Human-in-the-Loop Parsing

Luheng He **Julian Michael** **Mike Lewis** **Luke Zettlemoyer**
Computer Science & Engineering
University of Washington
Seattle, WA
{luheng,julianjm,mlewis,lsz}@cs.washington.edu

Abstract

This paper demonstrates that it is possible for a parser to improve its performance with a human in the loop, by posing simple questions to non-experts. For example, given the first sentence of this abstract, if the parser is uncertain about the subject of the verb "pose," it could generate the question *What would pose something?* with candidate answers *this paper* and *a parser*. Any fluent speaker can answer this question, and the correct answer resolves the original uncertainty. We apply the approach to a CCG parser, converting uncertain attachment decisions into natural language questions about the arguments of verbs. Experiments show that crowd workers can answer these questions quickly, accurately and cheaply. Our human-in-the-loop parser improves on the state of the art with less than 2 questions per sentence on average, with a gain of 1.7 F1 on the 10% of sentences whose parses are changed.

1 Introduction

The size of labelled datasets has long been recognized as a bottleneck in the performance of natural language processing systems (Marcus et al., 1993; Petrov and McDonald, 2012). Such datasets are expensive to create, requiring expert linguists and extensive annotation guidelines. Even relatively large datasets, such as the Penn Treebank, are much smaller than required—as demonstrated by improvements from semi-supervised learning (Søgaard and Rishøj, 2010; Weiss et al., 2015).

We take a step towards cheap, reliable annotations by introducing human-in-the-loop parsing, where non-experts improve parsing accuracy by answering questions automatically generated from the parser's output. We develop the approach for CCG parsing, leveraging the link between CCG syntax and semantics to convert uncertain attachment decisions into natural language questions. The answers are used as soft constraints when re-parsing the sentence.

Temple also said Sea Containers' plan raises numerous legal, regulatory, financial and fairness issues, but didn't ***elaborate***.	
Q:	What didn't ***elaborate***?
[1] ****	Temple
[2] *	Sea Containers' plan
[3]	None of the above.

Table 1: An automatically generated query from CCGbank. 4 out of 5 annotators correctly answered *Temple*, providing a signal that can be used to improve parse predictions.

Previous work used crowdsourcing for less structured tasks such as named entity recognition (Werling et al., 2015) and prepositional phrase attachment (Jha et al., 2010). Our work is most related to that of Duan et al. (2016), which automatically generates paraphrases from n-best parses and gained significant improvement by re-training from crowdsourced judgments on two out-of-domain datasets. Choe and McClosky (2015) improve a parser by creating paraphrases of sentences, and then parsing the sentence and its paraphrase jointly. Instead of using paraphrases, we build on the approach of QA-SRL (He et al., 2015), which shows that untrained crowd workers can annotate predicate–argument structures by writing question–answer pairs.

Our experiments for newswire and biomedical

Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, pages 2337–2342,
Austin, Texas, November 1-5, 2016.

text demonstrate improvements to parsing accuracy of 1.7 F1 on the sentences changed by re-parsing, while asking only less than 2 questions per sentence. The annotations we collected[1] are a representation-independent resource that could be used to develop new models or human-in-the-loop algorithms for related tasks, including semantic role labeling and syntactic parsing with other formalisms.

2 Mapping CCG Parses to Queries

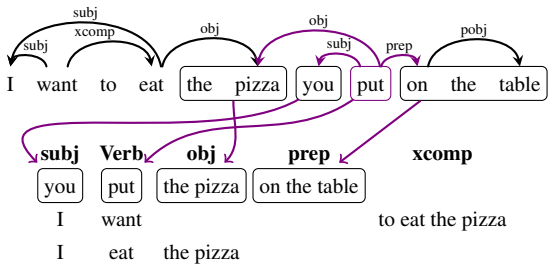
Our annotation task consists of multiple-choice *what*-questions that admit multiple answers. To generate them, we produce question–answer (QA) pairs from each parse in the 100-best scored output of a CCG parser and aggregate the results together.

We designed the approach to generate queries with high **question confidence**—questions should be simple and grammatical, so annotators are more likely to answer them correctly—and high **answer uncertainty**—the parser should be uncertain about the answers, so there is potential for improvement.

Our questions only apply to core arguments of verbs where the argument phrase is an NP, which account for many of the parser's mistakes. Prepositional phrase attachment mistakes are also a large source of errors—we tried several approaches to generate questions for these, but the greater ambiguity and inconsistency among both annotators and the gold parses made it difficult to extract meaningful signal from the crowd.

