German and French Neural Supertagging Experiments for LTAG Parsing

Tatiana Bladier **Andreas van Cranenburgh** **Younes Samih** **Laura Kallmeyer**
Heinrich Heine University of Düsseldorf
Universitätsstraße 1, 40225 Düsseldorf, Germany
{bladier,cranenburgh,samih,kallmeyer}@phil.hhu.de

Abstract

We present ongoing work on data-driven parsing of German and French with Lexicalized Tree Adjoining Grammars. We use a supertagging approach combined with deep learning. We show the challenges of extracting LTAG supertags from the French Treebank, introduce the use of *left-* and *right-sister-adjunction*, present a neural architecture for the supertagger, and report experiments of n-best supertagging for French and German.

1 Introduction

Lexicalized Tree Adjoining Grammar (LTAG; Joshi and Schabes, 1997) is a linguistically motivated grammar formalism. Productions in an LTAG support an extended domain of locality (EDL). This allows them to express linguistic generalizations that are not captured by typical statistical parsers based on context-free grammars or dependency parsing. Each derivation step is triggered by a lexical element and a principled distinction is made between its arguments and modifiers, which is reflected in richer derivations. This has applications in the context of other tasks which can make use of linguistically rich analyses, such as frame semantic parsing or semantic role labeling (Sarkar, 2007). On the other hand, the increased expressiveness of LTAG makes efficient parsing and statistical estimations more challenging.

Previous work (Bangalore and Joshi, 1999; Sarkar, 2007) has shown that the task of parsing with LTAGs can be facilitated through the intermediate step of *supertagging*—a task of assigning possible *supertags* (i.e. elementary trees) for each word in a given sentence (Chen, 2010). Supertagging has been referred to as "almost parsing" (Bangalore and Joshi, 1999), since supertagging performs a large part of the task of syntactic disambiguation and increases the parsing efficiency by lexicalizing syntactic decisions before moving on to the more expensive polynomial parsing algorithm (Sarkar, 2007).

Recently, several papers proposed neural architectures for supertagging with Combinatory Categorial Grammar (CCG; Lewis et al., 2016; Vaswani et al., 2016) and LTAG (Kasai et al., 2017). Supertagging with LTAG is more challenging than with CCG due to a higher number of supertags (counting on average 4000 distinct supertags for LTAGs). Also, almost half of the LTAG supertags occur only once. Nevertheless, the reported neural supertagging approach for LTAG (Kasai et al., 2017) reaches an accuracy of 88-90 % for English (compared to over 95 % for CCG). In this paper we apply a similar recurrent neural architecture to supertagging with LTAGs based on Samih (2017) and Kasai et al. (2017) to German and French data and compare against previously reported results. For the German data, we compare our results to the LTAG supertaggers reported in Bäcker and Harbusch (2002) and Westburg (2016). To our knowledge, no results for French supertagging based on LTAG or CCG have been reported so far.

2 Neural Supertagging with LTAGs

2.1 Lexicalized Tree Adjoining Grammar

A *Tree Adjoining Grammar* (TAG; Joshi and Schabes, 1997) is a linguistically and psychologically motivated tree rewriting formalism (Sarkar, 2007). A TAG consists of a finite set of *elementary trees*, which can be combined to form larger trees via the operations of *substitution* (replacing a leaf node marked with $\downarrow$ with an initial tree) or *adjunction* (replacing an internal node with an auxiliary tree).

