Dependency Parser Evaluation: L95 Coursework Report

Danyi (Oli) Liu
Christ's College
dl567@cam.ac.uk

Word count: 4679

1 Introduction

This report presents a comparison of extracting dependency relationships using three different parsers: Stanford neural parser (Chen and Manning, 2014), Stanford shift-reduce constituency parser, and Berkeley neural constituency parser (Kitaev and Klein, 2018)[1]. The report starts by briefly introducing how each parser works as well as the pre-processing they require. We will then compare and contrast the parsing results given by these parsers against the partial gold standard grammatical relations of selected sentences given in the coursework handout. Further evaluation of the parsers utilising selected sentences from the *Deep Dependencies (DeepDep)* dataset (Bender et al., 2011) were also described. The report concludes with a general conclusion of the relative merits and drawbacks of the three parsers.

2 Selected Parsers

Stanford neural parser generates dependency relations directly using an arc-standard transition-based system. This seminal parser was the first to replace feature templates with dense vectors for words, part-of-speech (POS) tags, and arc labels, which are used to represent current configuration and predict the next transition.

Stanford shift-reduce parser yields phrase structure (PS) constituency parse using a feature-based shift-reduce system.

Berkeley neural parser used a different formalism, span-based parsing, to generate PS parses. The central idea is to assign a score to (label, span) pair, before performing bottom-up decoding with a CKY-style algorithm. The score for each span is computed with a self-attentive encoder, while LSTM encoders are commonly used in related work. The input to the encoder includes word embeddings, POS tags, and token locations.

[1] relevant input & output files available here

3 Evaluation Setup

Pre-processing All three parsers require as input POS tags and tokenised words, and take in sentences with boundaries marked. The two Stanford parsers use lemmatised word tokens while the original word form is maintained at the input to the Berkeley neural parser, which can be further compounded with sub-word tokens.

Post-processing To convert PS trees into dependency parses, we used the functionality provided by the Stanford CoreNLP suite (De Marneffe et al., 2006; Manning et al., 2014), which essentially performs rule-based dependency extraction and typing for the input PS tree.

Annotation The grammatical representation scheme we used for all three parsers is the Stanford typed dependencies. In collaboration with William Liu (yl535), we created a partial gold-standard annotations for the first 14 sentences given in the coursework handout (`gs.conll`). Since the Stanford parsers tokenise clitics and hyphens slightly different from the Berkeley parser, we edited the gold-standard annotations accordingly to accommodate these discrepancies (see `gs_stan_eval.conll`, `gs_berk_eval.conll`).

4 Evaluation on Sentences from Handout

4.1 Short Sentences

4.1.1 Quantitative Evaluation

We utilise the `MaltEval` software (Nilsson and Nivre, 2008) to quantitatively measure the performance of the parsers. However, because extracting

gold-standard annotations and adapting the tokenisation to suit each parser is very time-consuming. Thus we only used 14 sentences consisting of 240 tokens, covering 35 types of dependency relations, with 11 types among these occurring less than three times. This means that it is highly likely that the aggregated statistics shown below is not statistically significant enough reflect the actual performance of the parsers.

In fact, it is arguable how well quantitative results obtained on any existing evaluation set represents the efficacy of the parser. A higher labeled attachment score (LAS) or unlabeled attachment score (UAS) does not necessarily translate to better performance of the parser when working with unseen words, disparate domains, or constructions that account for a small proportion of dependencies in sentences but are nevertheless crucial for downstream applications. Aggregating precision and recall by dependency types and studying the confusion matrix would shed more light on the kinds of constructions that each parser works well or struggles with. But to more closely understand obtained and identify potential improvements, a qualitative analysis of the parser would be necessary.

With all this in mind, we carried out a quantitative analysis for potential insights and also to familiarise with the commonly-used measures that would be more representative when taken to a larger evaluation set.

metric		SN	SR	BN
token	LAS	**0.811**	0.762	0.772
	UAS	0.852	0.803	**0.888**
sentence	LAS	0.071	**0.143**	0.000
	UAS	0.143	0.214	**0.286**

Table 1: token- and sentence-level attachment score, with that of sentence-level counting labeled/unlabeled exact match between labeled and gold-standard sentences (SN: Stanford neural parser, SR: Stanford shift-reduce parser, BN: Berkeley neural parser)

As discussed before, the attachment scores shown in Table 1 are largely inconclusive, although we did note that the UAS obtained by the Berkeley neural parser is consistently higher than the other two on both token- and sentence- levels.

