Evaluation and comparison of the Stanford F-PCFG and BUBS constituency parsers

Yash Shah
University of Cambridge
ys562@cam.ac.uk

Abstract

Constituency parsing performance can be improved by relying on larger grammars, where additional grammar rules can be added that are sensitive to surrounding context. However, a larger grammar also requires more efficient decoding. The Stanford Factored Probabilistic Context-Free Grammar (F-PCFG) parser (Klein and Manning, 2002) with its incorporation of lexical information and the BUBS parser (Dunlop et al., 2011) utilizing adaptive beam-width prediction (Bodenstab et al., 2011) offer two such solutions. This report uses quantitative methods to provide an overview of general performance, as well as a qualitative study to provide a detailed analysis of each parser's merits and limitations on a set of sample sentences (found in Appendix A).

1 Introduction

A naively constructed PCFG (probabilistic context-free grammar) with rules and their associated probabilities constructed empirically from treebanks may perform poorly due to the context-free assumption (Charniak, 1996; Klein and Manning, 2003). This assumption is often too strong (for example, believing that subject and object noun phrases share the same distribution) or may even be too weak (such as in cases involving long rewrites where parts of a sentence depend on each other over a large distance) (Petrov et al., 2006).

Lexicalization and latent-variable grammars are two of a variety of techniques to both refine and generalize the grammar to help improve parsing performance (Jamison, 2011), and are utilized by the Stanford F-PCFG and BUBS parsers respectively. Note that the BUBS parser is grammar-agnostic, but was specifically used with the Berkeley split-and-merge (SM6) latent-variable grammar (Petrov, 2017).

This paper compares the performance of both of these parsers quantitatively and qualitatively and is structured as follows. Section 2 provides details for exact reproduction of results and section 3 gives an overview of the sample data and modifications made to it. The workings of the parsers are described in sections 4 and 5. General performance of the parsers is evaluated using the PARSEVAL metric (Black et al., 1991) in section 6. A detailed discussion of parser performance involving specific merits and limitations, error categorization, severity judgments, explanation, and methods of addressing them are discussed in Section 7. The conclusion of this report can be found in section 8, with limitations discussed in 9.

2 Reproduction of results

All tools (parsers, grammar files, analysis tools, etc.), data (e.g., input sentences), and instructions for exact reproduction of results can be found in this report's associated GitHub repository [1].

Additionally, this project utilizes Nix (Dolstra et al., 2004)—a tool allowing for a reproducible environment for running of the parsers and analysis software, as well as compilation of associated tools, without undeclared dependencies.

The 2015-01-29 version of the Stanford parser and 20131221 version of the BUBS parser was used.

3 Sample data

The sample data consists of 13 sentences (found in Appendix A). The first 10 are from a gold standard and an additional 3 were selected from a crowd-sourced standard for unique structures they exhibited. A number of modifications were made to the first 10 gold standard sentences to enable closer conformance with the Penn Treebank bracketing guidelines (Bies et al., 1995a), which are detailed next.

[1] https://github.com/Paroxysmisch/parser-comparison

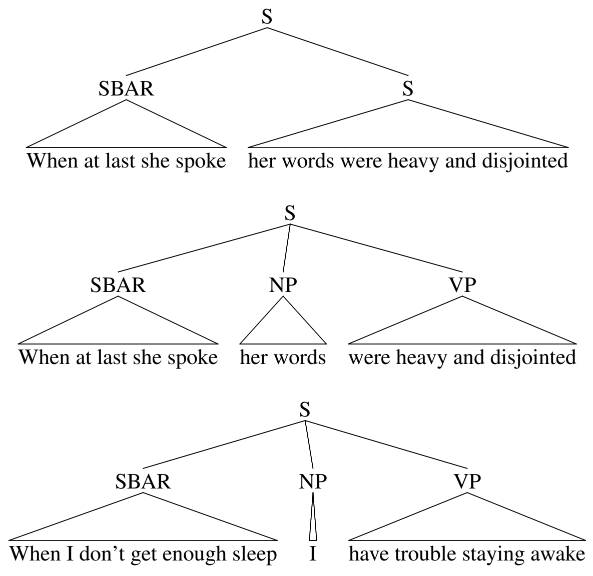
S
SBAR
S
When at last she spoke
her words were heavy and disjointed
S
SBAR
NP
VP
When at last she spoke
her words
were heavy and disjointed
S
SBAR
NP
VP
When I don't get enough sleep
I
have trouble staying awake

Figure 1: Top: gold standard sentence 5. Middle: altered sentence 5. Bottom: Reference sentence from Penn Treebank bracketing guidelines. Front arguments and adjuncts are at the same level as NP and VP.

The gold standard consistenly uses a binary branching structure for fronted elements, while the guidelines require a flatter structure, where the fronted element (e.g. PP, ADVP, or SBAR) is placed at the "top clause level" (Bies et al., 1995b) (Figure 1).