Generating Question–Answer Pairs Figure 1 shows how we generate QA pairs. Each QA pair corresponds to a dependency such that if the answer is correct, it indicates that the dependency is in the correct parse. We determine a verb's set of arguments by the CCG supertag assigned to it in the parse (see Steedman (2000) for an introduction to CCG). For example, in Figure 1 the word *put* takes the category ((S\NP)/PP)/NP (not shown), indicating that it has a subject, a prepositional phrase argument, and an object. CCG parsing assigns dependencies to each argument position, even when the arguments are reordered (as with *put* → *pizza*) or span long distances (as with *eat* → *I*).

[1]Our code and data are available at `https://github.com/luheng/hitl_parsing`.



Dependency	Question	Answer
want → I	What wants to eat something?	I
eat → I	What would eat something?	I
eat → pizza	What would something eat?	the pizza
put → you	What put something?	you
put → pizza	What did something put?	the pizza
on → table	What did something put something on?	the table

Figure 1: From a labeled dependency graph, we extract phrases corresponding to every argument of every verb using simple heuristics. We then create questions about dependencies, adding a *would* modal to untensed verbs and placing arguments to the left or right of the verb based on its CCG category. We only generate QA pairs for *subj*, *obj*, and *pobj* dependencies. To identify multiple answer options, we create QA pairs from all parses in the 100-best list and pool equivalent questions with different answers. See Table 2 for example queries.

To reduce the chance of parse errors causing nonsensical questions (for example, *What did the pizza put something on?*), we replace all noun phrases with *something* and delete unnecessary prepositional phrases. The exception to this is with copular predicates, where we include the span of the argument in the question (see Example 4 in Table 2).

Grouping QA Pairs into Queries After generating QA pairs for every parse in the 100-best output of the parser, we pool the QA pairs by the head of the dependency used to generate them, its CCG category, and their question strings. We also compute marginalized scores for each question and answer phrase by summing over the scores of all the parses that generated them. Each pool becomes a query, and for each unique dependency used to generate QA pairs in that pool, we add a candidate answer to the query by choosing the answer phrase that has the highest marginalized score for that dependency. For example, if some parses generated the answer phrase *pizza* for the dependency *eat* → *pizza*, but most of the high-scoring parses generated the answer phrase *the pizza*, then only *the pizza* appears as an answer.

Sentence	Question	Votes	Answers
(1) Structural Dynamics Research Corp. . . . said it introduced new technology in mechanical design automation that will ***improve*** mechanical engineering productivity.	What will ***improve*** something?	0	Structural Dynamics Research Corp
		5	**new technology**
		0	mechanical design automation
(2) He said disciplinary proceedings are confidential and declined to ***comment*** on whether any are being held against Mr. Trudeau.	What would ***comment***?	5	**he**
		0	disciplinary proceedings
(3) To ***avoid*** these costs, and a possible default, immediate action is imperative.	What would something ***avoid***?	4	**these costs**
		3	**a possible default**
(4) The price is a new high for California Cabernet Sauvignon, but it ***is*** not the highest.	What ***is*** not the highest?	2	the price
		3	**it**
(5) Kalipharma is a New Jersey-based pharmaceuticals concern that ***sells*** products under the Purepac label.	What ***sells*** something?	5	Kalipharma
		0	**a New Jersey-based pharmaceuticals concern**
(6) Further, he said, the company doesn't have the capital needed to ***build*** the business over the next year or two.	What would ***build*** something?	4	the company
		1	**the capital**
(7) Timex had requested duty-free treatment for many types of watches, ***covered*** by 58 different U.S. tariff classifications.	What would be ***covered***?	0	Timex
		0	duty-free treatment
		2	**many types of watches**
		3	watches
(8) You either believe Seymour can do it again or you ***do*** n't .	What ***does***?	3	**you**
		0	Seymour
		2	None of the above

Table 2: Example annotations from the CCGbank development set. Answers that agree with the gold parse are in bold. The answer choice *None of the above* was present for all examples, but we only show it when it was chosen by annotators.

From the resulting queries, we filter out questions and answers whose marginalized scores are below a certain threshold and queries that only have one answer choice. This way we only ask confident questions with uncertain answer lists.

3 Crowdsourcing

We collected data on the crowdsourcing platform CrowdFlower.[2] Annotators were shown a sentence, a question, and a list of answer choices. Annotators could choose multiple answers, which was useful in case of coordination (see Example 3 in Table 2). There was also a *None of the above* option for when no answer was applicable or the question was nonsensical.

