# What do we need to know about an unknown word when parsing German

**Bich-Ngoc Do**[⋆] and **Ines Rehbein**[♣] and **Anette Frank**[⋆]
Leibniz ScienceCampus
Universität Heidelberg[⋆] / Institut für Deutsche Sprache Mannheim[♣]
Germany
{do|rehbein|frank}@cl.uni-heidelberg.de

**Abstract**

We propose a new type of subword embedding designed to provide more information about unknown compounds, a major source for OOV words in German. We present an extrinsic evaluation where we use the *compound* embeddings as input to a neural dependency parser and compare the results to the ones obtained with other types of embeddings. Our evaluation shows that adding compound embeddings yields a significant improvement of 2% LAS over using word embeddings when no POS information is available. When adding POS embeddings to the input, however, the effect levels out. This suggests that it is not the missing information about the semantics of the unknown words that causes problems for parsing German, but the lack of morphological information for unknown words. To augment our evaluation, we also test the new embeddings in a language modelling task that requires both syntactic and semantic information.

## 1 Introduction

Parsing morphologically rich languages (MRLs) is a challenging task. One of the main problems is the high proportion of unknown words in the data, due to the high number of different inflected forms. In some languages, this problem is made even worse by *compounding*, a highly productive word formation process. Thus, handling unknown words is crucial for parsing MRLs and especially for German where compounding is a frequent phenomenon.

While word embeddings are a promising way to learn a general representation that captures syntactic and semantic properties of a word, they have not fully solved the sparse data problem. Recent studies are exploring representations at the subword level that can provide information even for rare and unseen words. Well-known examples are character and character-ngram-based embeddings (Sperr et al., 2013; dos Santos and Zadrozny, 2014; Ling et al., 2015; Vania and Lopez, 2017), morphological embeddings (Luong et al., 2013; Botha and Blunsom, 2014; Cotterell and Schütze, 2015; Cao and Rei, 2016), or byte embeddings (Plank et al., 2016; Gillick et al., 2016).

Ballesteros et al. (2015) were the first to integrate character-based embeddings into a syntactic parser and compared the effect for different languages with different levels of morphological richness. They showed that replacing word embeddings with character-based embeddings can be useful, especially for parsing agglutinative languages. Since then, character-based embeddings have become an ingredient in many parsing systems.

Other work has addressed the compounding problem on the level of word embeddings. Dima et al. have tried to model compound compositionality for English (Dima and Hinrichs, 2015) and German (Dima, 2015). However, experiments were on the semantic level, and the compounds were restricted to two components only. To the best of our knowledge, nobody has tried compound embeddings to tackle the unknown word problem in statistical parsing.

## 2 The problem with compounds

*Compounds* are words that include more than one stem. In some languages (e.g. English), the individual components are separated by spaces, while in other languages, such as German, they are merged into a new word form. Compounding is highly productive and thus, in languages like German, a major source of new, unseen words. Take,

*Proceedings of the First Workshop on Subword and Character Level Models in NLP*, pages 117–123,
Copenhagen, Denmark, September 7, 2017. ©2017 Association for Computational Linguistics.

| Threshold | en | de |
|---|---|---|
| 1 | 13.17 | 37.18 |
| 2 | 19.48 | 48.78 |
| 3 | 24.39 | 55.59 |
| 4 | 28.36 | 60.51 |
| 5 | 31.90 | 64.12 |

Table 1: The percentage of unknown words in the *test* data set with respect to different levels of cut-off threshold in the *training* data. Threshold of 1 means no words in the training data are discarded.

for example, the German compound *Verbraucherschutzgesetz* (consumer protection law). While all three parts are reasonably frequent and thus have a good chance of being included in a sufficiently large data set, the *merged* compound itself, most probably, is not.

This poses a problem for most statistical parsers. In our work, we focus on recent neural dependency parsers which, instead of using hand-crafted feature templates, directly learn the features from the training data (Chen and Manning, 2014; Zhang et al., 2017). These parsers usually introduce an UNKNOWN token for out-of-vocabulary words. A well-known technique for computing the embeddings of the UNKNOWN token is to discard infrequent words below a certain *threshold* and also treat them as *unknown*.