The guidelines recognize that consistently annotating argument/adjunct distinction is difficult and so recommends that all PP modifiers of nouns are Chomsky-adjoined to the NP (Bies et al., 1995c) (Figure 2).

Also, phrase coordination should be represented at the lowest level possible with single words coordinating at the word level, instead of projecting their own phrases, unless modifiers are present (Bies et al., 1995d). Thus, phrases such as `(VP (VP (VBD came)) (CC and) (VP (VBD went)))` (sentence 7) have been transformed to `(VP (VBD came) (CC and) (VBD went))`. Subtler changes include not labeling single-word ADJPs (Bies et al., 1995e), and always labeling "to" as TO (Santorini, 1990).

Sentence 9 was the only one that was substantially changed. "rain" is vacuously the head noun of the sentence so should not be nested in a PP, and "All through August" is a frontal prepositional modifier for the complete sentence "the rain hardly stopped". This, combined with the guideline's preference for flatter structure leads to Figure 3.

The same modifications were also made to the additional 3 sentences, along with changes including flattening the parenthetical structure, CONJP removed where there is a single-word conjunction (as it is reserved for multi-word conjunctions (Bies et al., 1995f)), and labeling of relative clauses when introduced by relative pronouns and adverbs (Bies et al., 1995g).

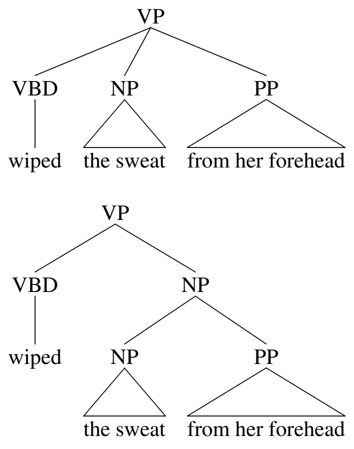

Figure 2: Top: gold standard (partial) sentence 2. Bottom: altered (partial) sentence 2. The PP is now Chomsky-adjoined to the NP.

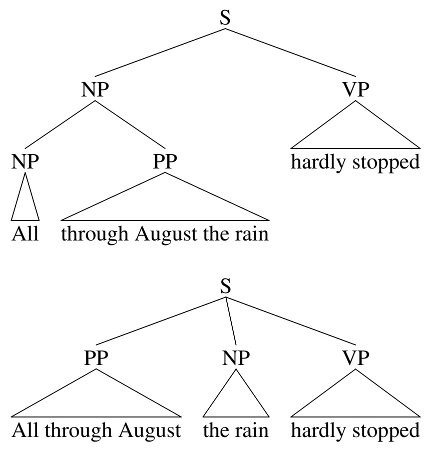

Figure 3: Top: gold standard sentence 9. Bottom: altered sentence 9. "the rain" is no longer part of the PP.

4 Stanford F-PCFG parser

Lexical preferences are known to be effective in resolving modifier and argument attachment ambiguities and can be incorporated into the notion of parsing by producing a lexicalized tree from text. A lexicalized tree can be viewed as the pair $L = (T, D)$ of a phrase structure tree T and dependency tree D.

If we approach parsing the lexicalized PCFG as if it were simply a larger unlexicalized PCFG using the Cocke–Younger–Kasami (CKY) (Younger, 1967) dynamic programming (chart-based) algorithm, we are led to an impractical $\mathcal{O}(n^5)$ combined parse cost. This is due to symbol-explosion where for each PCFG symbol, say NP, we have a whole collection of new symbols NP-x for all possible heads x in the vocabulary. Additionally, each rule becomes a family of rules, with each symbol parameterized with a head.

While a beam can be maintained over the CKY algorithm to maintain the $\mathcal{O}(n^3)$ complexity, this means potentially missing the optimal solution. Instead the *factored* Stanford F-PCFG parser notes that the phrase and dependency structure need not be jointly modelled $\mathcal{P}(T, D) = \mathcal{P}(T)\mathcal{P}(D)$ (where $\mathcal{P}$ refers to the probability of the tree).

Regular PCFG parsers (like Stanford U(nlexicalized)-PCFG) can be used to model $\mathcal{P}(T)$ and lexical dependency models for $\mathcal{P}(D)$. These sub-models are then used to calculate outside-scores for each edge (the core parse item in CYK parsing, where a non-terminal spans a segment of the input string), *not* the optimal parse. Next, an agenda-based parsing approach is used where edges that have been created, but have not yet been used to create new edges are put into a priority queue. If the inside-score (Manning and Schutze, 1999) of each edge is used as the priority queue's ranking function, then optimality is guaranteed in a proof similar to that of Dijkstra's algorithm.

This alone however, does not provide the speedup necessary. Fortunately, optimality is still guaranteed if we add to the inside score, any admissible heuristic (one that never overestimates the cost of reaching the goal) of completing the parse using that edge (proof similar to A* search). The sum of the two sub-models' outside scores is one such admissible heuristic.

In this way, the Stanford F-PCFG can efficiently utilize lexicalization to improve parse quality, while guaranteeing optimality of search (e.g. not using a beam that discards possibilities).

