknown words occured in the beginning of a sentence, the vanilla parser often failed at identifying the root, which improved considerably after normalizing.

For the INTEGRATED method, almost all the improvements made by NORM remained. On top of these, some additional improvements where made. Manual inspection revealed that these improvements often originated from a non-standard word, for which the correct normalization was ranked high. This then leads to improvements for the non-standard word as well as its context. In some cases, even incorrect normalization candidates lead to performance improvements. For example for 'Gma', where the normalization model ranked the original word first, but 'mom' second. Even though 'grandma' is the correct normalization, 'mom' occurs in similar contexts, and is much easier for the parser to process.

6 Conclusion

We showed that normalization can improve performance of a neural network parser, even when making use of character level information and external word embeddings. Integrating multiple normalization candidates into the parser leads to an even larger performance increase. Normalization has shown to be complementary to external embeddings, in contrast to character embeddings, which add no additional information. Our experiments revealed that our approach is robust, and it does not harm performance on more canonical data. However, when comparing our approach to the theoretical upperbound of using gold normalization, we saw that on different datasets the performance gain is of a different magnitude. Furthermore, we release a dataset containing 636 tweets annotated with both normalization and Universal Dependencies. The data and all code to reproduce the results in this paper is available at: https://bitbucket.org/robvanderg/normpar

[5]Even when using the best seed on the development data, INTEGRATED results in two-thirds of the performance improvement compared to GOLD.

Acknowledgements

We would like to thank Gosse Bouma for help with the data annotation, and Antonio Toral, Barbara Plank and the anonymous reviewers for feedback on the paper. This work is part of the 'Parsing Algorithms for Uncertain Input' project sponsored by the Nuance Foundation.

References

Tyler Baldwin and Yunyao Li. 2015. An in-depth analysis of the effect of text normalization in social media. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 420–429, Denver, Colorado. Association for Computational Linguistics.

Miguel Ballesteros, Chris Dyer, and Noah A. Smith. 2015. Improved transition-based parsing by modeling characters instead of words with LSTMs. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 349–359, Lisbon, Portugal. Association for Computational Linguistics.

Su Lin Blodgett, Johnny Wei, and Brendan O'Connor. 2018. Twitter Universal Dependency parsing for African-American and mainstream American English. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1415–1425. Association for Computational Linguistics.

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 (EMNLP)*, pages 740–750, Doha, Qatar. Association for Computational Linguistics.

Joachim Daiber and Rob van der Goot. 2016. The Denoised Web Treebank: Evaluating dependency parsing under noisy input conditions. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC 2016)*, Paris, France. European Language Resources Association (ELRA).

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 (Volume 1: Long Papers)*, pages 334–343, Beijing, China. Association for Computational Linguistics.

Jennifer Foster. 2010. "cba to check the spelling": Investigating parser performance on discussion forum posts. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 381–384, Los Angeles, California. Association for Computational Linguistics.

Jennifer Foster, Özlem Çetinoglu, Joachim Wagner, Joseph Le Roux, Joakim Nivre, Deirdre Hogan, and Josef van Genabith. 2011. From news to comment: Resources and benchmarks for parsing the language of web 2.0. In *Proceedings of 5th International Joint Conference on Natural Language Processing*, pages 893–901, Chiang Mai, Thailand. Asian Federation of Natural Language Processing.

Rob van der Goot and Gertjan van Noord. 2017a. MoNoise: Modeling noise using a modular normalization system. *Computational Linguistics in the Netherlands Journal*, 7:129–144.

Rob van der Goot and Gertjan van Noord. 2017b. Parser adaptation for social media by integrating normalization. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 491–497, Vancouver, Canada. Association for Computational Linguistics.

Alex Graves and Jürgen Schmidhuber. 2005. Framewise phoneme classification with bidirectional LSTM and other neural network architectures. *Neural Networks*, 18(5-6):602–610.