To illustrate the differential effect of this practice for languages that write compounds with word-internal spaces versus languages that use run-together compounds, let us take a look at the English Penn Treebank (PTB) (Marcus et al., 1993) and the German data set from the SPMRL 2014 shared task (Seddah et al., 2014), and compare the ratio of sparse or unknown words in the test sets for both treebanks with regard to different frequency thresholds. Table 1 shows that the ratio of words to be declared *unknown* is more than twice as high in the German data, due to a high amount of unknown common nouns. At a cutoff threshold of 5, the most frequent POS for unknown words in the German data are common nouns (47.4%) and proper nouns (17.3%). In the English data, however, proper names are the most frequent source for UNKNOWNs (35.4%) and common nouns only amount to 24.1%. One of the main reasons for this difference between the two Germanic languages is the high productivity of German compounds. We thus hypothesize that the high ratio of compounds will have a major impact on parsing German, which we address with our new *compound* embeddings.

## 3 Character vs Compound Embeddings

In a neural parsing system, each word is represented by a vector stored in a lookup table. One way to reduce the negative effect of unknown words in the vocabulary and also, if only indirectly, provide a treatment for compound words, is to replace the word lookup table by **character-based embeddings** (Ling et al., 2015). In this approach, each word is treated as a sequence of characters and the representation for each word is constructed from the representations for its characters, using a bi-directional long short-term memory network (LSTM) (Hochreiter and Schmidhuber, 1997). Given word $w$ as a sequence of characters $(c_1, c_2, ..., c_m)$ and $e_c$ as the vector representation of character $c$, we can compute the representation $e_w^{\text{char}}$ of word $w$ as follows:

$$s_t^F = \text{LSTM}_F(e_{c_t}, s_{t-1}^F) \quad (1)$$

$$s_t^B = \text{LSTM}_B(e_{c_t}, s_{t+1}^B) \quad (2)$$

$$e_w^{\text{char}} = D^F s_m^F + D^B s_0^B + b \quad (3)$$

where $s_t^F$ and $s_t^B$ are the hidden states of the forward and backward LSTMs at time $t$; $D^F$, $D^B$ and $b$ are the weight and bias vectors.

We now outline our compositional model for **compound embeddings**. We assume that most compounds have a transparent meaning that can be inferred from the meaning of its components and hypothesize that providing the parser with subword embeddings that combine the representations of the individual components will help the model to handle unseen compounds. To that end, we first split each compound into lexemes and then combine the sequence of lexemes as we did for the character-based embeddings, using a bi-directional LSTM.

For compound splitting, we use the IMS splitter (Weller and Heid, 2012) which adopts a frequency-based approach with additional linguistic features. The input information for the splitter (frequencies, POS and lemmas) was extracted from SdeWac (Faaß and Eckart, 2013), a cleaned-up version of the *deWac* corpus (Baroni et al., 2009) with automatic POS tags and lemmas.

## 4 Experiments

### 4.1 Parsing Model

Our parsing model is an extension of the *head-selection* parser of Zhang et al. (2017) (figure 1). Given the sentence $S = (w_0, w_1, ..., w_N)$ and $x_i$ as the *input representation* of word $w_i$, the model

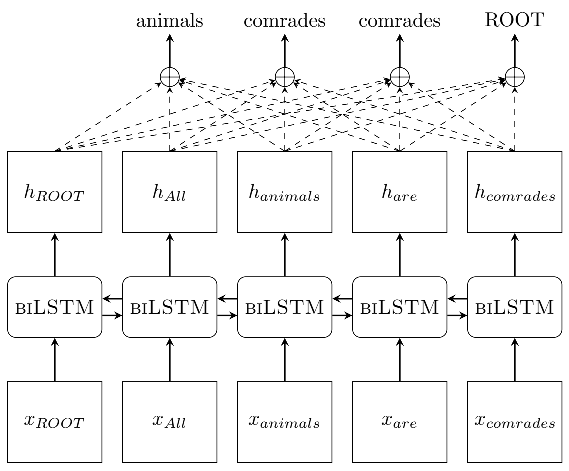

Figure 1: The parsing as head selection model

uses a bi-directional LSTM to learn a *feature vector* for each word in $S$:

