Parsing Free-Form Language Learner Data: Current State and Error Analysis

Christine Köhn and **Tobias Staron** and **Arne Köhn**
Department of Informatics
Universität Hamburg
{ckoehn,staron,koehn}@informatik.uni-hamburg.de

Abstract

Parsing learner data with high accuracy is important for all systems that want to analyze language learner input, such as computer-assisted language learning software. State-of-the-art parsers are typically trained on news text and not on language learner data since this kind of data is often not available in sufficient quantities. Our contribution is three-fold: We provide gold-standard syntactic annotations for sentences from language learners of German, evaluate the performance of state-of-the-art parser pipelines on this corpus and explore whether augmentation of a parser with weighted constraints to avoid common structural errors could lead to improvements.

1 Introduction

Syntax parsers are usually based on the assumption that the input is well-formed. Their statistical models are trained on large corpora, which are mostly annotated news text and therefore also comparatively well-formed. On the other hand, language learner (L2) data from people, who are learning a language that is not their native one, inherently contains malformed parts. This mismatch between well-formed training data and malformed run-time input may lead to a degradation in parser performance.

Training a parser directly on L2 data is not feasible for several reasons. First and foremost, there is too little annotated L2 data available for most languages. Also, training on L2 data would presuppose that all learners of the respective language make comparable mistakes, which is unlikely.

Extracting the syntactic structure of an L2 sentence is important for different applications, e. g. for assessing answers to reading comprehension questions or for deriving error diagnoses.

We are especially interested in free-form text where the sentence structure is not externally influenced, e. g. the text is not an answer to a question. For this type of input, it is especially hard to extract error diagnoses without syntactic analyses, since the contents of the L2 sentences are not known beforehand.

For the parser evaluation, we annotated 100 L2 German sentences (Falko-100dep) from a subcorpus of the Falko corpus (Reznicek et al., 2012) with dependency trees. The Falko corpus contains target hypotheses, i. e. manually corrected versions of the texts, which can be better automatically analyzed than the original learner sentences (Rehbein et al., 2012). Nevertheless, we evaluate the parsers on the original sentences since we want to assess their performance in a setting where manually created target hypotheses are not available.

2 Related Work

There is already a dependency annotated corpus of L2 German, which uses the same annotation standard as the Falko-100dep corpus: The CREG-109 corpus (Ott and Ziai, 2010), containing answers to reading comprehension questions. We chose to annotate essay texts from the Falko corpus because of their different characteristics. E. g., the sentences in the Falko-100dep corpus are much longer on average (18.9 tokens) than the responses to the respective questions in the CREG-109 corpus (8.3 tokens). Also, the language learner proficiency differs: The CREG-109 sentences were written by learners on the beginning and intermediate level, whereas the Falko-100dep sentences were written by upper intermediate to advanced learners.

Berzak et al. (2016) compiled a corpus of language learner sentences for L2 English, including part-of-speech (PoS) tags and Universal Dependency trees. They annotated the original, ungrammatical sentences as well as their corrected versions. Additionally, they provide a set of annotation

guidelines for syntactically annotating ungrammatical English.

Rehbein et al. (2012) examined the impact of PoS quality on parsing accuracy of language learner sentences. For this purpose, they annotated 100 L2 sentences from the Falko essay subcorpus with constituency structures. They compared the PoS accuracy when tagging the original sentence and a corrected form, the target hypothesis. The target hypotheses were formulated with the purpose of making them suitable for automatic processing. Tagging the target hypothesis of the L2 sentence and projecting the PoS tags back to the original sentence improved the PoS accuracy. Furthermore, they found that the manual correction of automatically assigned PoS tags for some of the taggers does not significantly improve parsing accuracy. Their approach of processing target hypotheses instead of the original sentence is appropriate for analyzing L2 corpora but we perform all experiments on the original learner sentence, simulating a setting where target hypotheses are not available.

Ragheb and Dickinson (2013) developed a multi-layered dependency annotation scheme for learner language and achieved good inter-annotator agreement for L2 English. One dependency layer represents morpho-syntactic information, while the other represents subcategorization information. The PoS tag annotation also consists of two layers: one for morpho-syntactic and one for distributional evidence.

Krivanek and Meurers (2011) compared a transition-based parser, MaltParser (Nivre, 2007), to a rule-based parser, WCDG (Foth and Menzel, 2006), for parsing L2 text by evaluating them on the CREG-109 corpus. They found that, while both parsers have a similar overall accuracy, MaltParser performs better at attaching optional relations, but WCDG is better at identifying the main functor-argument relations.

Hybrid parsing – i. e. incorporating more than one parsing approach – can be performed by using a statistical parser, such as MaltParser, as an additional input source for a rule-based parser, e.g. WCDG (Foth and Menzel, 2006). Khmylko et al. (2009) demonstrated that this approach is beneficial even if the statistical parser is superior to the rule-based one, i. e. the hybrid parser performs better than both its components. Köhn and Menzel (2013) showed that even though a combination of jwcdg, a Java re-implementation of WCDG (Beuck et al., 2013), and MaltParser is beneficial for newspaper text, combining both parsers does not help to improve parsing performance on the CREG-109 corpus.