In order to estimate the parsers' capability in capturing various constructions, we group arcs by their gold-standard label type and show the F1 score of selected dependencies types in the table below. A dependency was selected if it occurred more than five times in our mini-corpus. We left out any dependencies with lower dependencies because any differences in the F1 score would be too likely to be due to pure chance.

type	count	SN	SR	BN
nsubj	23	0.96	0.98	0.98
punct	21	0.96	0.96	0.98
prep	18	0.95	0.90	1.00
pobj	18	0.90	0.84	0.97
det	16	0.97	0.97	0.97
ROOT	14	0.86	0.86	0.86
dobj	14	0.67	0.60	0.67
advmod	12	0.80	0.76	0.64
cc	9	0.95	0.95	0.95
conj	8	0.96	0.86	0.80
dep	7	0.29	0.33	0.25
rcmod	7	1.00	0.92	1.00
poss	6	1.00	1.00	1.00
cop	6	0.93	0.86	0.86
...	$\leq$5	...	...	...
macro-avg	-	0.85	0.76	0.86

Table 2: F1 score by dependency type

From the table above, we can see that common dependency types that all three parsers struggled the most with are: `dep` and `dobj`. The dependency type `dep` being difficult to recover is not a surprise, since it is used as a wildcard for relations that do not belong to other categories. Having taken a closer look at the wrong labels, we found that `dobj` had low F1-scores because it was constantly mis-predicted where ref should be the correct label. In the Stanford dependencies scheme, `ref` is stipulated to represent the complementiser introducing the relative clause (RC) modifying the NP. Assuming we remove this relation and label all such uses of complementiser with `dobj`, the relationship between the relative clause and the noun phrase it modifies can still be readily recovered as long as the head word of the RC is correcly labeled as `rcmod`. So it is arguable whether we can get rid of the `ref` label altogether. On the other hand, relative clauses introduced by a complementiser is a rather fixed construction and thus the `ref` labels are expected to be easily recovered, particularly since the sentences in our mini-corpus here are reasonably short and simple. Thus we suspect that the confusion between `ref` and `dobj` labels was caused by the rules for converting phrase structure parses to dependency parses not considering

the former, although a definite conclusion would require a closer investigation into the conversion rules.

We also present the F1 scores aggregated by relation lengths. As one would expect, the longer the dependencies spans, the harder it was to recover it. However, as before, the parsers encountered very few long dependencies, which put this conclusion in high uncertainties.

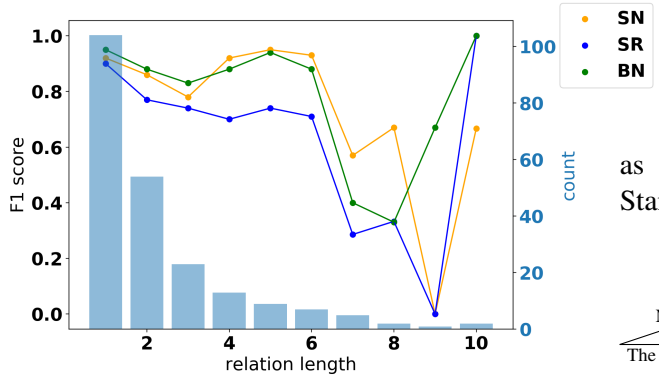


Figure 1: F1 score and count by relation length

4.1.2 Qualitative Evaluation

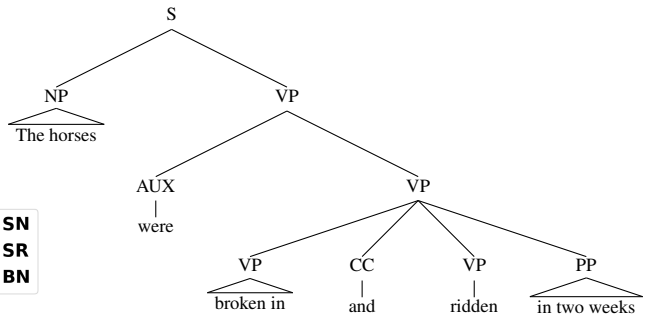
In this section, we provide a more detailed analysis of the (mis)behaviour of the parsers when tested with sentences given in the coursework handout.

(2) At least two men broke in and stole my TV.