$$h_i^F = \text{LSTM}_F(x_i, h_{i-1}^F) \quad (4)$$

$$h_i^B = \text{LSTM}_B(x_i, h_{i+1}^B) \quad (5)$$

$$h_i = [h_i^F; h_i^B] \quad (6)$$

The feature vector $h_i$ of word $w_i$ is the concatenation of the hidden states from the forward and backward passes of the bi-direction LSTM. An artificial root node $w_0$ token is appended at the beginning of each sentence.

Unlabeled parsing is modeled as choosing the most probable head for each word in a sentence. In sentence $S = (w_0, w_1, ..., w_N)$, the probability of word $w_j$ being the head of $w_i$ is calculated as:

$$P_{\text{head}}(w_j \mid w_i, S) = \frac{\exp(g(a_j, a_i))}{\sum_{k=0}^{N} \exp(g(a_k, a_i))} \quad (7)$$

where $g$ is a neural network that predicts a score for the feature vectors $h_i$ and $h_j$ as follows:

$$g(a_j, a_i) = v_a^\top \cdot \tanh(U_a \cdot a_j + W_a \cdot a_i) \quad (8)$$

Finally, an additional neural network is used to assign the grammatical function label to each edge in the unlabeled tree. The input to that network is the concatenation of the input representations and the learned feature vectors of head $j$ and dependent $i$:

$$[x_i; x_j; h_i; h_j] \quad (9)$$

Note that in our implementation we use a single hidden-layer rectifier network instead of the two-layer rectifier network in Zhang et al. (2017), since we achieve better results with only one hidden layer.

| Module | Hyperparameter | Value |
|---|---|---|
| Word emb. | size | 300 |
| POS emb. | size | 40 |
| Character-based emb. | character embedding size | 50 |
| | hidden size | 100 |
| Compound emb. | lexeme embedding size | 50 |
| | hidden size | 100 |
| BiLSTM | hidden size | 300 |
| Regularization | L2 | 1e-3 |
| | input dropout rate | 0.05 |
| | dropout rate | 0.5 |
| | max-norm | 5.0 |
| Optimization | optimizer | Adam |
| | learning rate | 0.001 |
| | 1st momentum | 0.9 |
| | 2st momentum | 0.999 |
| | no. epochs | 15 |
| Others | word cutoff threshold | 5 |

Table 2: Hyperparameters used in all experiments

## 4.2 Input representations

To assess the effect of different compound handling techniques on parsing performance, we systematically vary the input information for the parser, as described below:

**Word Embeddings (`+word`)** Each word $w$ in the lexicon is represented as a vector $e_w$ in the lookup table. We do not use any pre-trained embeddings; all embeddings are initialized randomly.

**POS Embeddings (`+pos`)** If word $w$ has POS tag $p$, we add an embedding $e_p$ for tag $p$ to the input information.

**Character-Based Embeddings (`+char`)** In addition to the word embeddings $e_w$ from the lookup table, we also use the character-based embeddings $e_w^{\text{char}}$ of word $w$ (equation 3).

**Compound Embeddings (`+comp`)** The compound embedding $e_w^{\text{comp}}$ of word $w$ is calculated based on the lexeme embeddings (see section 3).

When combining different types of information in the input, we use the *concatenation* of each embedding type.

## 4.3 Training

We train our own implementation of the parser, following Zhang et al. (2017). For optimization, we used Adam (Kingma and Ba, 2015) with default parameters. All models were trained in 15 epochs and the training process was regularized using common techniques like L2 regularization, max-norm and dropout (Srivastava et al., 2014). We chose all hyperparameters for our

| | | Input | UAS | LAS |
|---|---|---|---|---|
| **+pos** | **b1** | `+word,pos` | 90.50 | 88.06 |
| | **b2** | head:`+word,pos` | 90.46 | 88.13 |
| | | `+comp,pos` | 90.23 | 87.93 |
| | | `+comp,word,pos` | 90.39 | 88.10 |
| | | `+char,pos` | 90.53 | 88.49 |
| | | `+word,char,pos` | **90.69** | **88.56** |
| **-pos** | **b1** | `+word` | 86.27 | 83.08 |
| | **b2** | head:`+word` | 87.00 | 83.97 |
| | | `+word,comp` | 88.29 | 85.42 |
| | | `+word,char` | **90.42** | **88.20** |

Table 3: Results for different input combinations

experiments manually, following suggestions by Zhang et al. (2017) (see table 2).