It is also possible to build a hybrid parser the other way around: Seeker and Kuhn (2013) included morpho-syntactic constraints in statistical parsing to restrict the search space. In addition, morphological disambiguation is performed (which jwcdg also does). In contrast to the previously mentioned approaches, the constraints are not graded. Our approach, described in Section 5, is similar but uses graded constraints and does not perform morphological disambiguation.

Further work has been done on integrating grammars into data-driven parsers. Dhar et al. (2012) used MaltParser and the parses it generates are corrected by grammar rules which, in turn, are inferred from running MaltParser alone and analyzing its errors. This approach was tested for Bangla.

An alternative approach to develop hybrid parsers is to build an ensemble of statistical parsers. The parsers can be combined by n-best parsing and ranking, as performed by Björkelund et al. (2013). This approach yields the currently best results for the shared task on parsing morphologically rich languages.

Even a simple voting by several parsers can outperform the individual parser performances. For example, Sagae and Lavie (2006) combined several shift-reduce parsers similar to MaltParser as well as MST parser (McDonald and Pereira, 2006) by weighted voting for each edge. This approach yielded an increase of 1.7 percentage points in accuracy on the Penn Treebank.

3 The Falko-100dep Corpus

The FalkoEssayL2 corpus contains German essays written by language learners with varying degree of proficiency. Each learner had 90 minutes to write an essay on a given topic without help (neither machine nor human). In addition, each learner completed a C-test to assess their proficiency in German. The C-test scores can be translated into standard CEFR levels[1], which we did to cluster the texts into B2, C1, and C2, with C2 referring to the more advanced learners.

We randomly sampled 100 sentences from the

[1]The **C**ommon **E**uropean **F**ramework of **R**eference for Languages: Learning, Teaching, Assessment

	B2	C1	C2	all
Mean	13.6	20.8	22.6	18.9
Median	11.5	17.0	21.0	17.0

Table 1: Sentence length distribution in the Falko-100dep corpus by language proficiency

FalkoEssayL2 corpus v2.4[2], 33 to 34 for each level (B2, C1, C2) and used the manually corrected tokenization from the level ctok, where the text is otherwise completely untouched. The sentence segmentation was extracted from the level ZH0 (containing already corrected material) and mapped back to ctok, as ctok does not provide sentence segmentation. This way, the sentences we worked on are completely made up of uncorrected tokens, but manually tokenized and segmented.

The essays in the FalkoEssayL2 corpus cover four different topics, which are also represented in Falko-100dep: crime, academic studies, feminism, and wages. The sentence length correlates with language proficiency (see Table 1). The sentences produced by C2 learners are about twice as long as the sentences from B2 learners, suggesting that more experienced learners write more complex sentences.

The sentences contain all kinds of mistakes, e. g. spelling and grammatical mistakes. However, not all sentences contain mistakes. As evidenced by a high inter-annotator agreement (see next section), they do not prevent a reasonable annotation.

3.1 Annotation Process

Ragheb and Dickinson (2011) argue that learner language should be annotated with an annotation scheme specifically tailored to capture the phenomena of the learner's interlanguage and treat it as a system in its own right instead of comparing it to the target language or the learner's L2. We do not use an annotation scheme designed for learner language but use the annotation guidelines by Foth (2006)[3], which were designed for L1 German and are also used by the CREG-109 corpus and the Hamburg Dependency Treebank (HDT) (Foth et al., 2014). We did not use a scheme designed for L2, because this not available for German yet and consequently no parser can produce such an output since no training data is available. We are aware of the fact that an L1 scheme captures less information than the scheme proposed by Ragheb and Dickinson (2012) but we think that the annotations represent the syntactic structure of an L2 sentence well enough to serve as an input for further processing, e. g. error diagnosis.

Our annotation process was as follows: First, three annotators each annotated the first 12 sentences and met afterwards to discuss annotation decisions and to mutually decide on each controversial annotation to gain a higher agreement for the remaining annotation process. To speed up this process, the annotators did not start from scratch but corrected the parses of three different parsers (one parser per annotator: TurboParser, RBGParser, jwcdg - for an overview of these parsers, see Section 4). Two of the annotators continued with a partial overlap (sentences 13 to 22) and decided again on a gold standard annotation afterwards. They annotated sentences 23-100 separately, but again cross-checked the annotations of each other.

For the first 22 sentences, the two annotators achieve an inter-annotator agreement in terms of labeled attachment score (LAS) of 91.48%. The annotators agree on 93.73% of the dependencies and on 95.99% of the labels. This indicates an already high agreement between the two annotators which, in turn, shows that the sentences of our corpus can be annotated fairly well despite the mistakes they contain.

Comparing the annotations of the first 22 sentences with the resulting gold standard leads to a LAS of 94.99% for one annotator (96.99% of the dependencies and 97.49% of the labels remain) and a LAS of 95.49% for the other annotator (96.24% of the dependencies and 97.99% of the labels remain). This shows that only few changes were made when the annotators decided on the gold standard annotation.

Comparing the annotations for the remaining sentences 23 to 100 with the gold standard results in a LAS of 95.29% for the first annotator (agreement on 95.53% of the dependencies and on 97.52% of the labels) and 97.39% for the second one (agreement on 98.46% of the dependencies and on 97.87% of the labels) indicating improved annotations for sentences annotated later, based on comparing and discussing the annotations of the first 22 sentences.

