Arabic Tweets Treebanking and Parsing: A Bootstrapping Approach

Fahad Albogamy
School of Computer Science,
University of Manchester,
Manchester, M13 9PL, UK
albogamf@cs.man.ac.uk

Allan Ramsay
School of Computer Science,
University of Manchester,
Manchester, M13 9PL, UK
allan.ramsay@cs.man.ac.uk

Hanady Ahmed
CAS, Arabic department,
Qatar University,
2713, Al Hala St, Doha, Qatar
hanadyma@qu.edu.qa

Abstract

In this paper, we propose using a "bootstrapping" method for constructing a dependency treebank of Arabic tweets. This method uses a rule-based parser to create a small treebank of one thousand Arabic tweets and a data-driven parser to create a larger treebank by using the small treebank as a seed training set. We are able to create a dependency treebank from unlabelled tweets without any manual intervention. Experiments results show that this method can improve the speed of training the parser and the accuracy of the resulting parsers.

1 Introduction

Rule-based parsers have been developed and used for decades in the NLP community. In such parsers, linguistic rules are written to represent knowledge about the syntactic structure of a language. The parser produces the resulting parse trees by applying these rules to input sentences. It uses a dictionary or lexicon to store information about each word in input text before applying the linguistic rules. Although this kind of parser is widely-used in a variety of NLP systems to provide deep linguistic analyses, they have disadvantages: they are slow and it is time-consuming, expensive and tedious to construct dictionaries and to write the rules by expert linguists and hard to maintain them.

In recent years, data-driven parsers have been widely used due to the availability of annotated data such as the Penn Treebank (PTB) (Marcus et al., 1993) and the Penn Arabic Treebak (PATB) (Maamouri et al., 2004). These parsers are robust and produce state-of-the-art results compared to rule-based ones. However, the reliance on annotated data is one of the significant disadvantages of using data-driven parsers because a large amount of rich annotated data is not always available for many languages and domains due to various factors (Ramasamy and Žabokrtskỳ, 2011).

In the domain of Arabic tweets, we decided to use a data-driven approach, but for that a suitable collection of training data (treebank) should be available for training, and to our knowledge no such dataset has yet been created. For this reason, we have developed a bootstrapping technique for constructing a dependency treebank of the Arabic tweets by using a rule-based parser (RBP) and a data-driven parser (MaltParser). We are able to create a dependency treebank from unlabelled tweets without any manual intervention. Experiments results show that using MaltParser and RBP to construct a large training set (treebank) is better in terms of speed of training and accuracy of parsing than constructing it by using RBP only.

The rest of this paper is organised as follows: In Section 2, we give an overview of the related work, followed by a description of our bootstrapping approach in Section 3. In Section 4, we discuss the evaluation, results and their analysis. In Section 5, we reflect on the work described in the main paper.

2 Related Work

Although data-driven parsers have achieved state-of-the-art results on well-formed texts, they have not performed well on user-generated text because the nature of the texts found in user-contributed online forums rarely complies with the standard rules of the underlying language, which makes them challenging for traditional NLP tools, including data-driven approaches, even if domain adaptation techniques have been used (Seddah et al., 2012).

The nature of the text content of Arabic tweets

Proceedings of The Third Arabic Natural Language Processing Workshop (WANLP), pages 94–99,
Valencia, Spain, April 3, 2017.

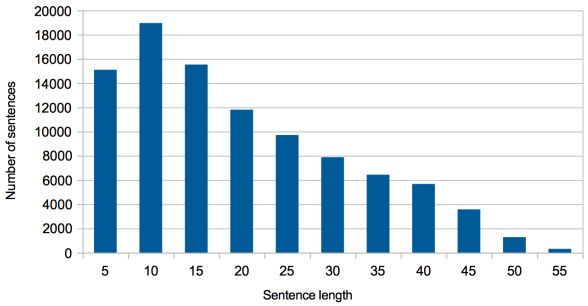

Figure 1: Length of tweets

is very mixed. It includes quite long sentences (20% of sentences are 25 or more words. See Figure 1) that are reasonably well-written (these can be considerably longer than sentences in English tweets because of the fact that Arabic is very economical with characters), very informal material with large numbers of errors and neologisms, and material which is just complete nonsense. Therefore, annotated data for well-edited genres such as PTB and PATB is not suited to the domain of tweets (Kong et al., 2014). As a result, there is an increasing interest in the development of treebanks of user-generated text. These are usually small manually annotated corpora: English Web Treebank (16K sentences) (Bies et al., 2012), Tweebank Treebank (929 sentences) (Kong et al., 2014), Spanish Web Treebank (2846 sentences) (Taulé et al., 2015) and French Social Media Bank (FSMB) (1700 sentences) (Seddah et al., 2012).