We report parsing performance (UAS and LAS) *with punctuation* on the German data set from the SPMRL 2014 shared task. The training set contains 40,472 sentences and the development and test sets both include 5,000 sentences.

The compound splitting (section 3) affected about one third of the lexemes in the lexicon, all of them nouns. Of all the unknown words in the test set (64.12% at a cutoff threshold of 5, see table 1), 24.92% now consist of known lexemes, 73.79% have only one unknown lexeme, and only 1.29% have more than one unknown component.

We compare our results against two baselines, (b1) using the original words for parsing and (b2) a greedy baseline *head*, where we discard all compound components except the rightmost one, since in most cases, the rightmost lexeme is the head of the compound. Baseline (b2) reduces the number of unknown words in the data by 10%.

## 4.4 Results

Table 3 (+pos) shows results for different combinations of input information. The `+word,pos` setting (baseline b1) is the one implemented in the original parser. The results show that our special treatment of compounds does not have the desired effect. In both settings, using only the head words (b2) and using compound embeddings, we see only minor changes in parsing accuracy and when replacing word with compound embeddings, results actually decrease. This strongly suggests that the parsing model is often able to make the right decision without actually knowing the word.

Adding the character embeddings improves the LAS by 0.5%, but does not have a significant effect on the UAS. Since German is a richly inflected, semi-free word order language, this suggests that the character-based embeddings have learned *morphological* information from the surface of the words that helps assign the correct grammatical function for each head-dependent pair. Table 4 confirms this by showing the improvements we get for the core arguments when adding character-based embeddings.

| Label | Freq. | `-char` | | `+char` | |
|---|---|---|---|---|---|
| | | P | R | P | R |
| SB | 6,638 | **90.7** | 91.2 | 90.6 | **92.2** |
| OA | 3,184 | 82.3 | 85.7 | **83.3** | **87.5** |
| DA | 568 | 73.2 | 55.3 | **78.9** | **63.9** |
| AG | 2,241 | 91.3 | 91.5 | **94.2** | **93.9** |
| OG | 21 | **100.0** | **4.8** | N/A | 0.0 |
| PD | 1,045 | 82.5 | 74.3 | **84.8** | **80.8** |

Table 4: Precision (P) and recall (R) for core grammatical functions with/without character-based embeddings. SB: subj, OA: accusative obj, DA: dative obj, AG: genitive attribute, OG: genitive obj, PD: predicate.

**The effect of POS tags** In the next set of experiments, we exclude the POS tag information to isolate the effect of the different techniques for handling compounds. Table 3 (-pos) shows that without POS information, we now see a significant effect. The greedy baseline that keeps only the head word for each compound increases UAS and LAS by 0.7% and 0.9% respectively, and our compound embeddings now improve both scores by more than 2%. Using character-based embeddings in combination with word embeddings, however, yields comparable results to the `+word,pos` system. We take that as evidence that the character-based embeddings implicitly learn morphological information that is complementary to the information included in the word embeddings. Our results are in line with previous results from the literature, claiming that character-based embeddings are able to capture morphological information (Ling et al., 2015; Cao and Rei, 2016; Kim et al., 2016).

Our results also corroborate findings by Köhn (2016) who evaluates different types of *word* embeddings in a syntax-based classification task, reporting that the embeddings yielded improvements *only* when no POS information was given.

## 4.5 Language Modeling

To validate our results in a different setting, we also test the compound embeddings in a language modeling task. Language models (LM) are an important ingredient in many NLP applications, e.g. in speech recognition and machine translation, and they require both syntactic and semantic information.

| Model | Word | Compound | Char |
|---|---|---|---|
| Perplexity | 36.954 | 35.987 | **32.273** |

Table 5: Perplexity for different language models on German texts from Wikipedia.