Bo Han and Timothy Baldwin. 2011. Lexical normalisation of short text messages: Makn sens a #twitter. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 368–378, Portland, Oregon, USA. Association for Computational Linguistics.

Rasoul Kaljahi, Jennifer Foster, Johann Roturier, Corentin Ribeyre, Teresa Lynn, and Joseph Le Roux. 2015. Foreebank: Syntactic analysis of customer support forums. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1341–1347, Lisbon, Portugal. Association for Computational Linguistics.

Mohammad Khan, Markus Dickinson, and Sandra Kuebler. 2013. Does size matter? text and grammar revision for parsing social media data. In *Proceedings of the Workshop on Language Analysis in Social Media*, pages 1–10, Atlanta, Georgia. Association for Computational Linguistics.

Eliyahu Kiperwasser and Yoav Goldberg. 2016. Simple and accurate dependency parsing using bidirectional LSTM feature representations. *TACL*, 4:313–327.

Lingpeng Kong, Nathan Schneider, Swabha Swayamdipta, Archna Bhatia, Chris Dyer, and Noah A. Smith. 2014. A dependency parser for tweets. In *Proceedings of the 2014 Conference*

on Empirical Methods in Natural Language Processing (EMNLP), pages 1001–1012, Doha, Qatar. Association for Computational Linguistics.

Marco Kuhlmann, Carlos Gómez-Rodríguez, and Giorgio Satta. 2011. Dynamic programming algorithms for transition-based dependency parsers. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 673–682, Portland, Oregon, USA. Association for Computational Linguistics.

Miryam de Lhoneux, Yan Shao, Ali Basirat, Eliyahu Kiperwasser, Sara Stymne, Yoav Goldberg, and Joakim Nivre. 2017a. From raw text to Universal Dependencies – look, no tags! In *Proceedings of the CoNLL 2017 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies.*, Vancouver, Canada.

Miryam de Lhoneux, Sara Stymne, and Joakim Nivre. 2017b. Arc-hybrid non-projective dependency parsing with a static-dynamic oracle. In *Proceedings of the The 15th International Conference on Parsing Technologies (IWPT).*, Pisa, Italy.

Chen Li and Yang Liu. 2014. Improving text normalization via unsupervised model and discriminative reranking. In *Proceedings of the ACL 2014 Student Research Workshop*, pages 86–93, Baltimore, Maryland, USA. Association for Computational Linguistics.

Chen Li and Yang Liu. 2015. Joint POS tagging and text normalization for informal text. In *Proceedings of the Twenty-Fourth International Joint Conference on Artificial Intelligence, IJCAI 2015, Buenos Aires, Argentina, July 25-31, 2015*, pages 1263–1269.

Yijia Liu, Yi Zhu, Wanxiang Che, Bing Qin, Nathan Schneider, and Noah A. Smith. 2018. Parsing tweets into Universal Dependencies. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 965–975. Association for Computational Linguistics.

David McClosky and Eugene Charniak. 2008. Self-training for biomedical parsing. In *Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics on Human Language Technologies: Short Papers*, pages 101–104. Association for Computational Linguistics.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *Proceedings of Workshop at ICLR*.

Dat Quoc Nguyen, Mark Dras, and Mark Johnson. 2017. A novel neural network model for joint POS tagging and graph-based dependency parsing. In *Proceedings of the CoNLL 2017 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*, pages 134–142.