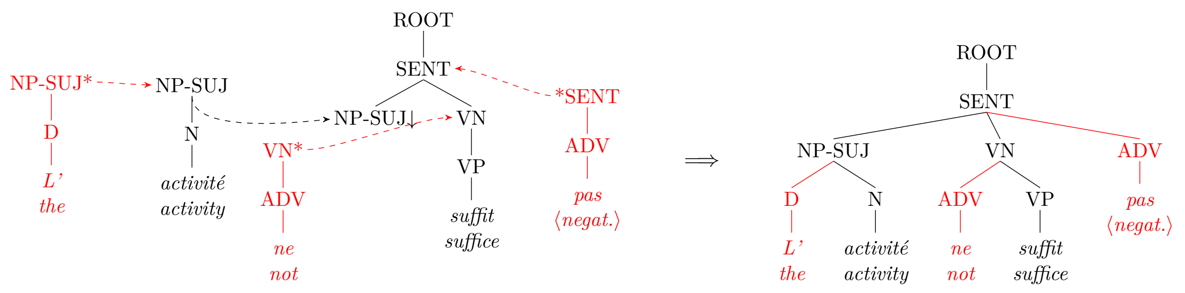


Figure 1: Supertagging with French LTAG for *L'activité ne suffit pas* ("The activity does not suffice")

An auxiliary tree has a *foot node* (marked with $*$) with the same label as the root node. When adjoining an *auxiliary tree* to some node n, the daughter nodes of n become daughters of the foot node. A sample TAG derivation is shown in Figure 2, in which the elementary trees for *Mary* and *pizza* are substituted to the subject and object slots of the *likes* tree and the auxiliary tree for *absolutely* is adjoined at the VP-node.

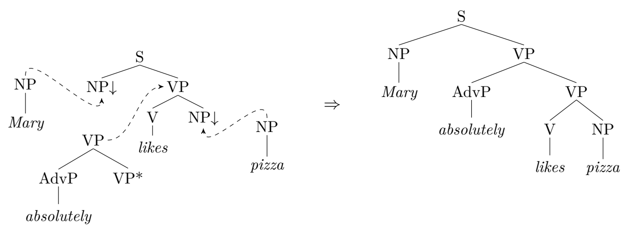


Figure 2: Elementary trees and a derived tree in LTAG

In a lexicalized version of TAG (LTAG) every tree is associated with a lexical item and represents the span over which this item can specify its syntactic or semantic constraints (for example, subject-verb number agreement or semantic roles) capturing also long-distance dependencies between the sentence tokens (Kipper et al., 2000).

2.2 RNN-based TAG supertagging

A supertagger is a partial parsing model which is used to assign a sequence of LTAG elementary trees to the sequence of words in a sentence (Sarkar, 2007). Supertagging can thus be seen as preparation for further syntactic parsing which improves the efficiency of the TAG parser through reducing syntactic lexical ambiguity and sentence complexity. Figure 1 provides an example of supertagging with an LTAG for French.

Several techniques were proposed for supertagging over the years, among which are HMM-based (Bäcker and Harbusch, 2002), n-gram-based (Chen et al., 2002), and Lightweight Dependency Analysis models (Srinivas, 2000). Recent advances show the applicability of recurrent neural networks (RNNs) for supertagging (Lewis et al., 2016; Vaswani et al., 2016; Kasai et al., 2017).

RNN-based supertagging with LTAGs can be seen as a standard sequence labeling task, albeit with a large set of labels (i.e., several thousand classes as supertags). Our deep learning pipeline is shown in Figure 3. A similar architecture showed good results for POS tagging across many languages (Plank et al., 2016).

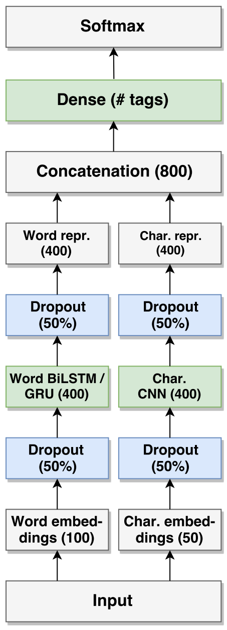


Figure 3: Supertagging architecture based on Samih (2017); dimensions shown in parentheses.

We use two kinds of embeddings: pre-trained word embeddings from the Sketch Engine collection of language models (Jakubíček et al., 2013; Bojanowski et al., 2016), and character embeddings based on the training set data. The pre-trained word embeddings encode distributional information from large corpora. The advantage of the character embeddings is that they can additionally encode subtoken information such as morphological features and help in dealing with unseen words, without doing any feature engineering on

Parameters	**French** (this work)	**German, reduced set** (Kaeshammer, 2012)	**German, full set** (Kaeshammer, 2012)	**English** (Kasai et al., 2017)
Supertags	5145	2516	3426	4727
Supertags occur. once	2693	1123	1562	2165
POS tags	13	53	53	36
Sentences	21550	28879	50000	44168
Avg. sentence length	31.34	17.51	17.71	appr. 20
Accuracy	**78.54**	85.91	**88.51**	**89.32**

Table 1: Supertagging experiments

morphological features.