[2]The corpus is available at `www.linguistik.hu-berlin.de/de/institut/professuren/korpuslinguistik/forschung/falko/zugang`

[3]The annotations guidelines are in German. Foth et al. (2014) give an overview of the dependency relations in English.

3.2 Annotation Decisions

The annotators knew what topics were dealt with in the FalkoEssayL2 corpus. The annotation was carried out with respect to an implicit target hypothesis (TH). Our TH is a grammatical correct sentence that makes sense as far as possible while at the same time tries to deviate from the original as little as possible. The TH is different to the ones defined in the Falko manual (Reznicek et al., 2012) since we want to attach as many words as possible to other words. The rules for our TH overlap largely with the rules for the minimal TH in the Falko corpus. The main difference is that we also change the verb if it seems more suitable.

4 Parser Evaluation

We evaluate three parsers: jwcdg (Beuck et al., 2013), TurboParser (Martins et al., 2013), and RBGParser (Zhang et al., 2014). While the first one is rule-based, the others are trained on a treebank. We used the first 100.000 sentences of part A of the HDT to train them.

TurboParser and RBGParser are quite similar in their feature set (at least in our experimental setup). However, their approach to decoding differs fundamentally. the decoding of jwcdg is similar to the one of RBGParser but it uses a hand-written weighted constraint grammar.

Since we want to assess the parses in a setup where gold standard PoS tags and target hypotheses are not available, we use predicted instead of gold standard PoS tags for all parsers. TurboTagger (distributed with TurboParser) assigns the PoS tags for both TurboParser and RBGParser. jwcdg requires multi-tagging and therefore uses TnT (Brants, 2000). TurboTagger was trained on the same data as the data-driven parsers.

4.1 TurboParser

TurboParser translates the parsing problem into a binary integer linear program (ILP) where each possible edge is assigned a variable. The ILP consists of two parts: The linear constraints make sure that each result is a tree while the objective function makes sure that the resulting tree is good. In contrast to the linear constraints, the objective function needs to be learned. To make learning and decoding feasible, the objective function is decomposed into local components (see Martins et al. (2013) for an overview). During decoding, a relaxation of the ILP is solved using dual decomposition.

Because each component of the objective function only scores a fixed set of edges (up to three), global constraints can not be learned. Due to the overlap between the different scoring components (edges of the dependency tree are part of several components), the best scoring tree needs to be locally consistent in each component. However, no component can enforce the existence of a specific construct, e.g. a subject for a verb.

4.2 RBGParser

RBGParser[4] is a data-driven dependency parser (Zhang et al., 2014). It exploits a variety of features: global as well as local features considering up to three connected edges in the dependency structure. Different models based on different subsets of features can be used. In this work, the standard model is used, which uses only local features comparable to the ones of TurboParser.

When using the standard model, RBG applies hill-climbing. It starts with a random parse and reassigns edges until the best parse stops changing. RBG repeats this procedure, each time starting with a newly sampled random parse, until the result converges in order to find a parse as optimal as possible. Because initial random parses for a given sentence are sampled independently from each other, using only first-order features, the scoring of an analysis is largely decoupled from the creation of new parses. Therefore, it is possible to use an arbitrarily complex scoring function. In addition to the TurboParser features, RBG employs a low-rank tensor component which scores single edges (Lei et al., 2014).

4.3 jwcdg

jwcdg (Beuck et al., 2013) implements the weighted constraint dependency grammar formalism (Schröder, 2002). It uses a grammar consisting of weighted constraints, which are used to score analyses, and taboo search (Foth et al., 2000) to find the optimal analysis for a sentence. This approach is comparable to the hill climbing performed by RBG, although the former is more complex.

Additionally, jwcdg is able to evaluate the constraints of its grammar on an already parsed sentence in order to determine constraint violations. Besides generating a score based on the constraint evaluation, the violated constraints can be inspected to analyze the parse.

[4] RBG in the remaining paper

		UAS			
	LAS	all	B2	C1	C2
RBG	80.32	86.70	86.72	84.40	88.79
Turbo	81.83	86.76	85.96	85.23	88.63
jwcdg	77.40	82.02	84.96	79.87	82.18
RBG_h	79.95	86.03	85.96	83.72	88.17

Table 2: Attachment scores for the Falko-100dep corpus (labeled and unlabeled) for RBG, TurboParser, jwcdg and the hybrid parser RBG_h (RBG augmented with constraints); UAS also by learner proficiency.

The grammar was co-developed during the annotation process of the HDT. In contrast to the scoring functions learned by the other parsers, each constraint (and its purpose) can be understood by humans as it is directly linguistically motivated. Since the constraints were created manually, they rely less on the word forms. E. g., differences in distributional attachment preferences for nouns are mostly not modeled. In this paper, we use jwcdg without external predictors, except for a PoS tagger, to assess the quality of the underlying grammar.

In contrast to the previously mentioned parsers, jwcdg co-optimizes dependency structures, dependency labels, and PoS tags and performs lexical disambiguation. jwcdg uses TnT in a multi-tagging mode to obtain weighted suggestions for PoS tags. Due to the lexical disambiguation, the grammar makes extensive use of features such as valence, number, and other morpho-syntactic information.