Joakim Nivre, Zeljko Agić, Lars Ahrenberg, Lene Antonsen, Maria Jesus Aranzabe, Masayuki Asahara, Luma Ateyah, Mohammed Attia, Aitziber Atutxa, Liesbeth Augustinus, Elena Badmaeva, Miguel Ballesteros, Esha Banerjee, Sebastian Bank, Verginica Barbu Mititelu, John Bauer, Kepa Bengoetxea, Riyaz Ahmad Bhat, Eckhard Bick, Victoria Bobicev, Carl Börstell, Cristina Bosco, Gosse Bouma, Sam Bowman, Aljoscha Burchardt, Marie Candito, Gauthier Caron, Gülşen Cebiroğlu Eryiğit, Giuseppe G. A. Celano, Savas Cetin, Fabricio Chalub, Jinho Choi, Silvie Cinková, Çağrı Çöltekin, Miriam Connor, Elizabeth Davidson, Marie-Catherine de Marneffe, Valeria de Paiva, Arantza Diaz de Ilarraza, Peter Dirix, Kaja Dobrovoljc, Timothy Dozat, Kira Droganova, Puneet Dwivedi, Marhaba Eli, Ali Elkahky, Tomaž Erjavec, Richárd Farkas, Hector Fernandez Alcalde, Jennifer Foster, Cláudia Freitas, Katarína Gajdošová, Daniel Galbraith, Marcos Garcia, Moa Gärdenfors, Kim Gerdes, Filip Ginter, Iakes Goenaga, Koldo Gojenola, Memduh Gökırmak, Yoav Goldberg, Xavier Gómez Guinovart, Berta Gonzáles Saavedra, Matias Grioni, Normunds Grūzītis, Bruno Guillaume, Nizar Habash, Jan Hajič, Jan Hajič jr., Linh Hà Mỹ, Kim Harris, Dag Haug, Barbora Hladká, Jaroslava Hlaváčová, Florinel Hociung, Petter Hohle, Radu Ion, Elena Irimia, Tomáš Jelínek, Anders Johannsen, Fredrik Jørgensen, Hüner Kaşıkara, Hiroshi Kanayama, Jenna Kanerva, Tolga Kayadelen, Václava Kettnerová, Jesse Kirchner, Natalia Kotsyba, Simon Krek, Veronika Laippala, Lorenzo Lambertino, Tatiana Lando, John Lee, Phuong Lê H`ông, Alessandro Lenci, Saran Lertpradit, Herman Leung, Cheuk Ying Li, Josie Li, Keying Li, Nikola Ljubešić, Olga Loginova, Olga Lyashevskaya, Teresa Lynn, Vivien Macketanz, Aibek Makazhanov, Michael Mandl, Christopher Manning, Cătălina Mărănduc, David Mareček, Katrin Marheinecke, Héctor Martínez Alonso, André Martins, Jan Mašek, Yuji Matsumoto, Ryan McDonald, Gustavo Mendonça, Niko Miekka, Anna Missilä, Cătălin Mititelu, Yusuke Miyao, Simonetta Montemagni, Amir More, Laura Moreno Romero, Shinsuke Mori, Bohdan Moskalevskyi, Kadri Muischnek, Kaili Müürisep, Pinkey Nainwani, Anna Nedoluzhko, Gunta Nešpore-Bērzkalne, Luong Nguy˜ên Thị, Huy`ên Nguy˜ên Thị Minh, Vitaly Nikolaev, Hanna Nurmi, Stina Ojala, Petya Osenova, Robert Ostling, Lilja Ovrelid, Elena Pascual, Marco Passarotti, Cenel-Augusto Perez, Guy Perrier, Slav Petrov, Jussi Piitulainen, Emily Pitler, Barbara Plank, Martin Popel, Lauma Pretkalniņa, Prokopis Prokopidis, Tiina Puolakainen, Sampo Pyysalo, Alexandre Rademaker, Loganathan Ramasamy, Taraka Rama, Vinit Ravishankar, Livy Real, Siva Reddy, Georg Rehm, Larissa Rinaldi, Laura Rituma, Mykhailo Romanenko, Rudolf Rosa, Davide Rovati, Benoît Sagot, Shadi Saleh, Tanja Samardžić, Manuela Sanguinetti, Baiba Saulīte, Se-