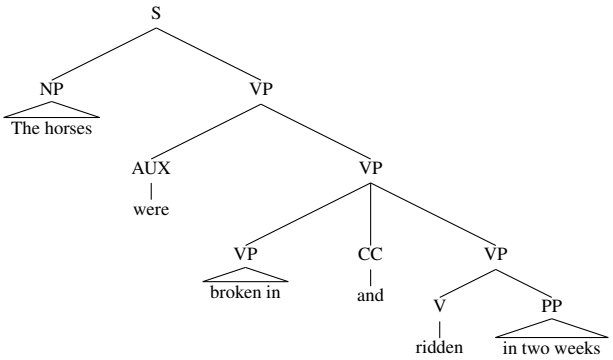
Only the Berkeley parser was able to recover the multiword label for "*at least*" and that the phrase is a quantifier phrase modifier of the the word "*two*", while both Stanford parsers treated it as a prepositional phrase. This could be because that the Berkeley parser uses a contextual encoder to represent the input sentence and is therefore better at capturing interactions and correlations between neighbouring word, which is important for identification of multi-word expressions (MWE). However, in the same sentence, the Berkeley parser failed to recognise that "*broke in*" was a phrasal verb and predicted `in` as an adverb modifying "*broke*". This calls into suspicion the previous argument that the contextual nature of the Berkeley parser makes it better at recognising MWE, since both Stanford parsers identified "*in*" as a particle.

(3) The horses were broken in and ridden in two weeks.

The Berkeley parser mistook "*ridden*" to be the auxiliary verb for "*broken*" in this sentence that both Stanford parsers got correct. Let us go back a step and look at the original constituency parse given by the Berkeley parser



as compared to the PS tree given by the Stanford shift-reduce parser shown below.



It is likely that this unconventional quaternary expansion of the VP was not taken into consideration when designing the rules applied to convert the parse to typed dependencies. The reason that the Berkeley parser gives non-binary expansions, which is not seen in traditional PTB style constituency trees (with the exception of the conjunction rule), is due to the introduction of an auxiliary empty label designed to handle spans that are not constituents. This allows greater flexibility in parsing complex or potentially un-stipulated constructions, but might cause confusion for the conversion system. As a result, evaluating the Berkeley parser on dependency relations does not do justice to the full efficacy of the parser.

(5) The horse which Kim sometimes rides is more bad tempered than mine.

This is a sentence where the Berkeley parser proved advantageous as compared to the Stanford parsers, both of which took "*bad*" to be the root of the sentence. All three parsers mistakenly labeled "*tempered*" to be a verb and bad as an adjective. The removal of an explicit grammar from the Berkeley neural parser is likely to have

made possible combining these two words to form a constituent.

(6) The horse as well as the rabbits which we wanted to eat have escaped.
The Stanford neural parser recognised "*horse*" to be the head of this sentence. The other two parsers showed potential benefit of generating dependency relations from constituent parses in this case, where the main verb and the subject was intercepted by a phrase modified by a RC. Also, the Berkeley parser was again the only parser that recognised "*as well as*" as a MWE. Another point to make with this sentence is that the main verb was linked to the latter conjunct, "*i.e. rabbit*", as in some other cases (sentence 3, 4). However, as stipulated in the Stanford dependency scheme, they should be directly connected to the head of the conjunction phrase, which is the head of the first conjunct in the phrase. Connecting the main verb (or prepositional phrase, in the case of sentence 3) to the closer conjunct is reasonable since it is ultimately linked to the first conjunct. However, this would require more care in designing the mapping between dependency parses to semantic representations, since we cannot assume all related constituents have been directly linked to the first conjunct as suggested by the dependency scheme.

(7) It was my aunt's car which we sold at auction last year in February.
This sentence revealed the inadequacy of the shift-reduce parser since it misjudged "*sold*" to be the `ccomp` (clausal complement) of the the sentence head "*car*". Nouns that have clausal complement include "fact" and "report", but definitely not "*car*". Although the shift-reduce parser is lexicalised, its representations for words is expected to have lower expressive power as compared to the dense representations used in the other two neural parsers, which often encode more information about the relationship between different words. An additional factor that might have confused the shift-reduce parser was that the subject of the sentence is a pronoun, which is often the case with sentences containing clausal complements ("*e.g. I admire the fact that you are honest*"). The parser failed to realise that "*it*" is expletive here. The fact that the Berkeley parser yielded the correct parse indicated that the constituency-to-dependency conversion software does not lack the mechanism to identify expletive "*it*".