Our work is, to best of our knowledge, the first step towards developing a dependency treebank for Arabic tweets which can benefit a wide range of downstream NLP applications such as information extraction, machine translation and sentiment analysis. We explore the idea that producing a small treebank using a rule-based parser will suffice to train an initial MALT-style parser, which we can then use, in conjunction with the rule-based parser, to produce a much larger treebank which can be used to improve the performance of the base parser.

We used RBP to produce a dependency treebank partly to save effort instead of annotating it manually but also to remove the nonsense from the training data. If a tweet is just complete nonsense (and very many are!), then even if we ascribed a structure to them we would not want to use this to train our data-driven parser, since this structure will not be replicated in unseen tweets. Given that the RBP will fail to provide an analysis of such material, it acts as an automatic filter to remove it. RBP is, however, quite slow and to construct a large treebank using it is very time-consuming and difficult to make a big treebank. Therefore, we just use it to produce a small treebank as a seed training for MaltParser and using a bootstrapping technique to make a larger treebank out of it.

3 The Bootstrapping Method

Bootstrapping is used to create labelled training data from large amounts of unlabelled data (Cucerzan and Yarowsky, 2002).

We use a rule-based chart parser (RBP) similar to the one described in (Ramsay, 1999). This parser stops if it finds a complete analysis of a tweet: if does not find a complete analysis after a predetermined number of edges have been created, it stops and returns the largest non-overlapping potential fragments. We have no lexicon because of the rapidly changing nature of tweets and the presence of misspellings, both accidental and deliberate – tweets make use of a very open lexicon, to the point that even after you have looked at over a million tweets you will still find that one in ten words is unknown (Albogamy and Ramsay, 2015). Instead we use a tagger with a coarse-grain tagset, simply labelling all nouns as NN, all verbs as VB and so on. It is striking that even without fine-grained subcategorisation labels (e.g. between intransitive, transitive and sentential complement verbs) RBP produces good quality analyses when it produces anything at all.

Because RBP looks for maximal fragments it can also analyse tweets which actually consist of more than one sentence with no punctuation between them. The following tweet for instance consists of three separate sentences:

@alabbas75@DrA_Farouk235
هذا حق بحمد الله و نحب الله ورسوله محمد وكل نبياء والرسل ولعن الله خوارج العصر

The RBP returns three sub trees (largest fragments) that represent these sentences as we can see in Figure 2.

