Proceedings of the 12th Conference on Language Resources and Evaluation (LREC 2020), pages 5219–5224
Marseille, 11–16 May 2020

Training a Swedish Constituency Parser on Six Incompatible Treebanks

Richard Johansson[♣], **Yvonne Adesam**[♠]
[♣]Department of Computer Science and Engineering, University of Gothenburg, Sweden
[♠]Språkbanken, Department of Swedish, University of Gothenburg, Sweden
richard.johansson@gu.se, yvonne.adesam@gu.se

Abstract

We investigate a transition-based parser that uses *Eukalyptus*, a function-tagged constituent treebank for Swedish which includes discontinuous constituents. In addition, we show that the accuracy of this parser can be improved by using a multitask learning architecture that makes it possible to train the parser on additional treebanks that use other annotation models.

Keywords: constituency parsing, Swedish, multitask learning

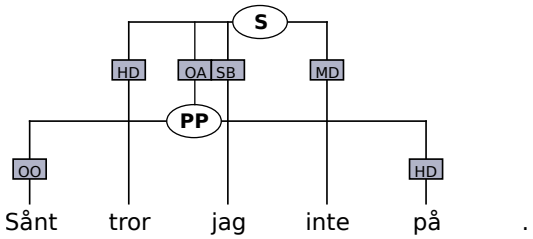
1. Introduction

Syntactic parsing is a widely used intermediate step in several natural language processing (NLP) tasks. For the last couple of decades, syntactic parsing has largely been based on machine learning systems, trained in a supervised fashion using large collections of hand-annotated sentences – *treebanks*. While the accuracy of automatic parsers had largely reached a plateau a few years ago, the introduction of deep learning techniques has led to recent improvements in accuracy.

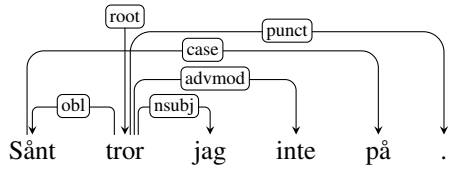
Hand-annotated treebanks are based on linguistic models that can vary drastically. The most widely noted difference among annotation models for syntax is probably the divergence between *constituency models* that conceive of the sentence structure as consisting of hierarchically organized *phrases* or *constituents*, and *dependency models* that represent the sentence structure as a graph where the tokens (words) are the edges. To exemplify this divergence, Figure 1 shows the Swedish sentence *Sånt tror jag inte på.* 'I don't believe in that kind of stuff.' annotated according to constituency and dependency models. However, even within one class of models, there are many theoretical design choices that can vary between different treebanks.

For syntactic parsing, as for supervised learning in general, the availability of a substantial number of annotated examples is crucial. Manual annotation is time-consuming and re-using previously annotated treebanks would therefore be beneficial. However, the variability of treebanks is a major nuisance: if we want to develop a parser that uses some particular annotation model, it is not evident that any other available treebanks can be used. This is an obvious problem for a language such as Swedish, for which the largest treebank (Nilsson et al., 2005) is significantly smaller than treebanks for e.g. English, Chinese, and German. At the same time multiple treebanks with different types of annotation – different types of encoded linguistic knowledge – are available. We are aware of at least *five* unique treebanks for Swedish, most of which are annotated according to their own respective models (some of them more than one model).

Is there a remedy: can we develop an approach that can utilize additional treebanks? For non-neural, feature-based dependency parsers, Johansson (2013) proposed two



(a) Constituency annotation.



(b) Dependency annotation.

Figure 1: An example sentence annotated according to two syntactic annotation models.

feature-based approaches to multi-treebank training and evaluated them using treebank pairs in four languages. The first approach was based on *multitask learning* (Caruana, 1997) using a shared feature representation (Daumé III, 2007), while the second approach used *stacking* or *guided parsing* (Nivre and McDonald, 2008). A more recent approach, based on *treebank embeddings* in a neural dependency parser, was presented by Stymne et al. (2018); however, their approach has so far only been evaluated for sets of treebanks that are very close in annotation style.