(11) Letters delivered on time by old-fashioned means are increasingly rare, so it is as well that that is not the only option available.
The difficulties posed in this sentence include the past participle acting as a modifier ("*delivered*") and the two contiguous "*that*". Both the shift-reduce parser and the Berkeley neural parser were able to recover the first dependency, with the Berkeley parser assigning the more accurate label `vmod`. However, only the Stanford neural parser recovered the second half of the sentence faithfully. It recognised that "*well*" is the head of the latter half of the sentence that is in parataxis with the first half, and that "*option*" is the head of the complement clause with the first "*that*" as the marker and the second as subject pronoun. The shift-reduce parser, on the other hand, treated the second half as an adverbial clause, presumably due to conflation of "*is*" with "*being*", since the latter is indeed often featured in adverbial clauses. The Berkeley neural parser, despite recognising the parataxis between the sentences, identified "*is*" to be the head of the second half and treated "*as well*" as an adverb. This error could potentially be attributed to the parsers' over-generalisation of MWE.

(13) The long and lonely road to redemption begins with self-reflection: the need to delve inwards to deconstruct layers of psychological obfuscation.
This is another sentence that showed the advantage of using the Berkeley neural parser. It was the only one among the three tested to locate the head of the phrase that follows the colon, although the wildcard label `dep` was given instead of the more appropriate `npadvmod`. It remains to be explored whether the constituency-to-dependency conversion scheme would be the one to blame here. Moreover, it correctly recovered the dependency between "*deconstruct*" and "*delve*", giving the accurate label of `vmod`. The two Stanford parsers gave similar wrong analyses for this sentence, treating the NP adverb as a parataxis. The key source of error was that they both labeled "*deconstruct*" as a noun. Thus, to explain the adverbial NP, the shift-reduce parser assigned "*need*" to be the object of "*begins*" while the

neural parser labeled it to be the subject of the second sentence in parataxis. The expressive, broad-coverage encoder of the Berkeley parser proved to be crucial in this example.

4.2 Long Sentences

(16) Making these decisions requires sophisticated knowledge of syntax; tagging manuals (Santorini, 1990) give various heuristics that can help human coders make these decisions and that can also provide useful features for automatic taggers.
The major challenge in analysing this sentence lies in the conjunction of the two relative clauses modifying *heuristics*" that are marked by two separate *that*". In addition, there is a parataxis between two parts of the sentences delimited by the semicolon. The shift-reduce parser was the only one that did not capture this latter parataxis. Instead, it conflated all the words up to ... *(Santorini, 1990)*" as the clausal subject of *give*". However, when we removed the parenthetical *(Santorini, 1990)*", the shift-reduce did pick up the parataxis. With regard to the conjunction, both Stanford parsers falsely conjuncted "*provide*" with "*give*", while the Berkeley parser accurately linked "*provide*" to "*help*", which could also be recovered by the Stanford parsers When the second "*that*" was removed. This shows that the Stanford parsers do not accept conjunction of two RCs since removing the second "*that*" would turn it into a conjunction of verb phrases. Therefore, this sentence showed again the greater flexibility of the grammar allowed by the Berkeley parser.

(17) The Penn Treebank tagset was culled from the original 87-tag tagset for the Brown Corpus. For example the original Brown and C5 tagsets include a separate tag for each of the different forms of the verbs do (e.g. C5 tag VDD for did and VDG tag for doing), be and have.
This sentence proved difficult for all three parsers because the words "*do, did, doing, be, have*" here are in fact "meta-language", referring to the words themselves, rather than acting as the auxiliary ("*do, have*"), copula ("*be*"), or verb predicate ("*do, did, doing, have*"), as they usually do. These are all common words that have a relatively fixed usage, and worse of all, they are all verbs of some type, which made it extra hard to interpret their roles in this sentence, which are closest to proper nouns. As a result, all three parsers generated parses that were difficult to make sense of at all, whether we look at the dependency relations or the parse tree. Removing the parenthetical could not change this for any better.