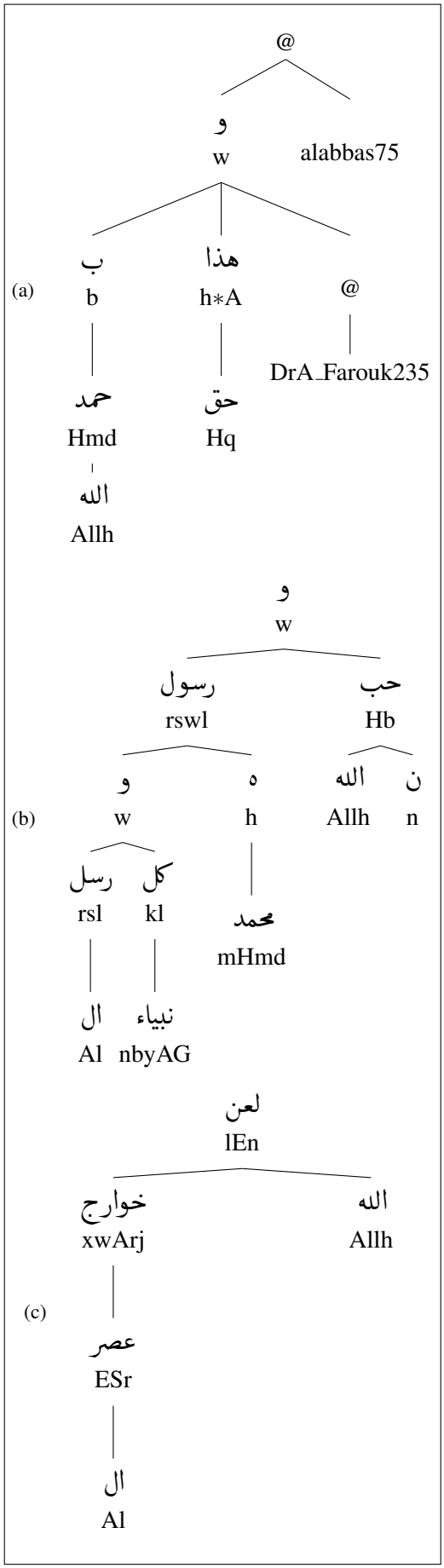

Figure 2: RBP segments a tweet to three fragments

We also use a data-driven parser (version of MaltParser) with three basic data structures a queue of unexamined words, a stack of words that have been considered but which have not been assigned heads, and a collection of `<head,relation,daughter>` triples; and with three basic actions, namely shift (move an item from the queue to the stack), `leftArc` (make the top item on the stack a daughter of the head of the queue and remove it from the stack), `rightArc` (make the head of the queue a daughter of the top item of the stack, remove the head of the queue and move the top item of the stack back to the queue) (Nivre et al., 2006).

The bootstrapping method (Figure 3) begins by using RBP to parse an existing POS-tagged corpus to create a small treebank of one thousand Arabic tweets. Then, MaltParser is trained on the small treebank which was created by RBP and used to parse a much larger set of POS-tagged Arabic tweets. During parsing, the RBP is used as a filter. To use it as a filter, we run the RBP but only allow hypothesises that correspond to links that were suggested by MaltParser so it produces a tree if and only if that tree was produced by MaltParser. As a result, all dependency analyses which do not conform to the defined language rules are omitted. All the resulting legal dependency trees are moved to the training pool to create a larger treebank. In Figure 4, MaltParser returns the whole tree for a tweets, but RBP agrees only upon the sub tree in the box. Finally, MaltParser is retrained on the large treebank. One potential drawback of the

1.Create dependency trees for a seed set of 1K Arabic tweets by using the rule-driven parser (an initial treebank).
2.Train MaltParser (a baseline model) on the initial treebank.
3.Parse 10K Arabic tweets with the baseline model and filter out all analyses which do not conform to the language rules by using RBP m1.
4.Train a new model on m1.
5.Test the new model on the reserved 1K test set.

Figure 3: Bootstrapping approach

bootstrapping technique is that the parser can reinforce its own bad behaviour. However, we control this by parsing a large amount of data and then

by using the largest legal fragments according to the grammar for which a well-formed parse is obtained and added to the training pool. By this way, we make sure that the parser will not learn from bad data.