In this work, we present the first results for parsing with the *Eukalyptus* treebank of written Swedish (Adesam et al., 2018), a function-tagged constituency treebank including discontinuous constituents. Furthermore, we show how a transition-based neural parser can be improved by using a multitask architecture that allows us to train the parser using a number of *auxiliary* treebanks, some of which are dependency treebanks. We are not aware of any previous work that has utilized constituency treebanks in this context, and especially not in a training process that uses *both* constituency and dependency treebanks.

2. Neural Transition-based Parsing

The parsers considered in this work belong to the class of *transition-based* parsers. In this approach, the parser builds the output structure in a step-by-step fashion by executing actions in a state machine. This state machine uses a stack to store partially built structures and a buffer that keeps the remaining part of the input. Our parser uses a transition system based on the *shift/promote/adjoin* system for constituency parsing introduced by Cross and Huang (2016), to which Stanojević and Garrido Alhama (2017) added a swap transition (Nivre, 2009) to allow for discontinuous constituents (e.g. the PP in Figure 1a).

This transition system allows five different actions: SHIFT, which moves an item from the buffer onto the stack; PROMOTE, which takes the top item of the stack and starts to build a constituent; LEFT-ADJOIN and RIGHT-ADJOIN, which attach an item to the left or to the right of a constituent, respectively; and SWAP, which moves the second-to-last item of the stack back into the buffer. Figure 2 states these actions formally; for brevity, we omit the RIGHT-ADJOIN action and the preconditions that determine whether an action is applicable.

$$\textsc{Shift} \quad \frac{\langle S,\, x|B \rangle}{\langle S|x,\, B \rangle}$$

$$\textsc{Promote}[C] \quad \frac{\langle S|x,\, B \rangle}{\langle S \,|\, C(x),\, B \rangle}$$

$$\textsc{Left-adjoin} \quad \frac{\langle S|x|C(X),\, B \rangle}{\langle S \,|\, C(x|X),\, B \rangle}$$

$$\textsc{Swap} \quad \frac{\langle S|x_1|x_2,\, B \rangle}{\langle S|x_2,\, x_1|B \rangle}$$

Figure 2: Actions in the transition system.

The model used in the parser by Stanojević and Garrido Alhama (2017) can be seen as a constituent-based variation of the model by Dyer et al. (2015). It relies on several variants of the long short-term memory (LSTM), a well-known model for representing sequential computations (Hochreiter and Schmidhuber, 1997). The components of the parsing model are the following:

- *word representations*: a bidirectional LSTM applied to word and part-of-speech tag embeddings;[1]
- *constituent representations*: complex linguistic units are represented by applying a *tree LSTM* (Tai et al., 2015) compositionally;
- *state representations*: two separate *stack LSTMs* (Dyer et al., 2015) represent the stack and buffer, respectively;[2]
- *action selector*: feedforward multiclass classifiers that determine the next action to execute, and the constituent label.

Figure 3 shows the representation of an intermediate state when selecting an action during the generation of the tree in Figure 1a; the correct action in this situation would be to SWAP the token *Sånt* back into the buffer, in order to build the discontinuous constituent.

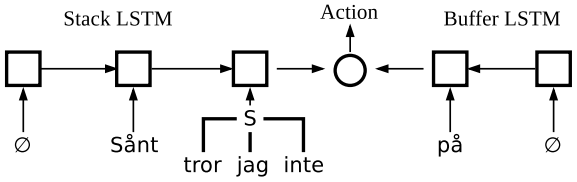


Figure 3: Representation of the parser's state for determining the next action.

The parser is trained using a static oracle, since no efficient dynamic oracle is known for this transition system (or its dependency counterpart, the arc-standard system). We note that dynamic oracles are available for closely related transition systems (Goldberg and Nivre, 2013), but require special care when a SWAP transition is used (de Lhoneux et al., 2017). We leave dynamic oracle training to future work.

We extended the model by Stanojević and Garrido Alhama (2017) so that it outputs function *edge labels* as well, e.g. OO, HD, OA, etc. in Figure 1a. The edge labels are generated when executing the PROMOTE, LEFT-ADJOIN, and RIGHT-ADJOIN actions. Taking edge labels into account required modifications of the tree LSTM that builds constituent representations compositionally, as well as a new feedforward unit to predict the edge labels.