bastian Schuster, Djamé Seddah, Wolfgang Seeker, Mojgan Seraji, Mo Shen, Atsuko Shimada, Dmitry Sichinava, Natalia Silveira, Maria Simi, Radu Simionescu, Katalin Simkó, Mária Simková, Kiril Simov, Aaron Smith, Antonio Stella, Milan Straka, Jana Strnadová, Alane Suhr, Umut Sulubacak, Zsolt Szántó, Dima Taji, Takaaki Tanaka, Trond Trosterud, Anna Trukhina, Reut Tsarfaty, Francis Tyers, Sumire Uematsu, Zdeňka Urešová, Larraitz Uria, Hans Uszkoreit, Sowmya Vajjala, Daniel van Niekerk, Gertjan van Noord, Viktor Varga, Eric Villemonte de la Clergerie, Veronika Vincze, Lars Wallin, Jonathan North Washington, Mats Wirén, Tak-sum Wong, Zhuoran Yu, Zdeněk Zabokrtský, Amir Zeldes, Daniel Zeman, and Hanzhi Zhu. 2017. Universal Dependencies 2.1. LINDAT/CLARIN digital library at the Institute of Formal and Applied Linguistics (’UFAL), Faculty of Mathematics and Physics, Charles University.

Olutobi Owoputi, Brendan O’Connor, Chris Dyer, Kevin Gimpel, Nathan Schneider, and Noah A. Smith. 2013. Improved part-of-speech tagging for online conversational text with word clusters. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 380–390, Atlanta, Georgia. Association for Computational Linguistics.

Slav Petrov, Dipanjan Das, and Ryan McDonald. 2012. A universal part-of-speech tagset. In *Proceedings of the Eight International Conference on Language Resources and Evaluation (LREC’12)*, Istanbul, Turkey. European Language Resources Association (ELRA).

Slav Petrov and Ryan McDonald. 2012. Overview of the 2012 shared task on parsing the web. In *Notes of the First Workshop on Syntactic Analysis of Non-Canonical Language (SANCL)*, volume 59.

Natalia Silveira, Timothy Dozat, Marie-Catherine de Marneffe, Samuel Bowman, Miriam Connor, John Bauer, and Christopher D. Manning. 2014. A gold standard dependency corpus for English. In *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC-2014)*.

Daniel Zeman, Martin Popel, Milan Straka, Jan Hajic, Joakim Nivre, Filip Ginter, Juhani Luotolahti, Sampo Pyysalo, Slav Petrov, Martin Potthast, Francis Tyers, Elena Badmaeva, Memduh Gokirmak, Anna Nedoluzhko, Silvie Cinkova, Jan Hajic jr., Jaroslava Hlavacova, Václava Kettnerová, Zdenka Uresova, Jenna Kanerva, Stina Ojala, Anna Missilä, Christopher D. Manning, Sebastian Schuster, Siva Reddy, Dima Taji, Nizar Habash, Herman Leung, Marie-Catherine de Marneffe, Manuela Sanguinetti, Maria Simi, Hiroshi Kanayama, Valeria dePaiva, Kira Droganova, Héctor Martínez Alonso, Çağrı Çöltekin, Umut Sulubacak, 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 Strnadová, Esha Banerjee, Ruli Manurung, Antonio Stella, Atsuko Shimada, Sookyoung Kwak, Gustavo Mendonca, 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*, pages 1–19, Vancouver, Canada. Association for Computational Linguistics.

Congle Zhang, Tyler Baldwin, Howard Ho, Benny Kimelfeld, and Yunyao Li. 2013. Adaptive parser-centric text normalization. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1159–1168, Sofia, Bulgaria. Association for Computational Linguistics.

A Supplemental Material

B Annotation Decisions

In this appendix we will give a short overview of annotation decisions. The Universal Dependencies English Web Treebank 2.1 (Silveira et al., 2014; Nivre et al., 2017) annotations are used as guidance for most annotation decisions. Since this treebank is sampled from different web domains, it already covers most phenomena occurring in Twitter data.