The embeddings go through a recurrent layer to capture the influence of tokens in the preceding and subsequent context for each token. For the recurrent layer we use either bidirectional Long Short Term Memory (LSTM) or Gated Recurrent Units (GRU). We use a Convolutional Neural Network (CNN) layer for character embeddings, since it was proved to be one of the best options for extracting morphological information from word tokens (Ma and Hovy, 2016). The results for the word and character models are concatenated and fed through a softmax layer that gives a probability distribution for possible supertags. Dropout layers are added to counter overfitting. We replaced words without an entry in the word embeddings with a randomly instantiated vector of the same dimension (100). Table 2 provides an overview of the hyper-parameters we used for the supertagger architecture.

Layer	**Hyper-parameters**	**Value**
Characters CNN	numb. of filters	40
	state size	400
Bi-GRU	state size	400
	initial state	0.0
Words embedding	vector dim.	100
	window size	5
Char. embedding	dimension	50
	batch size	128
Dropout	dropout rate	0.5

Table 2: Hyper-parameters of the supertagger.

3 LTAG induction from the French Treebank

Inducing a grammar from a treebank entails identifying a set of productions that could have produced its parse trees. In the case of LTAG this means decomposing the trees into a sequence of elementary trees, one for each word in the sentence.

In order to extract a TAG from the French Treebank (FTB; Abeillé et al., 2003), we applied the heuristic procedure described by Xia (1999). The main idea of this approach is to consider the trees in the treebank as derived trees from an LTAG. Elementary trees are extracted in top-down fashion using percolation tables to identify grammatically obligatory elements (i.e., complements), grammatically optional elements (i.e., modifiers), as well as a head child for each constituent. All sub-trees corresponding to modifiers and complements are extracted in a further step forming auxiliary trees and initial trees, respectively, while the head child and its lexical anchor are kept in the tree. When extracted in this way, elementary trees contain the corresponding lexical anchor and the branches represent a particular syntactic context of a construction with slots for its complements.

3.1 LTAG induction: pre-processing steps

Before induction of different LTAGs for French, we carried out pre-processing steps described in Candito et al. (2010) and Crabbé and Candito (2008) including extension of the original POS tag set in FTB from 13 to 26 POS tags and undoing multi-word expressions (MWEs) with regular syntactic patterns (e.g. *(MWN (A ancien) (N élève))* $\rightarrow$ *(NP (AP (A ancien)) (N élève))*). About 14 % of the word tokens (79,466 out of the total of 557,095 tokens) in FTB belong to flat MWEs. After rewriting compounds with regular syntactic patterns, the number of MWEs is reduced to approximately 5 %.

We also restructured some trees in order to bring the complements on a higher level in the tree. In particular, we shifted the initial prepositional phrase of the VPinf constituents to a higher level and raised the subordinating conjunction (*C-S*) of the final clause constituents (*Ssub*) (see Figure 4).

After the preprocessing we extracted the following LTAGs from FTB for our supertagging experiments: including 13 or 26 POS tags, with and without compounds, including and excluding

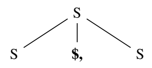
Gold supertag	Predicted supertag	Example
(PP (APPR < >)(NP↓))	(PP* (APPR < >)(NP↓))	**zu** einem Eigenheim zu verhelfen
(NP (DP↓)(NN < >))	(NP (NN < >))	das heutige und künftige **Kreditvolumen**
(S (S*)($, < >)(S↓))	($,* < >)	
(S (NP↓)(VVFIN < >)(NP↓)(PTKVZ↓))	(S (NP↓)(VVFIN < >)(NP↓))	der Umsatzminus **geht** auf 125 Millionen [...] zurück