3. Multitask Learning for Neural Network Parsers

Johansson (2013) applied multitask learning, using a feature representation that is partly shared between tasks (Daumé III, 2007), to train a feature-based dependency parser using multiple incompatible treebanks. This approach is not directly applicable here, since the model does not use an explicit feature representation. Instead, we follow recent work in multitask learning for NLP (Ruder, 2017), and train parsers for the different treebanks where some components are shared between the tasks; the idea is that the shared parts of the models will represent the commonalities between the tasks, abstracting away from the low-level peculiarities.

In this work, we follow the simple intuition that any component that does not explicitly depend on a treebank annotation model is shared between the parsing models for the different treebanks. That is, the representation models for words (word embedding and bidirectional LSTM) and the buffer are shared, while the representations for constituents and the stack, as well as the action selector, are kept separate for each treebank. We leave a full exploration of the best way to select the shared components to future work.

[1] The word embeddings were trained from scratch and we did not see any improvements when using pre-trained embeddings.

[2] Dyer et al. (2015) also use an LSTM representing the action sequence. Like Stanojević and Garrido Alhama (2017), we did not see any improvements by including this additional LSTM.

This architecture in principle allows us to use *different* types of transition systems, for instance for processing constituency and dependency treebanks differently. In this work we simply treat dependency trees as a special case of constituent tree, using a "dummy" constituent label. We can then apply the same type of transition system and learning model when processing all treebanks.

4. The Eukalyptus Treebank of Swedish

Early treebanks for Swedish such as *Talbanken* (Einarsson, 1976) and *Syntag* (Järborg, 1986) were annotated with constituency structures, and the recently created treebank *Eukalyptus* (Adesam et al., 2018) also has a constituency-based syntactic description. This treebank contains around 100 000 tokens, distributed over approximately 5 500 sentences. These were chosen from five different contemporary Swedish text types, which are public domain. They range from formal to informal, and from informative to entertaining, including both news text and blogs.

Eukalyptus has been manually annotated with part-of-speech tags, morphological features, word senses, and syntactic structure. The syntactic description is similar to for example the German NEGRA/TIGER scheme (Brants et al., 1999). Tokens are connected into phrases, and each child (edge) has a function label. Phrases may be discontinuous. Secondary edges are used to mark various types of shared information in constructions such as coordination and control. The syntactic description is described in Adesam et al. (2015a). The treebank also uses special phrase labels to connect multiword units, detailed in Adesam et al. (2015b). We distinguish two types of multiword units: *analyzable* – which have an internal syntactic representation, and where a multiword node is added to attach the multiword label, gathering the parts of the multiword unit with secondary edges – and *unanalyzable* – which do not receive a syntactic analysis but are attached to the tree through their multiword node.

A full example tree can be seen in Figure 4. The parts of speech together with the phrases and syntactic functions create a whole, where the different levels of information have complementary roles. However, for the current paper we will focus on parts of the syntactic annotation.

5. Parsing with Different Types of Treebanks

As mentioned, there are several treebanks available for Swedish, annotated using several different annotation models. In this work, our primary goal is to build and improve a parser for the *Eukalyptus* treebank described in Section 4.. To achieve this we will experiment with adding different types of treebanks during training.

5.1. Treebanks Used in the Experiments

The *Eukalyptus* treebank is the *primary* treebank, which we use in the single-task experiment and that is used to compute the evaluation score. As *auxiliary* treebanks in multitask training, we use five different treebanks. Two of them are constituency-based: *Talbanken05*, which is a modernized conversion (Nilsson et al., 2005) of the original Talbanken (Einarsson, 1976); and *Syntag* (Järborg, 1986). These two treebanks are annotated in a fairly flat style, in a manner similar to *Eukalyptus* and the TIGER treebank (Brants et al., 1999).

The remaining treebanks are dependency treebanks, all part of the *Universal Dependencies* project (Nivre et al., 2016); we used all three available Swedish UD treebanks: *Talbanken*, *LinES*, and *PUD*. Table 1 shows the sizes and structural properties of all treebanks: whether they allow discontinuities and whether they include constituent or edge labels.