4.4 Evaluation on Falko-100dep and CREG-109

We performed an evaluation on the 100 syntactically annotated Falko sentences[5]. RBG and TurboParser both produce structures with similar accuracy, but TurboParser is better at assigning dependency labels (see Table 2). jwcdg trails the other two parsers by more than 4 percentage points with respect to the unlabeled attachment score (UAS).

Overall, the performance degrades on L2 text relative to news text. On the HDT, TurboParser achieved an UAS of 93.66% (LAS: 91.35%) and RBG 93.20% (LAS: 90.76%). For both parsers, the attachment errors doubled on the Falko data.

[5]All evaluations exclude punctuation, since punctuation is always attached to 0 with an empty label in the annotation scheme and counting these attachments would only skew the results.

	RBG	Turbo	jwcdg	RBG_h
UAS	90.86	89.83	85.33	89.83
LAS	82.50	80.95	77.86	81.72

Table 3: Unlabeled and labeled attachment scores for RBG, TurboParser, jwcdg and the hybrid parser RBG_h (RBG augmented with constraints) on the CREG-109 corpus.

	LAS	UAS	LA
Falko-100dep	87.36	90.10	93.01
CREG-109	92.79	94.59	95.11
HDT	91.86	93.40	95.90

Table 4: The agreement of RBG and TurboParser on attachment scores (labeled and unlabeled) and label accuracy (LA)

We also evaluated how the language proficiency influences the parsing accuracy. Both RBG and TurboParser achieved the highest accuracy with considerable margin on C2 level data, i. e. data with little grammatical mistakes. In contrast, jwcdg performs best on B2 data, with only a small gap to the other parsers. This indicate a robustness of jwcdg against ill-formed input.

The results on CREG-109 are consistent with our findings on the Falko corpus (see Table 3). Compared to the results reported by Krivanek and Meurers (2011), RBG and TurboParser considerably outperform MaltParser on CREG-109, which is used for the automatically generated syntax layer of the FalkoEssayL2 corpus.

4.5 Analysis

RBG and TurboParser use the same feature sets and have a similar performance on Falko-100dep, CREG-109 and the HDT. However, they commit different errors as can be seen in Table 4. Notably, the difference on Falko-100dep is more pronounced than on the other two corpora.

On Falko-100dep, RBG and TurboParser assign most of the attachments (regent and label) with a similar recall and precision, but there are some major differences. Table 5 shows the attachments where RBG and TurboParser differ most. Moreover, RBG never correctly assigns (regent and label) the infrequent labels `EXPL` (expletive) and `OBJP` (prepositional object), whereas TurboParser at least identifies some of these dependencies cor-

	APP	KOM	OBJD
RBG recall	78.57	64.29	44.44
Turbo recall	71.43	100.00	22.22
RBG precision	91.67	60.00	57.14
Turbo precision	76.92	93.33	40.00

Table 5: Differences in attachments (regent and labels) on Falko-100dep. `APP`: apposition `KOM`: comparison word, `OBJD`: dative object

rectly (28.57% and 47.37% recall, 50% and 81.82% precision). Thus, RBG underrepresents the rare labels in its output.

The tagging error rate of TurboTagger (which is used by both RBG and TurboParser in our experiments) is 5.3%. jwcdg – which co-optimizes the PoS tags – only has an error rate of 4.5%. This difference highlights the benefits of optimizing the PoS tags together with the syntax.

4.6 Relabeling

Not only the syntactic structure but also the dependency labels are important for an analysis of a sentence. TurboParser assigns edge labels before parsing and therefore only uses information from the single edge. In contrast, RBG labels edges after the dependency tree is build, enabling it to use features from the dependency tree. Edge relabeling using information from the dependency tree (e.g. about neighboring edges) has proven to be beneficial for labeling accuracy (Köhn et al., 2014).

We use both the Maximum Entropy relabeler (MELabeler) described in Köhn et al. (2014) as well as TurboDependencyLabeler[6]. The relabelers were trained on part A of the HDT. Interestingly, both labelers actually decrease the labeling accuracy with respect to the original labeling by the parsers, as can be seen in Table 6. Since RBG and the relabelers assign labels after a dependency tree is build, it is not surprising that the labeling accuracy does not improve for RBG. We suspect that the noticeable decrease in labeling accuracy for TurboParser stems from the fact that the overall structure of a sentence is often not well-formed and an ill-formed part can influence more labels if the labeling decision is not made purely local. In a way, the relabelers overfit on well-formed data, whereas the simpler model employed by TurboParser cannot overfit in that way.

[6]which is distributed with TurboParser

	original	TurboLabeler	MELabeler
RBG	87.00	87.00	86.21
Turbo	87.42	85.30	86.03

Table 6: Labeling accuracy (in %) for the relabelers as well as the original labeling of the respective parsers.

5 Constraint-based Augmentation of RBGParser

RBG is able to generate accurate parses. If a sentence contains mistakes, which is the case for sentences acquired from language learners, the accuracy of RBG will decrease though, as we have shown in Section 4. Thus, there is room for improvement regarding the robustness of the parser.

We examined the RBG parses for 30 sentences from the FalkoEssayL2 corpus, that are not part of our Falko-100dep corpus, and for the first 20 sentences of Falko-100dep as follows: First, we evaluated the grammar of jwcdg for German on these parses. Next, we inspected the constraint violations for each word that RBG attached incorrectly. We observed that part of the wrong attachments violate constraints which express essential well-formedness conditions (for a parse) derived from the annotation guidelines. Such constraints e. g. express the requirement that a certain edge label goes together with specific PoS tags.