5 BUBS parser

The complexity of the CYK algorithm is $\mathcal{O}(n^3|G|)$, with $|G|$ representing the size of the grammar. As it is a constant, it is often omitted, however it has a non-neglibible impact on parsing speed when using large latent-variable (or lexicalized) grammars.

While other agenda-based parsers maintain a global priority queue (agenda) of edges ranked by their figure-of-merit (FOM) (an estimate of the product of an edge's inside and outside scores), BUBS maintains such an agenda for each CYK chart cell. Not only does this avoid problems when comparing edges spanning different lengths of the original sentence (an edge's probability decreases with increasing size as it is a product of more rules), but also improves speed as there is less agenda overhead and allows for parallelization across the CYK chart.

The BUBS paper notes that in 96% of cases, an edge belonging to the maximum likelihood parse tree belongs to the top-3 predicted edges in its respective CYK chart cell. This means that there are steep diminishing returns for increasing beam-width uniformly across the chart.

Instead, BUBS selectively increases the size of the beam-width only in ambiguous cases. The beam width for each cell is selected using a pre-trained log-linear model. In actual implementation, this is a collection of binary classifiers $c_1, \ldots, c_k$ each predicting whether the maximum likelihood edge belongs in the top-k ranking respectively (for that edge's cell). If all classifiers predict the maximum likelihood edge belongs outside the top-k, this is the case of maximal ambiguity, and thus the largest allowable beam-width is chosen.

6 PARSEVAL analysis

When the parsers are evaluated by evalb (Sekine et al., 2017a,b) (an implementation of the PARSEVAL metric (Black et al., 1991) that evaluates the parser's output by comparing the identity of labels and word spans to those in the gold trees) (data in Table 1), the BUBS parser performs markedly better than the Stanford F-PCFG parser, especially with regards to recall (85.08% compared to 78.45%).

BUBS gains this recall advantage over F-PCFG in sentences 10 and 13. This was because F-PCFG failed to capture, in both cases, the fundamental clause structure. For example, in sentence 10, "he didn't have to know if this" is a relative clause, however, F-PCFG marks this as the main clause of the sentence. This leads to a flood of divergences from the ground truth as "he", for example, is marked as the head noun phrase. This is a major issue as changing the head clause leads to drastically different semantics and also means that there are more errors in other parts of the parse which depend on

Parser	Bracketing Precision	Bracketing Recall	Bracketing F-score	Tag Accuracy
Stanford U-PCFG	79.55	77.35	78.43	93.89
Stanford F-PCFG	82.56	78.45	80.45	95.00
BUBS	**86.03**	**85.08**	**85.56**	**95.56**

Table 1: This tables displays the bracketing precision, recall, and F-score, as well as tagging accuracy of each parser on the sample dataset. Best results shown in **bold**.

	JJ	VB	IN	NNS	WDT
RB	1		2		
VBP		2			
WDT			1		
WBZ				1	
DT					1*

Table 2: The left side shows the true part-of-speech tag and the top shows tags incorrectly assigned by the parsers. The number corresponds to the occurrences of that mislabeling. * corresponds to only the Stanford parser making the error. The other errors were made by both Stanford F-PCFG and BUBS parsers.

fundamental structure.

The F-PCFG errors in sentence 10 likely stem from poor incorporation of lexical information. F-PCFG utilizes U-PCFG for its phrase structure tree, and U-PCFG performs well on this sentence. However, since F-PCFG uses the sum of the outside scores from both the phrase and dependency sub-models, good parsing performance relies on both of these sub-models agreeing on the outside probability for the maximum likelihood edges.

The dependency sub-model likely does not rank the true gold standard edge nearly as highly as the phrase sub-model, allowing for dramatic structural change. This could be improved by incorporating a penalty term within the F-PCFG heuristic that scales with the difference between phrase and dependency outside probabilities (ensuring this is still an admissible heuristic).

Regarding tag accuracy, both parsers perform very well ($> 95\%$), with the exception of sentence 12, where both score 76.47% (4 errors). Interestingly, both mark “strains” in the phrase “strains resources to breaking point” as a plural noun (NNS), instead of a verb (VBZ). This may be because both parsers have a bias towards expecting S→NP VP constructions for new (relative) clause constructions. This is also a fairly important error as correct identification of verbs is important for semantic understanding and may impact the performance of downstream NLP tasks considerably.

Table 2 summarizes all the tagging errors made by the parsers. We see a common trend that both struggle similarly with correct identification of adverbs in compound phrases such as "at last". These are minor in their impact since the phrase is still correctly identified and parsed as a whole. This could be fixed by performing a pre-processing pass where such compound phrases are identified and pre-assigned part-of-speech tags (possibly planting them with higher weighting into the appropriate CYK cells to still allow the parser to decide what it finally chooses).