Treebank	Size	Discont.	Const. labels	Edge labels
Eukalyptus	100 379	✓	✓	✓
Talbanken05	197 123	✓	✓	✓
Syntag	105 785			✓
Talbanken UD	96 819	✓		✓
LinES UD	79 816	✓		✓
PUD	19 074	✓		✓

Table 1: Treebanks used in the experiments.

Gold-standard part-of-speech annotations are available in all the treebanks, but use different annotation standards. Instead of using the gold-standard tags, we carried out the experiments using tags predicted automatically by HunPos (Halácsy et al., 2007), using the tagset defined by the Stockholm–Umeå corpus (Ejerhed et al., 1992). In addition to being a more realistic setup, this had the added advantage that all treebanks could use the same set of part-of-speech tags. We leave multitask learning at the part-of-speech level to future work.

In addition, a number of adaptations of the Eukalyptus treebank are carried out. First, secondary edges are ignored. In many cases this does not change the information available in the treebank. However, for shared information in for example coordinations, there are now parts missing. In addition, removing secondary edges affects the annotation of multiword units. For analyzed multiwords, only a unary node remains of the multiword annotation, which is therefore removed. Taking the secondary edges and analyzed multiword units into account is an interesting avenue for future research.

Finally, punctuation, interjections, and discourse particles – items that were not part of the regular tree – are attached to make each sentence a complete tree. These units are attached as high as possible without introducing discontinuities.

5.2. Experimental Protocol

We evaluate the baseline (the parser described in §2.), which uses just the primary treebank, as well as three different multitask learning setups where auxiliary treebanks are added (§3.): just dependency treebanks, just constituency treebanks, and all auxiliary treebanks. For each parser, we carry out a 10-fold cross-validation[3] over the primary treebank; for each fold, we run the parser twice, using different

[3]The treebank consists of five sections of roughly equal size, corresponding to different genres. Because we do not shuffle the

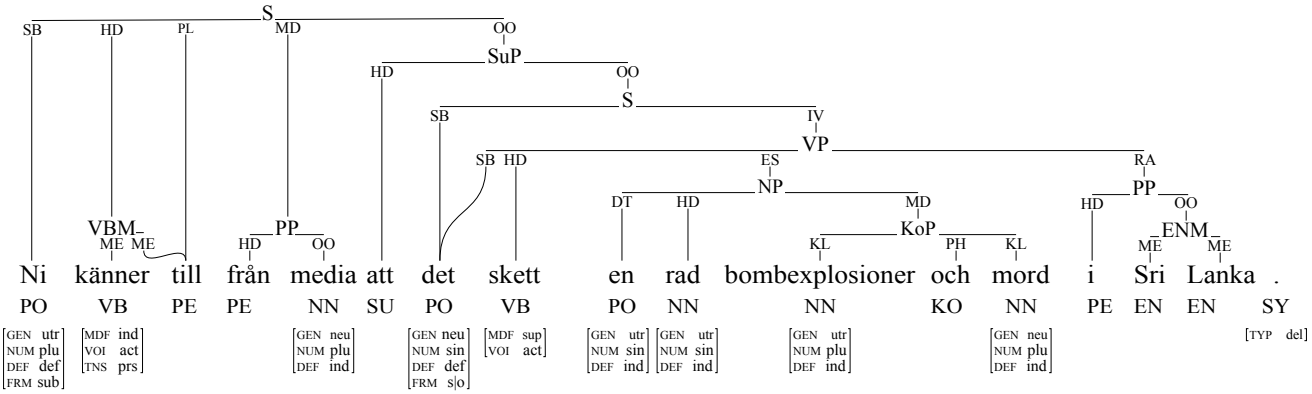


Figure 4: An annotated *Eukalyptus* tree including all layers of annotation.

random seeds, and select the model that gives the highest score on a development set.

The experiment uses two different evaluation metrics: (1) the F-score for finding and labeling constituents, such as `S` (sentence) and `PP` (prepositional phrase) in Figure 1a; (2) the accuracy of labeling edges, such as `SB` (subject) and `OO` (object) in Figure 1a. When evaluating, predicted and a gold-standard constituents are considered equal if their labels match and their yields (sets of covered tokens) are equal. Following Blaheta and Charniak (2000), only the correctly parsed constituents are included when computing the edge labeling accuracy. We used the evaluation module of DISCODOP (van Cranenburgh et al., 2016) to compute all scores.