Because of this observation, we developed the idea to integrate weighted constraints via the scoring function of jwcdg into RBG to obtain parses that adhere to the basic annotation principles. For the remainder of this paper, we call the resulting hybrid parser RBG_h.

RBG is suitable for this approach because of its property that the generation of parses and the computation of their scores is separated and not interwoven as it is the case for TurboPaser.

5.1 Integrating RBGParser and weighted constraints

When generating new parses during hill-climbing, RBG evaluates the parses using its scoring mechanism (see Figure 1). The parses it determines as local maxima are compared to the best global solution up to that point. This is where the grammar integration takes place. When the scoring component is called to evaluate a local maximum parse, jwcdg scores this parse based on a grammar (see

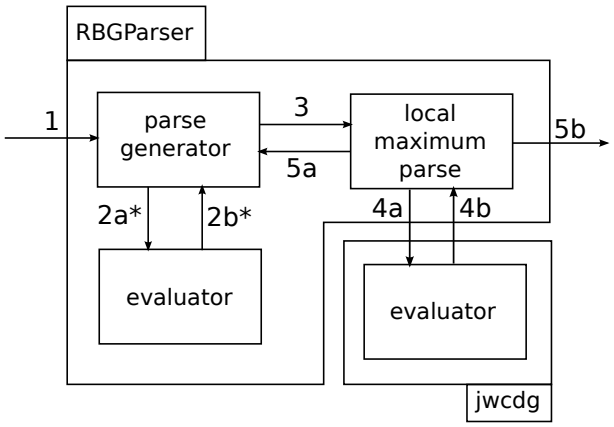

Figure 1: The process of parsing a sentence based on the hybrid approach of augmenting RBG with constraints. A sentence is given to RBG (1) and a parse is generated via hill-climbing performed on an initial random parse, with edges repeatedly passed on to the evaluator of RBG (2a) to receive their scores (2b), resulting in a local maximum parse (3), which is passed on to jwcdg (4a) to evaluate it on the grammar (4b). The RBG score of the local maximum parse is combined with the grammar penalty from jwcdg and is compared to the best parse so far. This procedure is repeated (5a) until the parsing converges (5b).

section 5.2). The score of jwcdg is converted into a penalty, the grammar penalty, which is combined with the rating of RBG into a single score.

In RBG, the syntactic labels are not assigned during decoding. For the hybrid parsing approach, RBG has been modified to assign syntactic labels already to edges of intermediate solutions. Thus, jwcdg has access to those labels in the hybrid setup.

To generate a score, the local maximum parses are passed on to jwcdg. It converts the parses into its internal representation, containing the word forms, their PoS tags and the dependency structure but no further lexical information. Then, jwcdg evaluates the constraints from the grammar and returns a score between 0 and 1, 0 for the violation of constraints that are not allowed to be violated under any circumstances, so-called hard constraints, and 1 if no violations occur at all.

One challenge is to combine the jwcdg and RBG scores since they are from different domains. We combine the two scores as follows: If, according to jwcdg, there are no constraint violations altogether, the RBG score is used without further modification. Otherwise, the jwcdg score is converted into a penalty and subtracted from the score of RBG. The lower the jwcdg score, the higher the respective penalty and vice versa. If a parse violates a hard constraint, the penalty is raised so that it becomes more probable that RBG_h prefers parses not violating any hard constraint over this parse.

5.2 Grammar

To evaluate the constraints RBG is augmented with, jwcdg is being used. Originally, jwcdg has a grammar for German that represents the German language as accurately as possible. It contains 1087 constraints. In this work, a subset of those constraints is used.

The constraints are divided into several groups. All groups for which the constraints are likely to be violated by L2 sentences are excluded. For example, all constraints are excluded from the grammar that are related to the word order or punctuation.

Because of the inherent lexical ambiguity of many word forms, groups of constraints which make use of lexical information were excluded. The reason is that RBG does not provide any form of lexicalization. Thus, jwcdg has to try to find an optimal one every time it receives a parse from RBG. This results in problems regarding the running time due to combinatorial issues. Another reason for omitting lexicalization are possible misspellings leading to wrong lexical information. The constraint groups remaining in the grammar deal with basic structural phenomena and express:

(a) which structure are licensed by the word categories in terms of PoS tags.

(b) which attachments to the root of a parse are allowed.

(c) that the labels of dependents of a word have to be unique for specific dependency labels.

(d) that particular attachments may not cross punctuation marks.

Some of the remaining constraints still depend on lexical information. Since no lexicon was used in the evaluation, those parts of the constraints would have evaluated to false, although no proposition could have actually been made. Therefore, those parts were relaxed so far that they do not influence the evaluation. If this was not possible, the respective constraint was removed from the grammar.

Also, only constraints were used whose violations mark severe mistakes. Less severe violations (which only encode preferences) were disregarded, resulting in a subset of 205 constraints, approximately a fifth of the original grammar. We call the resulting grammar minimal grammar for the remainder of this paper.