From Figure 4 we see that the ability of all parsers to recall bracketing from the sample data decreases with increasing sentence length, but the F-PCFG parser's recall performance appears to drop quicker than BUBS. Eventhough recall drops with length, precision remains somewhat constant with both parsers.

A possible explanation for this is the empirical observation that both parsers prefer flatter tree structures with larger branching factors compared to the sample sentences. Thus, fewer brackets are generated, decreasing recall, but maintaining precision as spurious bracketing is avoided. Since longer sentences have more constituents and details in the form of linguistic structures such as prepositional phrases, there are more opportunities for the parsers to avoid complex, deeply-nested bracketing, reducing recall. However, since the overall parser performance is still strong within each of these smaller constituents, precision continues to remain high.

By running PARSEVAL with F-PCFG's outputs as the gold standard, and BUBS's outputs as the test, we find that both parsers produce fairly similar results (note that flipping the evaluation order just flips the precision and recall metrics here). The tag accuracy of 98.33% shows that both parsers' part-of-speech tagging output is well-aligned.

The recall, precision, and F-score of 87.15%, 90.70%, and 88.89% show that inter-parser agreement is greater than each parser's with the sample data. This means that the parsers largely make the

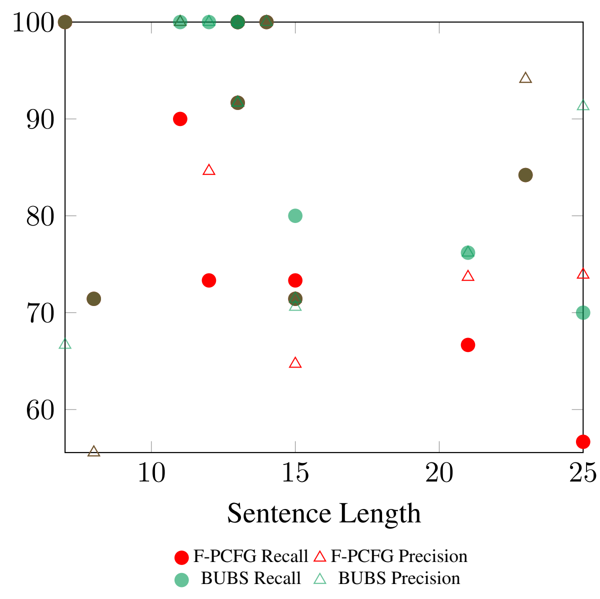

Figure 4: Scatter plot of precision and recall with sentence length for both parsers. Note the darker colors are where results overlap.

Error Type	Occurences	Nodes	Ratio
PP Attach	4	5	1.25
Clause Attach	4	7	1.75
Diff Label	1	2	2.0
Mod Attach	4	12	3.0
NP Attach	2	7	3.5
Co-ord	0	0	-
1-Word Span	11	11	1.0
Unary	7	7	1.0
NP Int	1	1	1.0
Other	3	3	1.0

Table 3: This table shows the number of occurrences (tree transformations required) to correct the Stanford F-PCFG parser's output, number of nodes affected, and the ratio of nodes to occurences with respect to each error type.

Error Type	Occurences	Nodes	Ratio
PP Attach	3	4	1.3
Clause Attach	1	2	2.0
Diff Label	0	0	-
Mod Attach	4	6	1.5
NP Attach	0	0	-
Co-ord	0	0	-
1-Word Span	11	11	1.0
Unary	3	3	1.0
NP Int	0	0	-
Other	4	4	1.0

Table 4: Analogous to Table 3, but for BUBS.

same types of errors, although this agreement declines significantly with increasing sentence complexity (e.g. recall of 71.43% with the very complex sentence 12), due to varied handling of linguistic structures such as relative clauses and prepositional phrases. Unsurprisingly, since longer sentences tend to be more complex, this agreement also declines with increased sentence length.

7 Berkeley Parser Analyzer results

This section focuses on collating and quantifying the different errors types from both parsers. The approach taken here uses the methods and software (Berkeley Parser Analyzer) laid out in (Kummerfeld et al., 2012a). Simple error breakdown by the type of the node may not be very informative since, for example, a single attachment error can lead to multiple errors stemming from the same issue.

(Kummerfeld et al., 2012a) proposes a two-stage solution to linguistically meaningful error classification. First, a series of tree transformations are applied to convert the output into the gold tree, using a template-based system utilizing spans of words. Next, the transformations applied to correct the parser's output are mapped to error types.

7.1 PP attachment

A PP attachment error occurs any time a transformation involves the movement of a prepositional phrase, or the bracket over the prepositional phrase is incorrect. This type of error is fairly frequent in both parsers (as shown in Tables 3 and 4) with 4 and 3 occurrences for F-PCFG and BUBS respectively. An example is sentence 9, where both parsers fail to accurately capture the complete prepositional phrase "All through August". Figure 5 shows that both parsers fail differently in the same scenario.