5.3. Results

Table 2 shows the results of the evaluation. The constituent F-score for the baseline parser, which was trained on the *Eukalyptus* treebank only, is about 6 absolute points lower than that reported for German by Stanojević and Garrido Alhama (2017). The lower score is unsurprising since this treebank is about 9 times smaller than the TIGER treebank used in their experiments.

Moreover, the results show that all three multi-treebank parsers improve over the single-treebank baseline: they are significantly better at finding constituents.[4] The edge labeling accuracy, however, does not show any significant improvements.

It seems that the main takeaway is "the more, the better," as the top-scoring setup uses all five auxiliary treebanks. However, we get a significantly stronger improvement from the constituency treebanks than from the dependency treebanks. This may partly be a size effect, because the combination of constituency treebanks is larger, but may also partly be because these treebanks provide a training signal that is more relevant when learning to create and label constituents in the target treebank.

Treebanks	F-score	Acc.
Primary	71.30	88.26
Primary + Dep	71.91	88.47
Primary + Const	72.66	88.44
Primary + Const + Dep	72.86	88.43

Table 2: Evaluation scores for the baseline parser and three different multi-treebank parsers.

6. Conclusions

We trained a parser on *Eukalyptus*, a Swedish function-tagged constituency treebank including discontinuous constituents. The baseline parser is an extension of the implementation by Stanojević and Garrido Alhama (2017) that allows edge labels to be predicted. Of more general interest, we showed that this parser can be improved by adding *auxiliary* treebanks in a multitask learning setup. Even if we have the goal of predicting outputs that adhere to a specific annotation model, treebanks that are incompatible with our target model do not need to be wasted. Constituent treebanks seem to be more useful as auxiliary treebanks when training a constituent parser, although the dependency treebanks also give an improvement.

This is our first investigation of multi-treebank training for neural transition-based constituency parsers and it remains an open research problem to fully explore the spectrum of sharing architectures and find the one that best utilizes the auxiliary treebanks. The importance of this design choice has been discussed extensively for other NLP tasks (Ruder, 2017); for instance, Søgaard and Goldberg (2016) designed a carefully crafted sharing architecture for sequence labeling tasks. Ruder et al. (2017) discuss a method to *learn* the sharing architecture. In addition, it would be useful to investigate how the utility of a multitask setup is affected by the size of the primary treebank (Johansson, 2013).

sentences before dividing into cross-validation folds, each test fold will typically be sampled from a single genre, which will then also be under-represented in the training set. It seems likely that shuffling would give us slightly higher evaluation scores.

[4]The p-values are less than 0.05 in all comparisons to the single-task baseline. We used an approximate randomization test.

Acknowledgments

We are grateful to Miloš Stanojević and Raquel Alhama for making their implementation available. RJ was funded by the Swedish Research Council (VR) under grant 2013–4944. The *Eukalyptus* corpus was developed in a project funded by Riksbankens Jubilemsfond, grant In13-0320:1.

7. Bibliographical References

Adesam, Y., Bouma, G., and Johansson, R. (2015a). Defining the Eukalyptus forest – the Koala treebank of Swedish. In *Proceedings of the 20th Nordic Conference of Computational Linguistics*, pages 1–10, Vilnius, Lithuania.

Adesam, Y., Bouma, G., and Johansson, R. (2015b). Multiwords, word senses and multiword senses in the Eukalyptus treebank of written Swedish. In *Proceedings of the 14th International Workshop on Treebanks and Linguistic Theories*, Warsaw, Poland.

Blaheta, D. and Charniak, E. (2000). Assigning function tags to parsed text. In *Proceedings of the 1st Meeting of the North American Chapter of the Association for Computational Linguistics*, pages 234–240, Seattle, United States.

Brants, T., Hendriks, R., Kramp, S., Krenn, B., Preis, C., Skut, W., and Uszkoreit, H. (1999). Das NEGRA-Annotationsschema. Technical report, Universität des Saarlandes University, Dept of Computerlinguistik, Saarbrücken.