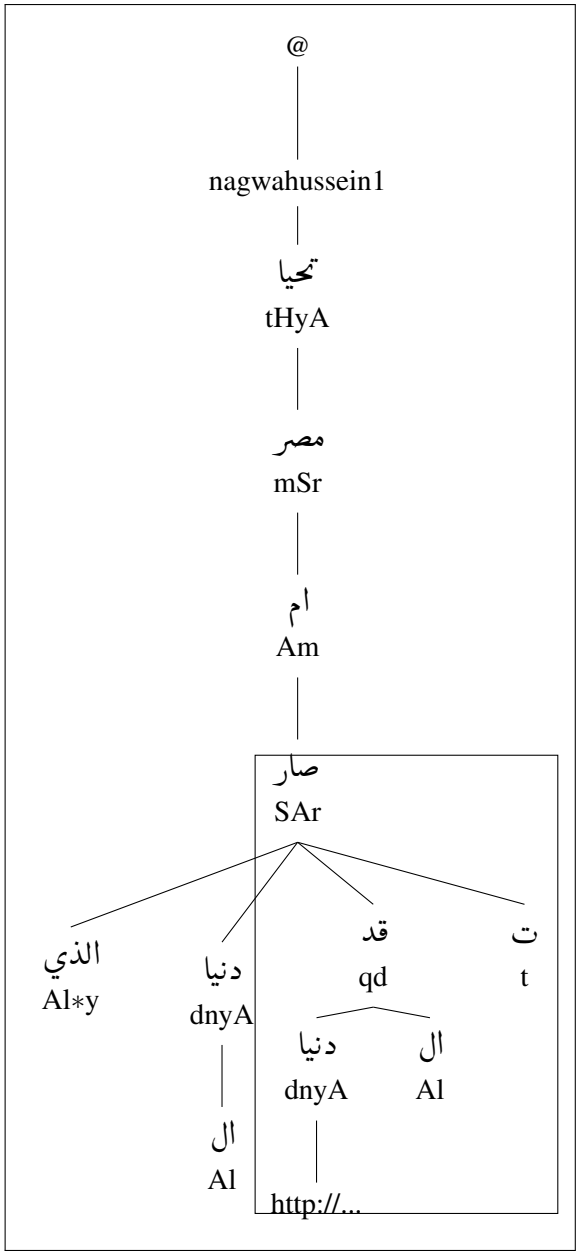

Figure 4: Using RBP as a Filter

4 Evaluation

The speed is a crucial factor to take into account when parsing Arabic tweets since there are millions of tweets that need to be parsed. Therefore, rule-based parsers are not suitable in this domain because they are slow (as mentioned in the literature and proved by the first experiment below). To parse Arabic tweets, we decided to use a data-driven parser, namely MaltParser. However, this kind of approaches needs training data (treebank). Therefore, we did two experiments to construct a treebank with a reasonable size. In the first experiment we used RBP only whereas in the second experiment we used RBP and MaltParser as described in Section 3.

4.1 Experimental Setup

The corpus from which we extract our dataset is an existing POS-tagged corpus taken from Twitter. Twitter Stream API was used to retrieve tweets from the Arabian Peninsula by using latitude and longitude coordinates of these regions since Arabic dialects in these regions share similar characteristics and they are the closest Arabic dialects to MSA. The corpus was tagged by the Arabic tweets tagger described in (Albogamy and Ramsay, 2016). We sampled 10K tagged tweets from the corpus to experiment on them.

4.2 Results

In our experiments, we used two different strategies to create a dependency treebank: using RBP only and using RBP and MaltParser in a bootstrapping technique. We do the evaluation on tweets that RBP gives analyses for. The accuracy of other tweets which do not have sensible analyses cannot be tested because it is impossible to say what the right answer would be. As we mentioned earlier, one of the reasons of using RBP is to eliminate nonsense tweets then it is reasonable to test only on filtered tweets because the vast majority of other tweets are nonsense and do not have sensible parsed trees.

In the first experiment, we used RBP to create a treebank from 10K tagged tweets. Then, we trained MaltParser on it. It took 20K seconds to construct the treebank and the accuracy of MaltParser after trained on the treebank is 68% (see Table 1). In the second experiment, we ran RBP on 1K tagged tweets to create a small treebank. It took 1K seconds to construct the small treebank and the accuracy of MaltParser after training on the small treebank (which we are using as a baseline model) is 64%. Then, the baseline model ran on 10K Arabic tweets and RBP is used to filter out all analyses which do not conform to the language rules and it took 4K seconds to construct larger treebank and the accuracy of MaltParser after training on the larger treebank is 71%. The

Size (words)	Strategy	Accuracy	Training time(sec)
162K	RBP	68%	20K

Table 1: Constructing treebank by using RBP

Size (words)	Strategy	Accuracy	Training time(sec)
15K	RBP	64%	1K
162K	Bootstrapping MALT+RBP	**71%**	4K

Table 2: Constructing treebank by bootstrapping

whole bootstrapping method took 5k seconds to create a reasonable size treebank.

In both experiments, we are able to create a dependency treebank from unlabelled tweets without any manual intervention. Experiments results show that using MaltParser and RBP in a bootstrapping approach to construct a large training set is better than constructing it by using RBP only in terms of the speed of training and the constructing the treebank (20K seconds to construct a 162K treebank just using RBP, 5K seconds to construct a treebank of the same size using RBP to analyse 15K words and MaltParser filtered by RBP to analyse the remaining 147K) and the accuracy of parsing (see Table 2). We tested on a reserved testset of 1K tweets, using 5-fold cross validation.