5.3 Error Analysis

As can bee seen in Table 2, the grammar integration has a negative effect on parsing performance. We compared the unlabeled attachments of the RBG and the RBG_h parses to find out why RBG_h performs worse. They differ in five parses. None of these five RBG_h parses have a higher UAS than their corresponding RBG parse, four parses have a lower one and one parse has the same. Overall, RBG_h attaches twelve words more incorrectly than RBG.

First, we checked whether the minimal grammar prevents RBG_h from selecting the gold standard parse for these five sentences, which is not the case: The gold standard annotations, including gold standard PoS tags and edge labels, are not penalized by the minimal grammar because none of them violates any constraints.

Next, we evaluated the minimal grammar on these five RBG and the RBG_h parses to answer the question why the grammar prefers the RBG_h parses to the RBG parses. Inspecting the parse with the same UAS shows that a constraint complains about an attachment in the RBG parse, which is indeed incorrect. Although the RBG_h parse has the same UAS, it represents the syntactic structure better than the RBG parse: In the gold standard annotation, the main clause is subordinated to its object clause contrary to the normal case where the object clause is subordinated. The annotation manual stipulates this attachment because otherwise a non-projective structure would arise for this sentence. If we disregard this projectivity rule and subordinate the object clause to the main clause, the RBG_h parse yields a higher UAS on the modified gold standard annotation (UAS: 34/39) than the RBG parse on both the gold standard annotation (32/39) and the modified gold standard annotation (31/19).

In case of the four RBG_h parses with a lower UAS, all of them have a lower grammar penalty than the respective RBG parses, three do not even violate any constraint. The four parses fall into two categories:

(a) The corresponding RBG parses violate hard constraints, even though RBG selected the correct regents for each of the rejected attachments. (3)

(b) An incorrect attachment rightfully violates a constraint in the corresponding RBG parse. (1)

The constraints for the parses under (a) are justifiably violated: One demands that the edge labels are consistent with the PoS tags of the dependent word, which is not the case in one parse due to a tagging error. The others require that a finite verb cannot have two complements of the same type, e. g. two subjects, which is not the case in two parses due to wrong edge labels.

Both PoS tags and edge labels cannot be changed retroactively by RBG_h: The PoS tags are determined beforehand by a PoS tagger and the labels are selected independently of the grammar penalty. Therefore, RBG_h has to change the attachments to achieve a lower penalty and, consequently, a better score in RBG_h itself. Figure 2 shows such a sentence, where RBG_h can not find the correct parse because it would require to change edge labels retroactively. Instead, RBG_h finds a parse with incorrect attachments but with a smaller grammar penalty.

The parse under (b) consists of two disconnected dependency trees although it should be connected. The RBG_h parse is connected at the price of an even lower UAS. The reason why RBG and RBG_h do not find the correct parse is probably due to faulty PoS tags (adjective and past participle are interchanged).

As we have seen before, RBG_h produces different parses only for five sentences and the UAS for these parses are at best the same as the UAS for the respective RBG parses. Thus, one hypothesis why the integration of constraints does not have a positive effect on the parsing performance is that the minimal grammar penalizes structures that occur in the gold standard annotation, and as a result prevents RBG_h from choosing the gold standard parses. To test this hypothesis, we evaluated the minimal grammar on the gold standard parses of the entire Falko-100dep corpus as well as on the RBG_h parses.

The minimal grammar does not prevent the parser from producing the correct parse: There are only two sentences for which the respective

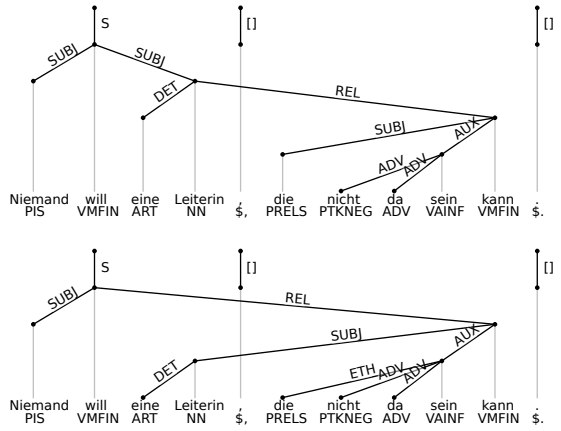

Figure 2: An example where the grammar integration deteriorates a parse ("Nobody wants a manager who cannot be there."). The RBG parse (top, UAS: 9/9) violates constraints of the minimal grammar because it assigns two subjects `SUBJ` to the finite verb "will". The RBG_h parse (bottom) is scored higher by the minimal grammar but has an UAS of only 6/9.

RBG_h parse has a lower grammar penalty than the gold standard. For both sentences the RBG parse is identical to the RBG_h parse.

The gold standard parses violate constraints in 18 sentences, which seems to be a high portion but the RBG_h parses violate constraints in 48 sentences. For 41 of these, the grammar penalty is higher than for the gold standard parse. For 32 of the 48 sentences, the gold standard does not violate any constraint.

The high amount of RBG_hparses violating constraints shows that RBG_h selects parses that are different from the gold standard annotations (including gold PoS tags), even though they are not preferred by the minimal grammar. This indicates that RBG_h can not find the gold standard parse. Reasons for this can be wrong PoS tags that the grammar penalizes or search errors due to the inability of RBG (and of RBG_h) to change dependency labels retroactively. As it turns out, this is indeed the case.