We instructed annotators to only choose options that *explicitly* and *directly* answer the question, to encourage their answers to closely mirror syntax. We also instructed them to ignore *who*/*what* and *someone*/*something* distinctions and overlook mistakes where the question was missing a negation. The instructions included 6 example queries with answers and explanations. We used CrowdFlower's quality control mechanism, displaying pre-annotated queries 20% of the time and requiring annotators to maintain high accuracy.

Dataset Statistics Table 3 shows how many sentences we asked questions for and the total number of queries annotated. We collected annotations for the development and test set for CCGbank (Hockenmaier and Steedman, 2007) as in-domain data and the test set of the Bioinfer corpus (Pyysalo et al., 2007) as out-of-domain. The CCGbank development set was used for building question generation heuristics and setting hyperparameters for reparsing.

5 annotators answered each query; on CCGbank we required 85% accuracy on test questions and on Bioinfer we set the threshold at 80% because of the difficulty of the sentences. Table 4 shows inter-annotator agreement. Annotators unanimously chose the same set of answers for over 40% of the queries; an absolute majority is achieved for over 90% of the queries.

[2] www.crowdflower.com

Dataset	Sentences	Covered	Queries	Q/S
CCG-Dev	1913	1155	1904	1.7
CCG-Test	2407	1460	2511	1.7
Bioinfer	500	360	680	1.9

Table 3: Sentence coverage, number of queries annotated, and average number of queries per sentence (Q/S).

k-Agreed	CCG-Dev	CCG-Test	Bioinfer
5	48.0%	40.2%	47.7%
$\geq$ **4**	76.6%	68.0%	75.0%
$\geq$ **3**	94.9%	91.5%	94.0%

Table 4: The percentage of queries with at least k annotators agreeing on the exact same set of answers.

Qualitative Analysis Table 2 shows example queries from the CCGbank development set. Examples 1 and 2 show that workers could annotate long-range dependencies and scoping decisions, which are challenging for existing parsers.

However, there are some cases where annotators disagree with the gold syntax, mostly involving semantic phenomena which are not reflected in the syntactic structure. Many cases involve co-reference, where annotators often prefer a proper noun referent over a pronoun or indefinite (see Examples 4 and 5), even if it is not the syntactic argument of the verb. Example 6 shows a complex control structure, where the gold CCGbank syntax does not recover the true agent of *build*. CCGbank also does not distinguish between subject and object control. For these cases, our method could be used to extend existing treebanks. Another common error case involved partitives and related constructions, where the correct attachment is subtle—as reflected by the annotators' split decision in Example 7.

Question Quality Table 5 shows the percentage of questions that are answered with *None of the above* (written *N/A* below) by at most k annotators. On all domains, about 80% of the queries are considered answerable by all 5 annotators. To have a better understanding of the quality of automatically generated questions, we did a manual analysis on 50 questions for sentences from the CCGbank development set that are marked *N/A* by more than one annotator. Among the 50 questions, 31 of them are either generated from an incorrect supertag or unanswerable given the candidates. So the *N/A* answer

k-N/A	CCG-Dev	CCG-Test	Bioinfer
0	77.6%	81.6%	79.3%
$\leq$ **1**	89.6%	92.6%	89.1%
$\leq$ **2**	93.8%	96.1%	92.8%

Table 5: The percentage of queries with at most k annotators choosing the *None of the above* (N/A) option.

can provide useful signal that the parses that generated the question are likely incorrect. Common mistakes in question generation include: bad argument span in a copula question (4 questions), bad modality/negation (3 questions), and missing argument or particle (5 questions). Example 8 in Table 2 shows an example of a nonsensical question. While the parses agreed with the gold category $\mathsf{S\backslash NP}$, the question they generated omitted the negation and the verb phrase that was elided in the original sentence. In this case, 3 out of 5 annotators were able to answer with the correct dependency, but such mistakes can make re-parsing more challenging.

Cost and Speed We paid 6 cents for each answer. With 5 judgments per query, 20% test questions, and CrowdFlower's 20% service fee, the average cost per query was about 46 cents. On average, we collected about 1000 judgments per hour, so we were able to annotate all the queries generated from the CCGbank test set within 15 hours.