While F-PCFG correctly captures the "All through August" constituent, it wrongly propagates "All" (which is acting as a specifier in this case) as the head of the sentence instead of "through", which then results in a PP attachment error as the constituent is wrongfully labeled as an NP. This issue was not caused by poor integration of lexical information as U-PCFG makes the same error. Perhaps PP→DT PP constructions are rare in the training corpus.

Comparatively, BUBS fails to even capture the "All through August" as a single phrase, resulting in a PP attachment error as "through August" is directly attached to the top-level S. It seems strange how BUBS allows a determiner to stand alone as a complete NP, and disallowing this could be one way of fixing this issue.

A less serious PP attachment issue was F-PCFG not Chomsky-adjoining the prepositional phrase "from her forehead" to the noun phrase "the sweat" in sentence 2 as stated by the Penn Treebank bracketing guidelines (Bies et al., 1995c).

Errors in PP attachment introduce ambiguities in discerning sentence structure and meaning. For example, the sentence (not in the test set) "I saw a man with a telescope" could be read as either the seeing was being done with the telescope, or the man had a telescope. This is a serious error, as it can negatively impact downstream tasks such as information extraction. Additionally, we see that each PP attachment error affects > 1.2 nodes, showing that it can lead to secondary errors, further increasing its severity.

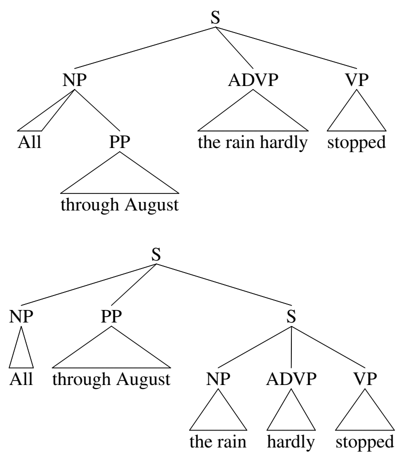

Figure 5: Top: Stanford F-PCFG sentence 9. Bottom: BUBS sentence 9. There are PP attachment issues present in both parsers. The F-PCFG PP attachment error leads to another problem regarding incorrect grouping and identification of "the rain hardly" as an adverbial phrase.

7.2 Clause attachment

When any form of S node is moved, this constitutes a clause attachment error. We see that the F-PCFG parser is much more vulnerable to these mistakes, with 4 compared to a single occurrence for BUBS. While most of the attachment errors are fairly benign (including the BUBS error), F-PCFG is also prone to more serious problems.

For example, in Figure 6 we see that the clause "which we wanted to eat" should first be grouped with the NP "the rabbits". This is because they are eating the rabbits, and not the horses. However, F-PCFG fails to recognize this, instead attaching the relative clause at the level of "The horse" and "... the rabbits", implying that they wanted to eat the horses too. This is a severe error as it changes the semantics of the sentence, and could greatly reduce the performance of downstream tasks such as machine translation, or questions answering (e.g. *What was eaten?*). It also disproportionately affects the parse with a high nodes to occurrences ratio.

A possible solution would be better incorporation of verb forms to alleviate this clausal attachment ambiguity. The verb "has" is correctly tagged VBZ, indicating that the parser understands that it is singular. Since it is singular, it cannot apply to both the horse and the rabbits. Thus, this would help the parser decide the correct SBAR attachment.

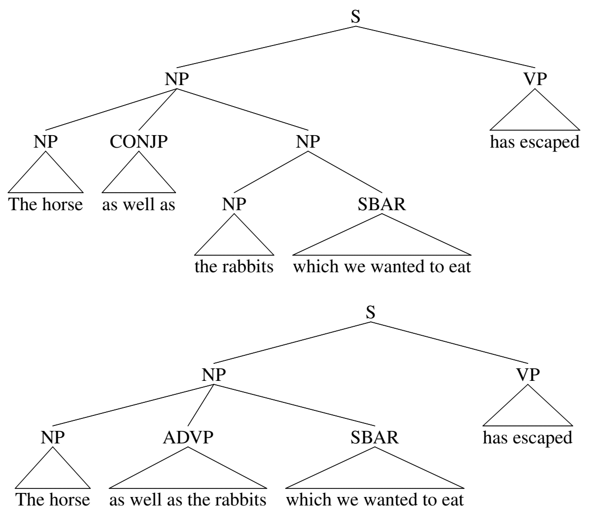

Figure 6: Top: (Modified) gold sentence 11. Bottom: Stanford F-PCFG sentence 11. There is a severe clausal attachment issue with the subordinate clause (SBAR) in the bottom parse.

The secondary type of clause attachment issue was present in both parsers (but more so in F-PCFG). This was to do with an over-flattening of the tree structure (shown in figure 7), but is not very problematic as the sentence semantics are not affected. It could be caused by the fact that both parsers use the probability of the tree structure (inside/outside scores) to rank their parses. Deeper trees, with reduced branching factors, will have lower probabilities, and are thus avoided.