In our experiment, we use the framework[1] and setup described in Vania and Lopez (2017) to build a language model for German texts. The framework includes implementations for word and subword-based (morpheme, character or character $n$-gram) embeddings and uses either bidirectional LSTMs or addition as the combination function of subwords.

The German data sets are from the preprocessed Wikipedia data (Al-Rfou et al., 2013). Hyperlinks have been removed and the input texts have been lower-cased before learning the word- and compound-based embeddings. For the character-based embeddings, the upper-cased letters have been preserved. We split the data into training, development and test sets, with approximately 1.2M, 150K and 150K tokens, respectively. For training and evaluation we closely follow Vania and Lopez (2017).

We report results for three language models. The *word* model and the *character* model (using a bidirectional LSTM as composition function)[2] are already implemented in the framework. For the *compound* embeddings, we first split the compounds in the data sets as described in section 3 and then combine them, again using a bidirectional LSTM as composition function.

The results are shown in table 5. Using compound-based embeddings yields better perplexity in comparison to the vanilla word model, but the compound model is still far behind the character-based embeddings which obtain the lowest perplexity. The results for the language model thus confirm the trend observed in the parsing experiments.

## 5 Discussion

In both tasks, parsing and language modelling, the character-based embeddings clearly outperformed the compound-based embeddings. This suggests that the character-based embeddings are able to pick up structural information that is important for both tasks.

For parsing, the results for the compound-based embeddings were even below the ones for the word embeddings when including POS information in the input. This implies that the information needed for *parsing* unknown words is not so much information about the semantics of a word but, crucially, morphological information. This was confirmed by the improved results for using character-based embeddings instead of the compound-based ones, where we were able to make up for the decrease in LAS that resulted from removing POS information from the input.

Our results are important, as they show that unknown words are not per se a problem for parsing, as long as we are able to learn something about their morphological properties.

## 6 Conclusions and future work

In the paper, we introduced a new type of subword embedding, the *compound* embedding. The new embeddings are designed to provide more information about unknown compounds which constitute a major part of OOV words in German.

We evaluated the embeddings in dependency parsing and showed that although the compound-based embeddings outperformed word embeddings when *no* POS information was available, the character-based model showed a performance superior to the one for word and compound embeddings. For language modelling, where not only syntactic but also semantic information is important, the results follow the same trend.

This leaves us with two avenues for future work. To provide an improved handling of OOV words for parsing, we need to optimise subword embeddings to represent morphological information for unknown words. In addition, we would like to test the compound embeddings in a purely *semantic* task where we can explore their full potential.

## Acknowledgments

This research has been conducted within the Leibniz Science Campus "Empirical Linguistics and Computational Modeling", funded by the Leibniz Association under grant no. SAS-2015-IDS-LWC and by the Ministry of Science, Research, and Art (MWK) of the state of Baden-Württemberg.

[1] `https://github.com/claravania/subword-lstm-lm`

[2] These are the same character-based embeddings that we used in the parsing experiment (sections 4.3, 4.4).

## References

Rami Al-Rfou, Bryan Perozzi, and Steven Skiena. 2013. Polyglot: Distributed word representations for multilingual NLP. In *Proceedings of the Seventeenth Conference on Computational Natural Language Learning*. Association for Computational Linguistics, Sofia, Bulgaria, pages 183–192.

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*. Association for Computational Linguistics, Lisbon, Portugal, pages 349–359.

Marco Baroni, Silvia Bernardini, Adriano Ferraresi, and Eros Zanchetta. 2009. The WaCky wide web: A collection of very large linguistically processed web-crawled corpora. *Language Resources and Evaluation* 43(3):209–226.

Jan A. Botha and Phil Blunsom. 2014. Compositional morphology for word representations and language modelling. In *Proceedings of the 31st International Conference on Machine Learning*. Beijing, China, ICML 2014.