We examined the 32 sentences for which the RBG_h parse violates constraints but the gold standard does not. In 28 RBG_h parses, at least one constraint is violated due to wrong labels or wrong PoS tags. For the other 4 sentences, the constraints are rightfully violated because RBG_h chooses the wrong regent for a word. Why RBG_h did not select a different parse for these 4 sentences has still to be analyzed. Presumably, the reason is that the alternative parses do not have a lower grammar penalty because of PoS tag errors and the labeling issue.

6 Conclusions and Outlook

In this paper, different state-of-the art parsers were analyzed on free-form L2 sentences. For evaluation, we created gold-standard annotations for 100 sentences of the FalkoEssayL2 corpus.

We evaluated three different parsers on this language learner corpus. TurboParser and RBG, the two data-driven parsers, outperform jwcdg, the grammar-based parser. They produce comparable results, with TurboParser performing slightly better. Both parsers have more problems with B2 and C1 than C2 data. jwcdg on the other hand is more robust with respect to learner level. Furthermore, relabeling does not improve label accuracy.

Augmenting RBG with weighted constraints results in a decreased performance despite the grammar preferring the gold standard to the RBG output. Our analysis detected two main sources, namely erroneous PoS tags and wrong syntactic labels provided by RBG, which clash with the grammar because the hybrid parser pipeline cannot change either one retroactively. The future development of this hybrid parsing approach has to tackle the challenge of co-optimizing the syntactic labels and PoS tags during parsing in order to increase the robustness of this approach towards ill-formed data like L2 sentences.

The individual impact of the constraints has not been evaluated yet. Once the constraint-augmented parser co-optimizes, the constraint set can be optimized for the domain it is used for – in this case L2 data. If hand-written constraints only augment the statistical model of a data-driven parser, the effort needed to create these rules is orders of magnitude smaller than creating a full grammar for a rule-based parser. In addition, the constraints could be used for high-level symbolic context integration.

Currently, a detailed analysis of the differences between learner levels is hindered by the small size of annotated L2 sentences for German. For this, a larger corpus of syntactically annotated gold-standard L2 sentences for German needs to be gathered.

Our material can be obtained from `gitlab.com/nats/KONVENS-2016-material`.

References

[Berzak et al.2016] Yevgeni Berzak, Jessica Kenney, Carolyn Spadine, Jing Xian Wang, Lucia Lam, Keiko Sophie Mori, Sebastian Garza, and Boris Katz. 2016. Universal dependencies for learner English. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*, pages 737–746, Berlin, Germany, August. Association for Computational Linguistics.

[Beuck et al.2013] Niels Beuck, Arne Köhn, and Wolfgang Menzel. 2013. Predictive incremental parsing and its evaluation. In Kim Gerdes, Eva Hajičová, and Leo Wanner, editors, *Computational Dependency Theory*, volume 258 of *Frontiers in Artificial Intelligence and Applications*, pages 186 – 206. IOS press.

[Björkelund et al.2013] Anders Björkelund, Ozlem Cetinoglu, Richárd Farkas, Thomas Mueller, and Wolfgang Seeker. 2013. (re)ranking meets morphosyntax: State-of-the-art results from the SPMRL 2013 shared task. In *Proceedings of the Fourth Workshop on Statistical Parsing of Morphologically-Rich Languages*, pages 135–145, Seattle, Washington, USA, October. Association for Computational Linguistics.

[Brants2000] Thorsten Brants. 2000. TnT – a statistical part-of-speech tagger. In *Proceedings of the Sixth Conference on Applied Natural Language Processing*, pages 224–231, Seattle, Washington, USA, April. Association for Computational Linguistics.

[Dhar et al.2012] Arnab Dhar, Sanjay Chatterji, Sudeshna Sarkar, and Anupam Basu. 2012. A hybrid dependency parser for Bangla. In *24th International Conference on Computational Linguistics; Proceedings of the 10th Workshop on Asian Language Resources*, pages 55–64, Mumbai, India, December.

[Foth and Menzel2006] Kilian A. Foth and Wolfgang Menzel. 2006. Hybrid parsing: Using probabilistic models as predictors for a symbolic parser. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the ACL*, pages 321–328, Sydney, Australia. Association for Computational Linguistics.

[Foth et al.2000] Kilian A. Foth, Wolfgang Menzel, and Ingo Schröder. 2000. A transformation-based parsing technique with anytime properties. In *4th Int. Workshop on Parsing Technologies, IWPT-2000*, pages 89 – 100, Trento, Italy.

[Foth et al.2014] Kilian A. Foth, Arne Köhn, Niels Beuck, and Wolfgang Menzel. 2014. Because size does matter: The Hamburg Dependency Treebank. In Nicoletta Calzolari (Conference Chair), Khalid Choukri, Thierry Declerck, Hrafn Loftsson, Bente Maegaard, Joseph Mariani, Asuncion Moreno, Jan Odijk, and Stelios Piperidis, editors, *Proceedings of the Language Resources and Evaluation Conference 2014*, Reykjavik, Iceland, may. LREC, European Language Resources Association (ELRA).

[Foth2006] Kilian A. Foth, 2006. *Eine umfassende Constraint-Dependenz-Grammatik des Deutschen*. urn:nbn:de:gbv:18-228-7-2048.