Caruana, R. (1997). Multitask learning. *Machine Learning*, 28(1):41–75.

van Cranenburgh, A., Scha, R., and Bod, R. (2016). Data-oriented parsing with discontinuous constituents and function tags. *Journal of Language Modelling*, 4(1):57–111.

Cross, J. and Huang, L. (2016). Incremental parsing with minimal features using bi-directional LSTM. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 32–37, Berlin, Germany.

Daumé III, H. (2007). Frustratingly easy domain adaptation. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 256–263, Prague, Czech Republic.

Dyer, C., Ballesteros, M., Ling, W., Matthews, A., and Smith, N. A. (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.

Goldberg, Y. and Nivre, J. (2013). Training deterministic parsers with non-deterministic oracles. *Transactions of the Association for Computational Linguistics*, 1:403–414.

Halácsy, P., Kornai, A., and Oravecz, C. (2007). HunPos – an open source trigram tagger. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics Companion Volume Proceedings of the Demo and Poster Sessions*, pages 209–212, Prague, Czech Republic.

Hochreiter, S. and Schmidhuber, J. (1997). Long short-term memory. *Neural computation*, 9(8):1735–1780.

Johansson, R. (2013). Training parsers on incompatible treebanks. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 127–137, Atlanta, United States.

de Lhoneux, M., Stymne, S., and Nivre, J. (2017). Arc-hybrid non-projective dependency parsing with a static-dynamic oracle. In *Proceedings of the 15th International Conference on Parsing Technologies*, pages 99–104, Pisa, Italy.

Nilsson, J., Hall, J., and Nivre, J. (2005). MAMBA meets TIGER: Reconstructing a Swedish treebank from antiquity. In *Proceedings of NODALIDA Special Session on Treebanks*, Joensuu, Finland.

Nivre, J. and McDonald, R. (2008). Integrating graph-based and transition-based dependency parsers. In *Proceedings of ACL-08: HLT*, pages 950–958, Columbus, United States.

Nivre, J. (2009). Non-projective dependency parsing in expected linear time. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, pages 351–359, Suntec, Singapore.

Ruder, S., Bingel, J., Augenstein, I., and Søgaard, A. (2017). Learning what to share between loosely related tasks. *CoRR*, abs/1705.08142.

Ruder, S. (2017). An overview of multi-task learning in deep neural networks. *CoRR*, abs/1706.05098.

Søgaard, A. and Goldberg, Y. (2016). Deep multi-task learning with low level tasks supervised at lower layers. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 231–235, Berlin, Germany.

Stanojević, M. and Garrido Alhama, R. (2017). Neural discontinuous constituency parsing. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1666–1676, Copenhagen, Denmark.

Stymne, S., de Lhoneux, M., Smith, A., and Nivre, J. (2018). Parser training with heterogeneous treebanks. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 619–625, Melbourne, Australia.

Tai, K. S., Socher, R., and Manning, C. D. (2015). Improved semantic representations from tree-structured long short-term memory networks. 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 1556–1566, Beijing, China.

8. Language Resource References

Adesam, Y., Bouma, G., Johansson, R., Borin, L., and Forsberg, M. (2018). Eukalyptus treebank of written

Swedish. Språkbanken, https://spraakbanken.gu.se/eng/resource/eukalyptus.

Einarsson, J. (1976). Talbanken. Lund University, distributed by Språkbanken, https://spraakbanken.gu.se/eng/resource/talbanken.

Ejerhed, E., Källgren, G., Wennstedt, O., and Åström, M. (1992). The linguistic annotation system of the Stockholm-Umeå corpus project – description and guidelines. Technical report, Department of Linguistics, Umeå University. Distributed by Språkbanken, https://spraakbanken.gu.se/eng/resource/suc3.

Järborg, J. (1986). Syntag treebank. Språkbanken, https://spraakbanken.gu.se/eng/resource/syntag.

Nivre, J., de Marneffe, M.-C., Ginter, F., Goldberg, Y., Hajič, J., Manning, C. D., McDonald, R., Petrov, S., Pyysalo, S., Silveira, N., Tsarfaty, R., and Zeman, D. (2016). Universal dependencies v1: A multilingual treebank collection. In Nicoletta Calzolari (Conference Chair), et al., editors, *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC 2016)*.