Table 3: Most common error classes for German TAG supertagging with TiGer treebank

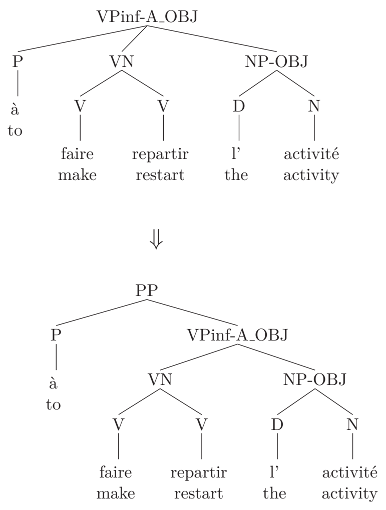


Figure 4: FTB preprocessing: complement raising

punctuation marks. Table 1 provides some statistics on the extracted LTAG which led to the most accurate supertagging results (13 POS tags, without compounds, including punctuation marks).

3.2 Left- and right-sister-adjunction

Extraction of an LTAG from FTB is challenging due to the flat structure of the trees, which allows any combination of arguments and modifiers. In order to preserve the original flat structures in the FTB as far as possible and to facilitate the extraction of the elementary trees we decided against the traditional notion of adjunction in TAG which relies on nested structures and apply *sister-adjunction*; i.e., the root of a sister-adjoining tree can be attached as a daughter of any node of another tree with the same node label.

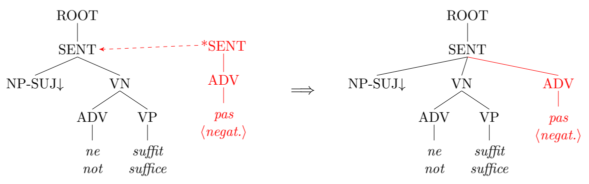


Figure 5: Left-sister-adjunction

Since a modifier can appear on the right or on the left side relative to the position of the constituent head, we distinguish between *right-* and *left-sister-adjoining trees* (marked with * on the left or the right side of the root label as shown in Figure 5).

A left-sister-adjoining tree γ can only be adjoined to a node η in the tree τ if the root label of γ is the same as the label of η and the anchor of the elementary tree τ comes in the sentence before the anchor of γ. The children of γ are inserted on the right side of the children in η and become the children of η. A *right-sister-adjunction* is defined in a similar way.

The resulting LTAGs with sister-adjunction are basically LTIGs (*Lexicalized Tree Insertion Grammar*; Schabes and Waters, 1995) in the way that the auxiliary trees do not allow wrapping adjunction or adjunction on the root node but permit multiple simultaneous adjunction on a single node of initial trees. However, since LTIG is a special variant of LTAG, we refer to the extracted grammar as LTAG in the remainder of the paper.

4 Experiments and error analysis

4.1 Experimental setups for German and French

In order to compare the performance of our supertagger with previous work of Kasai et al. (2017) and LTAG-based supertaggers for German (Bäcker and Harbusch, 2002; Westburg, 2016), we experimented with the supertags extracted by Kaeshammer (2012) from the German TiGer treebank (Brants et al., 2004). The set of supertags for German has the following train, test, and dev. split: 39,925, 5035, and 5040 sentences. We ran a supertagging experiment with this number of sentences, since it is compatible with the experimental setup described in Kasai et al. (2017). Since the number of sentences in FTB is smaller than in TiGer, we created a sample of the train set of the TiGer treebank with a comparable number of sentences in the train set (18,809). For the supertagging experiments with the French LTAG, we divided FTB in the standard train, development and test sets (19,080, 1235, and 1235 sentences), making our test and dev. sets comparable to the dev. and test set reported in Candito et al. (2009).

Tables 3 and 5 show the most frequent erroneous supertags for German and French. The symbol < > in the supertags signifies the spot for the lexical anchor, while * marks the foot node of auxiliary trees and ↓ represents a substitution site.