Further credence can be given to this argument since F-PCFG uses a global agenda, whereas BUBS's local agenda was specifically designed to reduce the impact of this issue. With BUBS, since rankings are only made within the CYK chart cell, the local agenda only compares edges of the same span. While this does not completely alleviate the problem as the algorithm can still prefer parses using fewer total rules, it appears to have helped, as BUBS reports a much lower number of clause attachment errors.

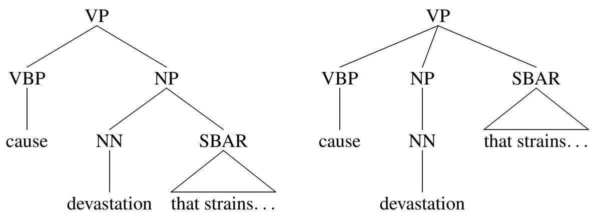

Figure 7: Top: partial gold sentence 12. Bottom: partial BUBS sentence 12. Both parsers favor flatter structures.

7.3 Modifier attachment

Incorrectly placed adjectives and adverbs, as well as corrective subtree movements involving the creation of nodes such as ADVP are classified as modifier attachment errors. Both parsers have trouble associating adverbs with the correct clauses. Figure 8 shows that both parsers misplace the fronted adjunct "Prudently" by attaching it directly to the top-level sentence, rather than the clause "they had diversified into banking and finance". This shows that both parsers fail to capture the semantic relation that "Prudently" is acting on the verb "diversified".

While modifiers act to add more information to a phrase, this additional information can be key in understanding the true semantics, especially in cases where modifiers are instances of negation. Thus, this is an important error.

One way this could be resolved is if the parsers made better use of punctuation and the conjunction present. There is a comma between "finance", and the coordinating conjunction "and" in sentence 3. This clearly hints that the ideas to the left of the conjunction should not interact with ideas from the right (so that the top-level structure is S→S CC S). Attaching "Prudently" to the top-level S would violate this, and so this issue would be resolved.

7.4 NP attachment

Errors which involve the movement of noun phrases are categorized as NP Attachment problems. NP attachment errors are uncommon with F-PCFG, and never occurred in any of the test sentences with BUBS.

Additionally, the NP attachment errors with F-PCFG were an indirect consequence of another issue. For example, in Figure 9, we see that the

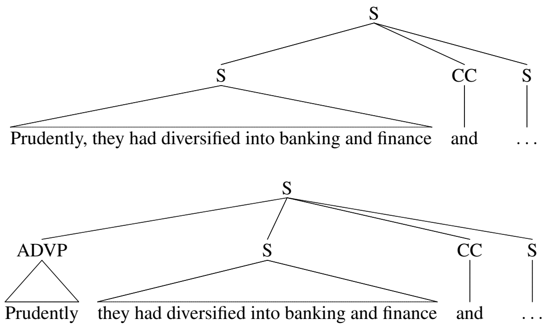
S
S
CC
S
Prudently, they had diversified into banking and finance
and
. . .
S
ADVP
S
CC
S
Prudently
they had diversified into banking and finance
and
. . .

Figure 8: Top: fragment of gold sentence 3. Bottom: fragment of both Stanford F-PCFG and BUBS sentence 3—both parsers produced the same modifier attachment error. Punctuation nodes omitted for clarity.

noun phrase around "resources" is initially missing. Then, due to a part-of-speech tagging error with "strains", which is incorrectly labeled as a plural noun, the "resources" noun phrase is attached to the prepositional phrase "to breaking point", rather than first forming a verb phrase with "strains".

Although correct NP attachment is important for semantic understanding of the sentence, the errors are relatively rare, and do not cause major issues that would inhibit the ability for downstream tasks like machine translation to perform well (in the cases observed—not in general). Since the issues regarding NP attachment were the result of other parts of the parser performing poorly, this problem can likely be automatically resolved as those troublesome parts (e.g. the part-of-speech tagging) are improved.

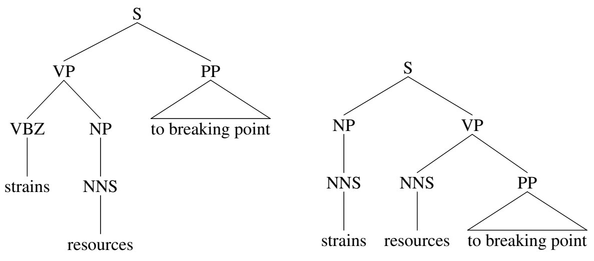

Figure 9: Top: fragment of gold sentence 12. Bottom: fragment of both Stanford F-PCFG sentence 12. There is an NP attachment issue with "resources".

7.5 Single word phrase

Single word phrase (or 1-word span) errors were the most common type of error with 11 occurrences in each of the two parsers. Despite this, they are not very problematic, as their scope for damaging the rest of the parse is constrained (nodes to occurences ratio of 1.0) and the instances were mostly minor.