[Khmylko et al.2009] Lidia Khmylko, Kilian A. Foth, and Wolfgang Menzel. 2009. Co-parsing with competitive models. In *Proceedings of the International Conference RANLP-2009*, pages 173–179, Borovets, Bulgaria. Association for Computational Linguistics.

[Köhn and Menzel2013] Arne Köhn and Wolfgang Menzel. 2013. Incremental and predictive dependency parsing under real-time conditions. In *Proceedings of the International Conference Recent Advances in Natural Language Processing RANLP 2013*, pages 373–381, Hissar, Bulgaria, September. INCOMA Ltd. Shoumen, BULGARIA.

[Köhn et al.2014] Arne Köhn, U Chun Lao, AmirAli B Zadeh, and Kenji Sagae. 2014. Parsing morphologically rich languages with (mostly) off-the-shelf software and word vectors. In *Proceedings of the 2014 Shared Task of the COLING Workshop on Statistical Parsing of Morphologically Rich Languages*.

[Krivanek and Meurers2011] Julia Krivanek and Detmar Meurers. 2011. Comparing rule-based and data-driven dependency parsing of learner language. In Kim Gerdes, Eva Hajicova, and Leo Wanner, editors, *Proceedings of Depling 2011*, pages 310–317.

[Lei et al.2014] Tao Lei, Yu Xin, Yuan Zhang, Regina Barzilay, and Tommi Jaakkola. 2014. Low-rank tensors for scoring dependency structures. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1381–1391, Baltimore, Maryland, June. Association for Computational Linguistics.

[Martins et al.2013] Andre Martins, Miguel Almeida, and Noah A. Smith. 2013. Turning on the turbo: Fast third-order non-projective turbo parsers. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 617–622, Sofia, Bulgaria, August.

[McDonald and Pereira2006] Ryan McDonald and Fernando Pereira. 2006. Online learning of approximate dependency parsing algorithms. In *EACL*, pages 81–88.

[Nivre2007] Joakim Nivre. 2007. Incremental non-projective dependency parsing. In *Human Language Technologies 2007: The Conference of the North American Chapter of the Association for Computational Linguistics; Proceedings of the Main Conference*, pages 396–403, Rochester, New York, April. Association for Computational Linguistics.

[Ott and Ziai2010] Niels Ott and Ramon Ziai. 2010. Evaluating dependency parsing performance on German learner language. In Markus Dickinson, Kaili Müürisep, and Marco Passarotti, editors, *Proceedings of the Ninth International Workshop on Treebanks and Linguistic Theories*, volume 9 of *NEALT Proceeding Series*, pages 175–186.

[Ragheb and Dickinson2011] Marwa Ragheb and Markus Dickinson. 2011. Avoiding the comparative fallacy in the annotation of learner corpora. In *Selected Proceedings of the 2010 Second Language Research Forum: Reconsidering SLA Research, Dimensions, and Directions*, pages 114–124, Somerville, MA. Cascadilla Proceedings Project.

[Ragheb and Dickinson2012] Marwa Ragheb and Markus Dickinson. 2012. Defining syntax for learner language annotation. In *Proceedings of the 24th International Conference on Computational Linguistics (COLING 2012), Poster Session*, pages 965–974, Mumbai, India, December.

[Ragheb and Dickinson2013] Marwa Ragheb and Markus Dickinson. 2013. Inter-annotator agreement for dependency annotation of learner language. In *Proceedings of the Eighth Workshop on Innovative Use of NLP for Building Educational Applications*, pages 169–179, Atlanta, Georgia, June. Association for Computational Linguistics.

[Rehbein et al.2012] Ines Rehbein, Hagen Hirschmann, Anke Lüdeling, and Marc Reznicek. 2012. Better tags give better trees – or do they? *Linguistic Issues in Language Technology (LiLT)*, 7(10), 1.

[Reznicek et al.2012] Marc Reznicek, Anke Lüdeling, Cedric Krummes, Franziska Schwantuschke, Maik Walter, Karin Schmidt, Hagen Hirschmann, and Torsten Andreas, 2012. *Das Falko-Handbuch*. http://www.linguistik.hu-berlin.de/de/institut/professuren/korpuslinguistik/forschung/falko/Falko-Handbuchv2.0.pdf.

[Sagae and Lavie2006] Kenji Sagae and Alon Lavie. 2006. Parser combination by reparsing. In *Proceedings of the Human Language Technology Conference of the NAACL, Companion Volume: Short Papers*, pages 129–132, New York City, USA, June. Association for Computational Linguistics.

[Schröder2002] Ingo Schröder. 2002. *Natural Language Parsing with Graded Constraints*. Ph.D. thesis, Universität Hamburg.

[Seeker and Kuhn2013] Wolfgang Seeker and Jonas Kuhn. 2013. Morphological and syntactic case in statistical dependency parsing. *Computational Linguistics*, 39(1):23–55, March.

[Zhang et al.2014] Yuan Zhang, Tao Lei, Regina Barzilay, Tommi Jaakkola, and Amir Globerson. 2014. Steps to excellence: Simple inference with refined scoring of dependency trees. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 197–207, Baltimore, Maryland, June. Association for Computational Linguistics.