(19) Thus the EM-trained "pure HMM" tagger is probably best suited to cases where no training data is available, for example, when tagging languages for which no data was previously hand-tagged.
In analysing the first half of this sentence, the Stanford neural parser and the Berkeley neural parser correctly identified the clause "... *where no training is available* ..." as a RC modifying "*cases*" while the shift-reduce parser decided that it was a adverbial clause elaborating on the whole sentence. This is understandable since "*where*" is an adverb, and the shift-reduce parser is likely to exploit POS tags to a greater extent as subcategorisation information as compared to the neural parsers. As in sentence (7), this example illustrated once again that the lexical representation of the feature-based shift-reduce parser is unsatisfactory. The second half of the sentence is a adverbial clause containing a RC. To further complicate the matter, the matrix clause is in a reduced form (the full form could be "*when you are tagging languages for which ...*"). In this case, the shift-reduce parser identified both clauses ("*tagging*" as the head of the adverbial clause and related to the RC before "*for example*"; "*(hand-)tagged*" as the head of the enclosed RC with "*languages*" as its head). While the Berkeley parser captured the RC, it failed to recognising the enclosing adverbial clause, identifying "*languages*" to be the head of the segment, linked to the earlier RC via the wildcard dependency `dep`. Finally, the Stanford neural parser did not identify the enclosed RC but treated the entire segment as an adverbial clause modifying the earlier RC, with "*(hand-)tagged*" being the head of the adverbial clause. The reason for the success of the shift-reduce parser in this case can probably be attributed to it having a stipulated grammar as its backbone. The transition-based Stanford neural parser also include considerations for grammar by only selecting from certain allowed transitions, but the role of the grammar is less significant and explicit than that in the shift-reduce constituency parser. As a result, the parse given by the Stanford neural parser deviated from the gold standard to a smaller extent than the Berkeley neural parser,

which does not include any explicit grammar production rules.

(22) An MoD spokesman said: "Surveys of Astute have now been completed and she will proceed to Faslane under her own power. She is being escorted by tugs and HMS Shoreham."
All three parsers picked up that the content within the quotation marks are the clausal complement (`ccomp`) of the word "*said*". However, the POS-tagger within the Berkeley neural parser pipeline mistook the named-entity "*Faslane*" to be a verb, leading to confusion analysis of "*she will proceed to Faslane*", reflecting the limited robustness of the parser's encoder.

(24) Instead of constantly worrying about funding, the faculty and students can focus on their project, with the exception of sponsors' weeks, when they have to convince companies to start or continue their support.
All three parsers captured that "*instead of constantly worrying about funding*" was a prepositional phrase linked to "*focus*". However, the shift-reduce parser included "*the faculty and students*" into the prep phrase, leading to confusion in the later part of the sentence. Even though "*Instead of constantly worrying about funding, the faculty and students*" is a perfectly valid construct, this would leave the main part of the sentence without a subject. Since the shift-reduce parser explicitly stipulates a grammar, a reasonable deduction would be that the construct of "prep phrase + main sentence" was not included in the grammar, rendering the parser frail against this relatively non-conventional constructions.

5 Evaluation on Deep Dependencies

5.1 Test Sentences

The shortcomings of aggregated parsing metrics are readily recognised in the research community, and have prompted work that examine parser targeted performances in extracting unbounded dependencies (Rimell et al., 2009; Nivre et al., 2010; Bender et al., 2011). These are some of the most error-prone constructions that are often useful for downstream applications like question answering. Having searched for the dataset mentioned in Rimell et al. (2009) to no avail, we made use of the *DeepDep* dataset (Bender et al., 2011) to investigate how well the considered parsers tackle their selected constructions. This dataset consists of 1000 sentences demonstrating 10 types of linguistic phenomenon. These sentences were extracted from Wikipedia and each was annotated with up to two dependencies reflecting the core property of one phenomenon of interest. Recall of these dependencies is used as the only metric for performance. Recall on its own is a valid measure here because in the basic Stanford dependency scheme that we used, each word has only one root and the tree structure is preserved. Thus there is no danger of abusing the metric which can be achieved by, *e.g.* returning all possible word pairs with all possible labels.

Unfortunately we were unable to produce dependency label files that were compatible with the evaluation scripts provided in the dataset. Thus, instead of computing the aggregated results across the whole dataset, we selected three sentences to represent each type of phenomenon and manually compared the labels given by the parsers against the gold standard annotations. When selecting test sentences, we tried to choose one short sentence (less than 20 words), one long sentence, and one that contains more than two named entities for each construction when possible.