4.2 German TAG supertagging with TiGer

Generally, results for supertagging with German LTAGs appear to be slightly lower than for English. Westburg (2016) reports an accuracy of 82.92 % for German TAG with a supertagger based on perceptron training algorithm, while Bäcker and Harbusch (2002) reached 78.3 % with a HMM-based TAG supertagger.

Supertagging for German is more challenging than for English due to a higher number of word order variations and the resulting sparseness of the data (Bäcker and Harbusch, 2002). However, our experiments show that the proposed neural supertagging architecture reaches the best performance among the previously described supertaggers for German (88.51 %) and gets comparable results to the supertagging model for English described in Kasai et al. (2017) (see Tables 1 and 4).

System	Accuracy
Bäcker and Harbusch (2002) (HMM-based)	78.3
Westburg (2016)	82.92
This work, full training set (Bi-LSTM)	87.67
This work, full training set (GRU)	**88.51**
This work, reduced training set (Bi-LSTM)	85.26
This work, reduced training set (GRU)	**85.91**

Table 4: Supertagging experiments with German TiGer treebank.

The biggest class of errors for German supertagging contains wrong predictions concerning the type of the elementary tree (e.g. the supertagger predicts an auxiliary tree instead of an initial tree or vice versa). The main reason for this kind of error is the particularity of German which allows dependent elements in a sentence being divided by a big number of other tokens. For example, a determiner and the determined word or the separable verb prefix and the verb stem can be separated by a dozen other tokens, as in the sentence *Der Umsatzminus* ***geht*** *auf 125 Millionen [..]* ***zurück*** (Engl. "The sales drop goes down to 125 millions"), the verb *geht* and its prefix *zurück* are separated by 11 tokens (see Table 3).

Since the window size of tokens presented to the supertagger is limited, the connection between the tokens can be overlooked by the supertagger. However, increasing the window size leads to greater noise in the data. We experimented with window sizes of 5, 9, and 13 for German and got the best results with a window size of 5 (two words before and after the token).

Another source of mistakes for German is the intersentential punctuation in large complex sentences containing several subordinated clauses. This error can also be explained by the window size of tokens presented to the supertagger—the supertagger does not capture the complex structure of the sentence and classifies the punctuation mark as a one-child auxiliary tree (see Table 3).

Another big class of errors comes from PPs which can be either optional (modifiers) or obligatory elements. For example, the supertagger did not recognize that the verb *verhelfen* (Engl. "to help") requires a prepositional phrase as an argument (e.g. *zu einem Eigenheim zu verhelfen*; Engl. "to help someone to buy a property") and erroneously classified this complement as a modifier PP.

4.3 French TAG supertagging with FTB

Supertagging with French LTAGs appears to be more challenging compared to German or English. There are several general reasons for the performance drop of the supertagger, one of which is a higher average sentence length in FTB (31.34 tokens per sentence, compared to 17.51 in TiGer). Sentences in FTB more frequently have a complex syntactic structure including explicative elements separated with brackets or commas.

The large number of supertags lead to higher data sparsity and make the sequence labeling problem more difficult for the supertagger. One explanation for the larger number of supertags, besides the longer and more complex sentence structures in FTB, is the large number of flat multi-word expressions in FTB. Our experiments show that rewriting MWEs with regular compounds improves the supertagging performance.

A large number of supertagging errors for French occur due to different sites of attachment of the intersentential punctuation marks in FTB. The punctuation marks in FTB are attached to the corresponding constituents and not consistently to the

Gold supertag	Predicted supertag	Example
(NP* (PP (P <>) (NP↓)))	(PP (P <>) (NP↓))	32 % **par** an
(NP* (PONCT < >))	(SENT* (PONCT <>))	**-LRB-** 66,7 % **-RRB-**
(NP* (N < >))	(N < >)	Mme Dominique **Alduy**
(ROOT (SENT (NP↓) (VN (V < >))))	(VN (V < >))	le droit est officiellement **transgressé**

Table 5: Most common error classes for LTAG supertagging with French Treebank

System	Accuracy
This work (GRU), 13 POS, undone comp.	**78.54**
This work (GRU), 13 POS, no punct. marks	74.44
This work (GRU), 13 POS, with compounds	76.78
This work (GRU), 26 POS, with compounds	74.84
This work (Bi-LSTM), 13 POS, undone comp.	77.67

Table 6: Supertagging experiments with French Treebank (FTB).

root node of the whole sentence. However, since punctuation marks also help to identify possible constituents, omitting them does not improve supertagging.