One common instance of these errors was a result of nonconformity to the Penn Treebank bracketing guidelines on issues such as not wrapping single-word nouns (NNs) such as “banking” and “insurance” (sentence 3) in NPs. The other common instance was errors in the part-of-speech tagging. For example F-PCFG and BUBS both tag “last” in “at last” (sentence 5) as an adjective, instead of an adverb. Both parsers tend to make the same single word phrase errors, and this was likely a large contributor to high inter-parser agreement.

The only serious instance of this error is in sentence 9 (Figure 5) where the singleton determiner “All” is labeled as a noun phrase. This means that F-PCFG got the prepositional structure of the phrase “All through August” incorrect. A fix for this is forbidding phrases solely containing a determiner to stand as complete noun phrases.

7.6 Unary and all other error types

The unary errors correspond to often missing or sometimes extra unary productions that are not linked to nearby errors, and so are not very problematic. The missing productions likely arise as the training corpus for both parsers involves parses where these were sometimes skipped. Since additional productions decrease the probability of the final parse, they may have been avoided.

The one different label error for F-PCFG was in sentence 9, regarding the mislabeled NP (instead of being a PP). The NP Internal and Other error type were simply the result of applying minor corrections for both parsers.

The BUBS parser encounters fewer occurrences of such minor errors, as indicated by the lower count for Unary and Other types of errors.

8 Conclusion

In conclusion, the BUBS parser outperforms the Stanford F-PCFG parser in most cases, with a higher bracketing F-score and tagging accuracy. Additionally, it suffers from fewer minor, and major failures (such as avoiding the Stanford F-PCFG’s clause attachment error).

BUBS also tends to maintain better precision and recall with increasing sentence length, which is important as real-world sentences are often longer than those in the test set.

However, it is important to note that BUBS is not perfect and makes its own severe PP attachment errors. Both parsers perform worse than their F-scores would suggest in (Kummerfeld et al., 2012b) where Stanford F-PCFG and BUBS score 85.78 and 88.50 respectively. Stanford F-PCFG’s lexicalized grammar proved to be less useful than expected, especially in cases involving clause and PP attachment ambiguities, while BUBS with the Berkeley SM6 latent-variable grammar often proved more effective.

9 Limitations

The PARSEVAL and Berkeley Parser Analyzer evaluation techniques used in this paper are not perfect.

Since PARSEVAL directly measures only the syntactic correctness of the produced parse, it cannot distinguish the severity of linguistic errors, or give even partial credit in cases where, for example, the syntactic categories were correctly identified, but with a slight error in the phrase boundary. Thus, severe mislabellings can have minor impacts on the final score, while benign differences like preferring a flatter branching structure can dramatically alter the metric. In fact, prior work (Buckley and Voorhees, 2017) has warned against the use of PARSEVAL for comparing parsers trained on treebanks with different annotation schemes.

A key strength of the Berkeley Parser Analyzer is that it categorizes errors according to the tree transformations necessary to fix them. However, this makes the error classification sensitive to the ordering and selection of which previous transformations were applied. To avoid large runtime cost, it also uses a greedy, bottom-up approach to fixing the parse tree, which can be sub-optimal, by introducing spurious error classes, while avoiding the root cause of the problem.

Due to the limitations of such automated parsing metrics, this paper uses both a quantitative and qualitative approach for evaluation. However, this report is not immune to problems either. While most of the gold standard adjustments can be justified with treebank guidelines, ambiguities are introduced by the guidelines omitting issues like the conditions for a certain branching-factor preference. The small test set may fail to surface rarer (but possibly severe) errors, and may inaccurately report the error distribution statistics. Specific downstream use cases (like machine translation) were also not evaluated, which would impact which errors are considered more important for parser performance.

References

Ann Bies, Mark Ferguson, Karen Katz, Robert MacIntyre, Victoria Tredinnick, Grace Kim, Mary Ann Marcinkiewicz, and Britta Schasberger. 1995a. Bracketing guidelines for treebank ii style penn treebank project. *University of Pennsylvania*, 97.

Ann Bies, Mark Ferguson, Karen Katz, Robert MacIntyre, Victoria Tredinnick, Grace Kim, Mary Ann Marcinkiewicz, and Britta Schasberger. 1995b. Bracketing guidelines for treebank ii style penn treebank project. *University of Pennsylvania*, 97:29–31.

Ann Bies, Mark Ferguson, Karen Katz, Robert MacIntyre, Victoria Tredinnick, Grace Kim, Mary Ann Marcinkiewicz, and Britta Schasberger. 1995c. Bracketing guidelines for treebank ii style penn treebank project. *University of Pennsylvania*, 97:14.

Ann Bies, Mark Ferguson, Karen Katz, Robert MacIntyre, Victoria Tredinnick, Grace Kim, Mary Ann Marcinkiewicz, and Britta Schasberger. 1995d. Bracketing guidelines for treebank ii style penn treebank project. *University of Pennsylvania*, 97:26–27.

Ann Bies, Mark Ferguson, Karen Katz, Robert MacIntyre, Victoria Tredinnick, Grace Kim, Mary Ann Marcinkiewicz, and Britta Schasberger. 1995e. Bracketing guidelines for treebank ii style penn treebank project. *University of Pennsylvania*, 97:26–27.