5.2 Error Analysis

Among the ten types of dependencies, all three parsers correctly recovered the following constructions in all selected sentences: **'control'** construction, **'right node rising'** (conjunction of non-constituents), **tough adjectives**, **verbal gerunds**, and **verb-participle**. While the Berkeley neural parser also accurately identified the **'finite *that*-less relative clauses (barerel)'** and **'interleaved argument and adjunct'** constructions, the Stanford parsers, the neural parser in particular, struggled more with these two types of constructions.
For the following **'barerel'** sentence

This season marked the last the packers would play under legendary coach curly Lambeau.

only the Berkeley parser linked "*play*" to "*last*" with dependency typed as `rcmod`. The shift-reduce parser incorrectly identified the predicate "*marked*" as the head of "*play*" with wildcard dependency type `dep`. The Stanford neural parser was much more hopeless, marking "*play*" as the main predicate while "*this season marked*" was taken to be a clausal subject.

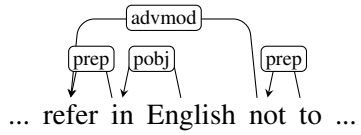
In fact, the Stanford neural parser got the other

two sentences corresponding to the 'barerel' construction wrong as well. In both cases, it identified the relationship between the RC and the matrix clause as parataxis. This shared pattern of mistake was indicative but not yet conclusive of the parser's inadequacy in recognising *that*-less RC.

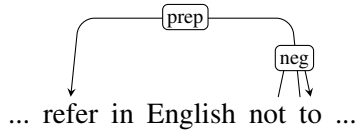
In the case of **'interleaved argument and adjunct'**, two out of three sentences were problematic for the Stanford neural parser.

(a) ... these words refer in English not to brotherly love but to sexual attraction
and

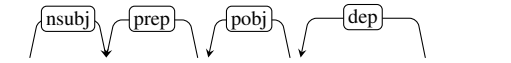
(b) In 1924, in milan he establishes with Morzenti his own publishing house, the Mauzan Morzenti agency.
For (a), the Stanford neural produced the following dependencies:



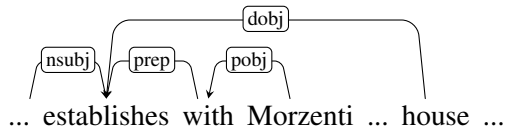
while the other two parsers recovered the gold standard dependencies:



For (b), the Stanford parser extracted:



in contrast to:



These errors revealed an additional inadequacy in the grammar of the Stanford neural parser.

DeepDep includes two types of phrasal modifiers: **'ned'** (which refers to a noun taking the verbal *-ed* ending and modified another noun or adjective, *e.g.* "wide eyed"), and **'absolutive'** (a non-finite verb following an NP, modifying the predicate of the NP).

The **'ned'** phrases were accurately identified as `amod` by all three parsers when they function like adjectives that modify nouns. However, the dependency was harder to recover when the phrase follows and modifies a verb, as in the case of following sentence:

... mob members who answered that question incorrectly are eliminated from the game, leaving empty handed.

The Stanford neural parser identified "*(empty-) handed*" to be the object (`dobj`) of "*leaving*" while the other two parsers took it to be the adjectival complement `acomp`, which was slightly closer to the gold-standard label `advmod`.

When it comes to **'absolutive'**, all three parsers gave poor results. There was only one correct label given by the Stanford neural parser to one of the sentences, connecting the non-finite verb to the predicate verb. In all other cases, the non-finite verb was linked to the NP that preceded the modifier, either directly with the wildcard label `dep`, or mediated via the subject of the modifier clause, which was linked to the preceding NP with label `appos`.

One final construction of interest is a non-dependency, *i.e.* the **expletive *it***. Although the Stanford scheme reserved the label `expl` for expletives, it seems that this relation is only used to represent existential *there*. On the other hand, the expletive *it* occurs in more varied constructions with more unpredictable functions. For instance, in the three sentences under consideration, the uses were: *it turned out that...*, *it is suggested that...* and *it costs them....* In fact, although the *it* in these constructions are devoid of meaning on their own, it would be difficult to stipulate a fixed way of representing the dependency role of these expletive *it*-s, since they fit into different constructions in different ways. In fact, such stipulation might not be necessary if our end goal is to extract semantic representation, since we could use post-processing techniques to remove expletive 'this' before retrieving the semantic representation as long as the predicate and its related arguments are preserved. For these reasons, while Bender et al. (2011) counted as incorrect any dependency involving referential *it*, we think this issue is more relevant to the design of dependency scheme rather than the efficacy of the parser and thus omitted it from our evaluation.