Similar to supertagging with German LTAGs, PP attachments are also a major source of errors with French LTAGs. In addition to difficulties with classifying PPs as modifiers or complements (as with German data), the supertagger for French more frequently encounters problems with identifying the correct site for attaching the PPs to a node in the syntactic tree. The reason for these errors could be that FTB—in comparison to TiGer—does not offer additional function marks to distinguish PPs as modifiers from prepositional complements of the support verbs.

4.4 N-best supertagging experiments

The softmax layer of the supertagging model we described in section 2.2 provides a distribution of probabilities of the supertags when classifying words in a sentence, and we used this distribution to enable our supertagger to predict n-best supertags.

n-best	Accuracy German (full set)	Accuracy German (red. set)	Accuracy French
1-best	88.51	85.91	78.54
2-best	94.37	93.04	87.34
3-best	96.08	95.00	90.85
5-best	97.45	96.66	94.38
7-best	98.03	97.40	96.00
10-best	98.52	97.97	97.08

Table 7: N-best supertagging experiments.

We experimented with different numbers of n-best supertags for every word, counting the number of accurately predicted supertags each time when at least one of the n-best supertags was predicted correctly. The experiments show a quick growth in accuracy prediction up to 5-best supertags, while for ranks $n > 5$ the improvement of accuracy is not as big (see Table 7).

5 Conclusion and Future Work

We proposed a neural architecture for supertagging with TAG for German and French and carried out experiments to measure the performance of the supertagging model for these languages. We induced several different LTAGs from FTB in order to compare the supertagging performance. The results with German LTAG show that the neural supertagging model achieves comparable results to the state-of-the art TAG supertagging model described in Kasai et al. (2017) for English, even though German is more difficult for supertagging due to the free word order and the data sparseness. Supertagging for French appears to be more difficult due to the larger average length of sentences and a big number of multiword expressions.

In future work we plan to increase performance of the supertagger for French by dividing the supertagging algorithm in two steps: factorization of the extracted supertags in tree families and deciding afterwards on the correct supertag within the predicted tree family. We plan to use the improved supertagger for graph-based parsing. In particular, we aim at adapting the A*-based PARTAGE parser for LTAGs developed by Waszczuk (2017) for parsing with extracted supertags. We also intend to add deep syntactic features and information on semantic roles to the supertags in order to test whether the proposed supertagging architecture can be used for semantic role labeling.

Acknowledgements

This work was carried out as a part of the research project TREEGRASP (http://treegrasp.phil.hhu.de) funded by a Consolidator Grant of the European Research Council (ERC). We thank three anonymous reviewers for their careful reading, valuable suggestions and constructive comments.

References

Anne Abeillé, Lionel Clément, and François Toussenel. 2003. Building a treebank for french. In *Treebanks*, pages 165–187. Springer.

Jens Bäcker and Karin Harbusch. 2002. Hidden markov model-based supertagging in a user-initiative dialogue system. In *Proceedings of TAG+ 6*, pages 269–278.

Srinivas Bangalore and Aravind K Joshi. 1999. Supertagging: An approach to almost parsing. *Computational linguistics*, 25(2):237–265.

Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2016. Enriching word vectors with subword information. *arXiv preprint arXiv:1607.04606*.

Sabine Brants, Stefanie Dipper, Peter Eisenberg, Silvia Hansen-Schirra, Esther König, Wolfgang Lezius, Christian Rohrer, George Smith, and Hans Uszkoreit. 2004. Tiger: Linguistic interpretation of a german corpus. *Research on language and computation*, 2(4):597–620.

Marie Candito, Benoît Crabbé, and Pascal Denis. 2010. Statistical french dependency parsing: treebank conversion and first results. In *Seventh International Conference on Language Resources and Evaluation-LREC 2010*, pages 1840–1847. European Language Resources Association (ELRA).