Ann Bies, Mark Ferguson, Karen Katz, Robert MacIntyre, Victoria Tredinnick, Grace Kim, Mary Ann Marcinkiewicz, and Britta Schasberger. 1995f. Bracketing guidelines for treebank ii style penn treebank project. *University of Pennsylvania*, 97:28.

Ann Bies, Mark Ferguson, Karen Katz, Robert MacIntyre, Victoria Tredinnick, Grace Kim, Mary Ann Marcinkiewicz, and Britta Schasberger. 1995g. Bracketing guidelines for treebank ii style penn treebank project. *University of Pennsylvania*, 97:20–21.

E. Black, S. Abney, D. Flickenger, C. Gdaniec, R. Grishman, P. Harrison, D. Hindle, R. Ingria, F. Jelinek, J. Klavans, M. Liberman, M. Marcus, S. Roukos, B. Santorini, and T. Strzalkowski. 1991. A procedure for quantitatively comparing the syntactic coverage of English grammars. In *Speech and Natural Language: Proceedings of a Workshop Held at Pacific Grove, California, February 19-22, 1991*.

Nathan Bodenstab, Aaron Dunlop, Keith Hall, and Brian Roark. 2011. Adaptive beam-width prediction for efficient cyk parsing. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics*, pages 440–449.

Chris Buckley and Ellen M Voorhees. 2017. Evaluating evaluation measure stability. In *ACM SIGIR Forum*, volume 51, pages 235–242. ACM New York, NY, USA.

Eugene Charniak. 1996. Tree-bank grammars. In *Proceedings of the National Conference on Artificial Intelligence*, pages 1031–1036.

Eelco Dolstra, Merijn De Jonge, Eelco Visser, et al. 2004. Nix: A safe and policy-free system for software deployment. In *LISA*, volume 4, pages 79–92.

Aaron Dunlop, Nathan Bodenstab, and Brian Roark. 2011. Efficient matrix-encoded grammars and low latency parallelization strategies for cyk. In *Proceedings of the 12th International Conference on Parsing Technologies*, pages 163–174.

Emily Jamison. 2011. Using grammar rule clusters for semantic relation classification. In *Proceedings of the ACL 2011 Workshop on Relational Models of Semantics*, pages 46–53.

Dan Klein and Christopher D Manning. 2002. Fast exact inference with a factored model for natural language parsing. *Advances in neural information processing systems*, 15.

Dan Klein and Christopher D Manning. 2003. Accurate unlexicalized parsing. In *Proceedings of the 41st annual meeting of the association for computational linguistics*, pages 423–430.

Jonathan K. Kummerfeld, David Hall, James R. Curran, and Dan Klein. 2012a. Parser showdown at the wall street corral: An empirical investigation of error types in parser output. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 1048–1059, Jeju Island, South Korea.

Jonathan K Kummerfeld, David Hall, James R Curran, and Dan Klein. 2012b. Parser showdown at the wall street corral: An empirical investigation of error types in parser output. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 1048–1059.

Christopher Manning and Hinrich Schutze. 1999. *Foundations of statistical natural language processing*, pages 388–402. MIT press.

Slav Petrov. 2017. Berkeley parser. `https://github.com/slavpetrov/berkeleyparser`.

Slav Petrov, Leon Barrett, Romain Thibaux, and Dan Klein. 2006. Learning accurate, compact, and interpretable tree annotation. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, pages 433–440.

Beatrice Santorini. 1990. Part-of-speech tagging guidelines for the penn treebank project.

Satoshi Sekine, Michael John Collins, David Ellis, David Brooks, and Shlomi Hod. 2017a. Evalb. `https://github.com/shlomihod/Evalb`.

Satoshi Sekine, Michael John Collins, Terry Koo, David Brroks, David Ellis, and Don Blaheta. 2017b. Evalb. `https://github.com/shlomihod/Evalb`.

Daniel H. Younger. 1967. Recognition and parsing of context-free languages in time n3. *Information and Control*, 10(2):189–208.

A Sample sentences

Num.	Sentence
1	As she walked past it, the driver's glass started to open.
2	With a handkerchief she wiped the sweat from her forehead.
3	Prudently, they had diversified into banking and insurance, and as a result their influence was felt at the highest level.
4	The arranged marriage would be the social event of the following year.
5	When at last she spoke, her words were heavy and disjointed.
6	The road to the coast was busy with traffic in both directions.
7	The expected date came and went.
8	She sighed at the irony of it all, the waste of it all.
9	All through August the rain hardly stopped.
10	Thank the gods he didn't have to know of this.
11	The horse as well as the rabbits which we wanted to eat has escaped.
12	Natural disasters – storms, flooding, hurricanes – occur infrequently but cause devastation that strains resources to breaking point.
13	Letters delivered on time by old-fashioned means are increasingly rare, so it is as well that that is not the only option available.

Table 5: The sentences making up the test set.