The results of the tests on our parsing approach yield an accuracy of 71%. We have compared our parsing accuracy and the size of treebank with similar work for English tweets (Kong et al., 2014) and French social media data (Seddah et al., 2012). Those two parsers yield accuracies 80% and 67.8% respectively and the size of our treebank is much larger than their treebanks. Our results show improvements over the performance of French parsing for social media data and it is not far behind English parsing for tweets. Moreover, we did not use any manual intervention for creating our treebank whereas they used human annotated data.

5 Conclusion

We have described a bootstrapping technique, which uses a rule-based parser to construct a small treebank of Arabic tweets based on an existing POS-tagged corpus, which then trains a data-driven parser on this treebank to parse a much larger pool of unlabeled Arabic tweets to create a large treebank. The results reported from the evaluation of this approach show that it can make a reasonable size dependency treebank that conforms to the rules of the rule-based parser and improve the speed of training and the accuracy of the parsing. This method does not require annotated data or human-supervised training.

Acknowledgments

The authors would like to thank the anonymous reviewers for their encouraging feedback and insights. Fahad would also like to thank King Saud University for their financial support. Hanady Ahmed and Allan Ramsay's contribution to this publication was made possible by the NPRP award [NPRP 7-1334-6-039 PR3] from the Qatar National Research Fund (a member of The Qatar Foundation). The statements made herein are solely the responsibility of the authors.

References

Alfred V. Aho and Jeffrey D. Ullman. 1972. *The Theory of Parsing, Translation and Compiling*, volume 1. Prentice-Hall, Englewood Cliffs, NJ.

Fahad Albogamy and Allan Ramsay. 2015. POS tagging for Arabic tweets. *Recent Advances in Natural Language Processing*, page 1.

Fahad Albogamy and Allan Ramsay. 2016. Fast and robust POS tagger for Arabic tweets using agreement-based bootstrapping. *LREC*.

American Psychological Association. 1983. *Publications Manual*. American Psychological Association, Washington, DC.

Ann Bies, Justin Mott, Colin Warner, and Seth Kulick. 2012. English web treebank. *Linguistic Data Consortium, Philadelphia, PA*.

Ashok K. Chandra, Dexter C. Kozen, and Larry J. Stockmeyer. 1981. Alternation. *Journal of the Association for Computing Machinery*, 28(1):114–133.

Silviu Cucerzan and David Yarowsky. 2002. Bootstrapping a multilingual part-of-speech tagger in one person-day. In *proceedings of the 6th conference on Natural language learning-Volume 20*, pages 1–7. Association for Computational Linguistics.

Dan Gusfield. 1997. *Algorithms on Strings, Trees and Sequences*. Cambridge University Press, Cambridge, UK.

Lingpeng Kong, Nathan Schneider, Swabha Swayamdipta, Archna Bhatia, Chris Dyer, and Noah A Smith. 2014. A dependency parser for tweets.

Mohamed Maamouri, Ann Bies, Tim Buckwalter, and Wigdan Mekki. 2004. The Penn Arabic Treebank: Building a large-scale annotated Arabic corpus. In *NEMLAR conference on Arabic language resources and tools*, volume 27, pages 466–467.

Mitchell P Marcus, Mary Ann Marcinkiewicz, and Beatrice Santorini. 1993. Building a large annotated corpus of English: The Penn Treebank. *Computational linguistics*, 19(2):313–330.

Joakim Nivre, Johan Hall, and Jens Nilsson. 2006. Maltparser: A data-driven parser-generator for dependency parsing. In *Language Resources and Evaluation*.

Loganathan Ramasamy and Zdeněk Žabokrtskỳ. 2011. Tamil dependency parsing: results using rule based and corpus based approaches. In *International Conference on Intelligent Text Processing and Computational Linguistics*, pages 82–95. Springer.

Allan Ramsay. 1999. Parsing with discontinuous phrases. *Natural Language Engineering*, 5(03):271–300.

Djamé Seddah, Benoit Sagot, Marie Candito, Virginie Mouilleron, and Vanessa Combet. 2012. The French social media bank: a treebank of noisy user generated content. In *COLING 2012-24th International Conference on Computational Linguistics*.

Mariona Taulé, M Antonia Martí, Ann Bies, Montserrat Nofre, Aina Garí, Zhiyi Song, Stephanie Strassel, and Joe Ellis. 2015. Spanish treebank annotation of informal non-standard web text. In *International Conference on Web Engineering*, pages 15–27. Springer.