Marie Candito, Benoît Crabbé, and Djamé Seddah. 2009. On statistical parsing of french with supervised and semi-supervised strategies. In *Proceedings of the EACL 2009 Workshop on Computational Linguistic Aspects of Grammatical Inference*, CLAGI '09, pages 49–57, Stroudsburg, PA, USA. Association for Computational Linguistics.

John Chen. 2010. Semantic labeling and parsing via tree-adjoining grammars. *Supertagging: Using Complex Lexical Descriptions in Natural Language Processing*, pages 431–448.

John Chen, Srinivas Bangalore, Michael Collins, and Owen Rambow. 2002. Reranking an n-gram supertagger. In *Proceedings of the Sixth International Workshop on Tree Adjoining Grammar and Related Frameworks (TAG+ 6)*, pages 259–268.

Benoit Crabbé and Marie Candito. 2008. Expériences d'analyse syntaxique statistique du français. In *15ème conférence sur le Traitement Automatique des Langues Naturelles-TALN'08*, pages pp–44.

Miloš Jakubíček, Adam Kilgarriff, Vojtěch Kovář, Pavel Rychlỳ, and Vít Suchomel. 2013. The tenten corpus family. In *7th International Corpus Linguistics Conference CL*, pages 125–127.

Aravind K Joshi and Yves Schabes. 1997. Tree-adjoining grammars. In *Handbook of formal languages*, pages 69–123. Springer.

Miriam Kaeshammer. 2012. *A German treebank and lexicon for tree-adjoining grammars*. Ph.D. thesis, MasterâĂŹs thesis, Universitat des Saarlandes, Saarlandes, Germany.

Jungo Kasai, Bob Frank, Tom McCoy, Owen Rambow, and Alexis Nasr. 2017. Tag parsing with neural networks and vector representations of supertags. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1712–1722.

Karin Kipper, Hoa Trang Dang, William Schuler, and Martha Palmer. 2000. Building a class-based verb lexicon using tags. In *Proceedings of the Fifth International Workshop on Tree Adjoining Grammar and Related Frameworks (TAG+ 5)*, pages 147–154.

Mike Lewis, Kenton Lee, and Luke Zettlemoyer. 2016. LSTM CCG parsing. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 221–231.

Xuezhe Ma and Eduard Hovy. 2016. End-to-end sequence labeling via bi-directional lstm-cnns-crf. *arXiv preprint arXiv:1603.01354*.

Barbara Plank, Anders Søgaard, and Yoav Goldberg. 2016. Multilingual part-of-speech tagging with bidirectional long short-term memory models and auxiliary loss. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 412–418, Berlin, Germany. Association for Computational Linguistics.

Younes Samih. 2017. *Dialectal Arabic processing Using Deep Learning*. Ph.D. thesis, Düsseldorf, Germany.

Anoop Sarkar. 2007. Combining supertagging and lexicalized tree-adjoining grammar parsing. *Complexity of Lexical Descriptions and its Relevance to Natural Language Processing: A Supertagging Approach*, page 113.

Yves Schabes and Richard C Waters. 1995. Tree insertion grammar: a cubic-time, parsable formalism that lexicalizes context-free grammar without changing the trees produced. *Computational Linguistics*, 21(4).

Bangalore Srinivas. 2000. A lightweight dependency analyzer for partial parsing. *Natural Language Engineering*, 6(2):113–138.

Ashish Vaswani, Yonatan Bisk, Kenji Sagae, and Ryan Musa. 2016. Supertagging with LSTMs. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 232–237.

Jakub Waszczuk. 2017. *Leveraging MWEs in practical TAG parsing: towards the best of the two world*. Ph.D. thesis.

Anika Westburg. 2016. Supertagging for german - an implementation based on tree adjoining grammars. Master's thesis, Heinrich Heine University of Düsseldorf.

Fei Xia. 1999. Extracting tree adjoining grammars from bracketed corpora. In *Proceedings of the 5th Natural Language Processing Pacific Rim Symposium (NLPRS-99)*, pages 398–403.