Note on Punctuations The original sentences from the *DeepDep* dataset contained no punctuations. The above analyses were performed on these 30 sentences with manually added-in punctuations. In all but a few cases, the presence of punctuations made no difference to the analyses.

The exception was with analysing **interleaved argument and adjunct** sentences, where removing punctuations allowed the Stanford neural parser to correct the parse for one sentence, but confused the shift-reduce parser to give a different and wrong analysis for another.

6 Conclusions

The Stanford neural parser is a strong performer on the English Penn Treebank (Chen and Manning, 2014). It is a combination of expressive dense lexical representation and stipulated grammar structure. Moreover, it is directly optimised for dependency parsing when the quality of the grammatical relations extracted from the other two parsers are dependent on the conversion program for constituency-to-dependency. However, upon qualitative analysis on its parses for *DeepDep* test sentences, we discover that it struggles with constructions like finite *that*-less relative clause and interleaved arguments & adjuncts, revealing potential loopholes in the grammatical restrictions applied in choosing choosing state transitions.

The Stanford shift-reduce parser, on the other hand, uses feature-based representations and thus sometimes lacks the lexical semantic information to produce correct dependencies where multiple constructions fit the segment on the pure syntactic level. The parser has the most explicitly stipulated grammar among the three, which might be instrumental in recognising certain constructions that are statistically uncommon yet covered in its grammar (recall sentence 19 from section 4.2).

Finally, the Berkeley neural parser uses the most complex and powerful model for encoding the dynamics between lexical items and their corresponding dependency labels. This expressiveness allows it to more accurately categorise infrequent words (recall *deconstruct* in sentence 13) and could potentially explain its higher sensitivity with multiword expressions. However, this expressiveness also backfires when it makes a confident but wrong prediction about an unknown word (*Faslane* in sentence 22). The parser also has a higher flexibility in terms of the grammatical structure it allows. In fact, there is no grammar rules stipulated explicitly but only exploits the statistical dynamics between the words to induce structure. This flexibility allowed it to capture certain atypical constructions like the conjunction of non-constituents or clauses, but again backfires when dealing with certain constructions that are stipulated in the grammar used by another parser but probably not statistically significant enough to be captured by this grammar-less parser.

A key difference between transition- and chart-based grammar is that the search algorithm that they use, which has implications for both efficiency and optimality. The transition-based parser often uses greedy search and is prone to error propagation, while the CKY algorithm used for chart-based parsing often has higher complexity. However, in our experiment, the Berkeley parser was faster than the other two by a large margin, although this is hardly conclusive with the execution efficiency depending on the actual implementation.

parser	SN	SR	BN
time (s)	278.3	355.9	155.4

Table 3: Time taken by the three parsers to parse the selected *DeepDep* sentences

References

Emily M. Bender, Dan Flickinger, Stephan Oepen, and Yi Zhang. 2011. Parser Evaluation over Local and Non-Local Deep Dependencies in a Large Corpus. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 397–408, Edinburgh, Scotland, UK. Association for Computational Linguistics.

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

Marie-Catherine De Marneffe, Bill MacCartney, Christopher D Manning, et al. 2006. Generating typed dependency parses from phrase structure parses.

Nikita Kitaev and Dan Klein. 2018. Constituency Parsing with a Self-Attentive Encoder. *arXiv:1805.01052 [cs]*.

Christopher D Manning, Mihai Surdeanu, John Bauer, Jenny Rose Finkel, Steven Bethard, and David McClosky. 2014. The stanford corenlp natural language processing toolkit. In *Proceedings of 52nd annual meeting of the association for computational linguistics: system demonstrations*, pages 55–60.

Jens Nilsson and Joakim Nivre. 2008. Malteval: an evaluation and visualization tool for dependency parsing.

Joakim Nivre, Laura Rimell, Ryan McDonald, and Carlos Gómez-Rodríguez. 2010. Evaluation of Dependency Parsers on Unbounded Dependencies. In *Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010)*, pages 833–841, Beijing, China. Coling 2010 Organizing Committee.

Laura Rimell, Stephen Clark, and Mark Steedman. 2009. Unbounded Dependency Recovery for Parser Evaluation. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*, pages 813–821, Singapore. Association for Computational Linguistics.