B.1 Tokenization

As a starting point, we used the tokenization from the previous annotation (Li and Liu, 2015). On top of this, we ran a simple rule-based tokenizer to make the data better suitable for syntactic annotation. Phrasal abbreviations (e.g. lol, smh) are treated as one token. We also included the normalization from the original corpora in the MISC column, which is manually corrected after tokenization (see Figure 3).

```
# text = damn im finna roll up again...
1    damn    Norm=damn
2    i       SpaceAfter=No;Norm=I
3    m       Norm=am
4    fin     SpaceAfter=No;Norm=going
5    na      Norm=to
6    roll    Norm=roll
7    up      Norm=up
8    again   Norm=again
9    ...     Norm=...
```

Figure 3: An example of tokenization in the CoNLL-U format (Only the ‘ID’, ‘FORM’ and ‘MISC’ column are shown here)

No sentence segmentation is performed on the input data because the Tweet-unit is inherent to this domain. Instead we use the `parataxis` relation to connect different utterances. The head of the first utterance is always the root, and all next utterance are dependents of this node, see Figure 4 for an example.

B.2 POS tags

Our parser does not make use of POS tags, but because they were already annotated and are closely related to the choice of dependency relations we corrected them during annotation. POS tags were first automatically mapped to universal tags (Petrov et al., 2012) and then manually corrected.

B.3 Unknown Words

If the annotator is unsure about the meaning of a word, other tweets containing the same word are searched and where necessary `www.urbandictionary.com` is consulted. If the annotator still could not understand the word, it is annotated as X with the `dep` relation. This only occurs five times in our data.

B.4 Emoticons, Emojis, URL's and Phrasal Abbreviations

Since words belonging to this category are often not syntactically connected, we annotate them as dependant of the head of the nearest utterance (see B.1). The relations and POS tags used are similar to the English Web Treebank: emoticons and emojis are a SYMB connected with relation `discourse`, URL's are annotated as X with relation `appos` and phrasal abbreviations like 'lol' and 'smh' are considered to be an INTJ with the `discourse` relation.

B.5 Domain Specific Tokens

Mentions are used in Twitter to direct tweets towards a specific person/account. They consists of the '@' symbol followed by the targeted username. Because mentions are used to focus a Tweet to a specific user we annotated it as PROPN with the relation `vocative`.

Hashtags are used to specify the topic or mood of the Tweet. They are often located at the end of the Tweet. Their usage is similar to interjections in the English Web Treebank, so they are annotated accordingly as INTJ and `discourse`.

A retweet is indicated by the token 'RT', which is usually found at the beginning of the Tweet. We tag it with the X tag and the `discourse` relation.

Because these phenomena are often not syntactically connected to the sentence, we connect them to the root. Note that all of these phenomena can also be used in (syntactic) context, then they are annotated accordingly (see example in Figure5).

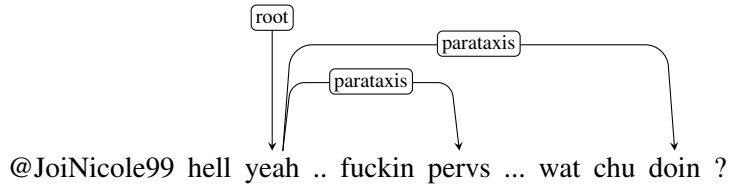

Figure 4: Annotation of the sentence "@JoiNicole99 hell yeah..fuckin pervs...watchu doin?", only relevant relations are shown.

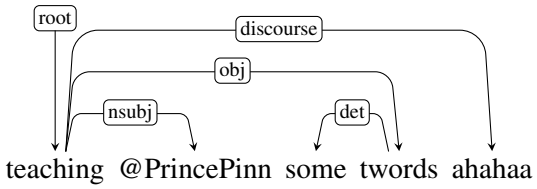

Figure 5: Annotation of the sentence "teaching @PrincePinn some twords ahahaa"