Kris Cao and Marek Rei. 2016. A joint model for word embedding and word morphology. In *Proceedings of the 1st Workshop on Representation Learning for NLP*. Berlin, Germany, RepL4NLP-2016.

Danqi Chen and Christopher D. 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)*. Association for Computational Linguistics, Doha, Qatar, pages 740–750.

Ryan Cotterell and Hinrich Schütze. 2015. Morphological word-embeddings. In *The 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Denver, Colorado, USA, May 31 - June 5, 2015*. NAACL HLT 2015, pages 1287–1292.

Corina Dima. 2015. Reverse-engineering language: A study on the semantic compositionality of German compounds. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, Lisbon, Portugal, pages 1637–1642.

Corina Dima and Erhard Hinrichs. 2015. Automatic noun compound interpretation using deep neural networks and word embeddings. In *Proceedings of the 11th International Conference on Computational Semantics*. Association for Computational Linguistics, London, UK, pages 173–183.

Cícero Nogueira dos Santos and Bianca Zadrozny. 2014. Learning character-level representations for part-of-speech tagging. In *Proceedings of the 31th International Conference on Machine Learning, ICML 2014, Beijing, China, 21-26 June 2014*. pages 1818–1826.

Gertrud Faaß and Kerstin Eckart. 2013. SdeWaC - A corpus of parsable sentences from the web. In *Language Processing and Knowledge in the Web: 25th International Conference, GSCL 2013, Darmstadt, Germany, September 25-27, 2013. Proceedings*. Springer, Berlin, Heidelberg, pages 61–68.

Dan Gillick, Cliff Brunk, Oriol Vinyals, and Amarnag Subramanya. 2016. Multilingual language processing from bytes. In *The 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. San Diego California, USA, NAACL HLT 2016, pages 1296–1306.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural Computing* 9(8):1735–1780.

Yoon Kim, Yacine Jernite, David Sontag, and Alexander M. Rush. 2016. Character-aware neural language models. In *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence*. AAAI Press, AAAI'16, pages 2741–2749.

Diederik Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *The International Conference on Learning Representations*. San Diego.

Arne Köhn. 2016. Evaluating embeddings using syntax-based classification tasks as a proxy for parser performance. In *Proceedings of the 1st Workshop on Evaluating Vector Space Representations for NLP*. Berlin, Germany, pages 67–71.

Wang Ling, Tiago Luís, Luís Marujo, Ramón Fernandez Astudillo, Silvio Amir, Chris Dyer, Alan W Black, and Isabel Trancoso. 2015. Finding function in form: Compositional character models for open vocabulary word representation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, Lisbon, Portugal, pages 1520–1530.

Thang Luong, Richard Socher, and Christopher D. Manning. 2013. Better word representations with recursive neural networks for morphology. In *Proceedings of the Seventeenth Conference on Computational Natural Language Learning*. Sofia, Bulgaria, CoNLL 2013, pages 104–113.

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.

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*. Berlin, Germany, ACL 2016.

Djamé Seddah, Sandra Kübler, and Reut Tsarfaty. 2014. Introducing the SPMRL 2014 shared task on parsing morphologically-rich languages. In *Proceedings of the First Joint Workshop on Statistical Parsing of Morphologically Rich Languages and Syntactic Analysis of Non-Canonical Languages*. Dublin City University, Dublin, Ireland, pages 103–109.

Henning Sperr, Jan Niehues, and Alex Waibel. 2013. Letter n-gram-based input encoding for continuous space language models. In *Proceedings of the Workshop on Continuous Vector Space Models and their Compositionality*. Sofia, Bulgaria, pages 30–39.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research* 15:1929–1958.

Clara Vania and Adam Lopez. 2017. From characters to words to in between: Do we capture morphology? In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics.

Marion Weller and Ulrich Heid. 2012. Analyzing and aligning German compound nouns. In *Proceedings of the Eight International Conference on Language Resources and Evaluation (LREC'12)*. European Language Resources Association (ELRA), Istanbul, Turkey.

Xingxing Zhang, Jianpeng Cheng, and Mirella Lapata. 2017. Dependency parsing as head selection. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*. Association for Computational Linguistics, Valencia, Spain, pages 665–676.