4 Re-Parsing with QA Annotation

To improve the output of the parser, we re-parse each sentence with an augmented scoring function that penalizes parses for disagreeing with annotators' choices. If q is a question, a is an answer to q, d is the dependency that produced the QA pair $\langle q, a \rangle$, and $v(a)$ annotators chose a, we add re-parsing constraints as follows:

- If $v(\textit{None of the above}) \geq T^+$, penalize parses that agree with q's supertag on the verb by w^t
- If $v(a) \leq T^-$, penalize parses containing d by w^-
- If $v(a) \geq T^+$, penalize parses that do not contain d by w^+

where T^+, T^-, w^t, w^-, and w^+ are hyperparameters. We incorporate these penalties into the parsing model during decoding. By using soft constraints, we mitigate the risk of incorrect annotations worsening a high-confidence parse.

Data	L16	HITL
CCG-Dev	87.9	88.4
CCG-Test	88.1	88.3
Bioinfer	82.2	82.8

Table 6: CCG parsing accuracy with human in the loop (HITL) versus the state-of-the-art baseline (L16) in terms of labeled F1 score. For both in-domain and out-domain, we have a modest gain over the entire corpus.

Data	L16	HITL	Pct.
CCG-Dev	83.9	87.1	12%
CCG-Test	84.2	85.9	10%

Table 7: Improvements of CCG parsing accuracy on changed sentences for in-domain data. We achieved significant improvement over the 10%–12% (Pct.) sentences that were changed by re-parsing.

Some errors are predictable: for example, if a is a non-possessive pronoun and is closer to the verb than its referent a', annotators often choose a' when a is correct (See Example 4 in Table 2). If a is a subspan of another answer a' and their votes differ by at most one (See Example 7 in Table 2), it is unlikely that both a and a' are correct. In these cases we use disjunctive constraints, where the parse needs to have at least one of the desired dependencies.

Experimental Setup We use Lewis et al. (2016)'s state-of-the-art CCG parser for our baseline. We chose the following set of hyperparameters based on performance on development data (CCG-Dev): $w^+ = 2.0$, $w^- = 1.5$, $w^t = 1.0$, $T^+ = 3$, $T^- = 0$. In the Bioinfer dataset, we found during development that the pronoun/subspan heuristics were not as useful, so we did not use them in re-parsing.

Results Table 6 shows our end-to-end parsing results. The larger improvement on out-of-domain sentences shows the potential for using our method for domain adaptation. There is a much smaller improvement on test data than development data, which may be related to the lower annotator agreement reported in Table 4.

There was much larger improvement (1.7 F1) on the subset of sentences that are changed after re-parsing, as shown in Table 7. This suggests that our method could be effective for semi-supervised learning or re-training parsers. Overall improvements on CCGbank are modest, due to only modifying 10% of sentences.

5 Discussion and Future Work

We introduced a human-in-the-loop framework for automatically correcting certain parsing mistakes. Our method identifies attachment uncertainty for core arguments of verbs and automatically generates questions that can be answered by untrained annotators. These annotations improve performance, particularly on out-of-domain data, demonstrating for the first time that untrained annotators can improve state-of-the-art parsers.

Sentences modified by our framework show substantial improvements in accuracy, but only 10% of sentences are changed, limiting the effect on overall accuracy. This work is a first step towards a complete approach to human-in-the-loop parsing.

Future work will explore the possibility of asking questions about other types of parsing uncertainties, such as nominal and adjectival argument structure, and a more thorough treatment of prepositional-phrase attachment, including distinctions between arguments and adjuncts. We hope to scale these methods to large unlabelled corpora or other languages, to provide data for re-training parsers.

Acknowledgments

This work was supported by the NSF (IIS-1252835, IIS-1562364), DARPA under the DEFT program through the AFRL (FA8750-13-2-0019), an Allen Distinguished Investigator Award, and a gift from Google. We are grateful to Chloé Kiddon for helpful comments on the paper, and Kenton Lee for help with the CCG parser. We would also like to thank our workers on Crowdflower for their annotation and the anonymous reviewers for their valuable feedback.

References

Do Kook Choe and David McClosky. 2015. Parsing paraphrases with joint inference. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics*.

Manjuan Duan, Ethan Hill, and Michael White. 2016. Generating disambiguating paraphrases for struc-

turally ambiguous sentences. In *Proceedings of the 10th Linguistic Annotation Workshop*.

Luheng He, Mike Lewis, and Luke Zettlemoyer. 2015. Question-answer driven semantic role labeling: Using natural language to annotate natural language. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*.

Julia Hockenmaier and Mark Steedman. 2007. Ccgbank: a corpus of ccg derivations and dependency structures extracted from the penn treebank. *Computational Linguistics*.

Mukund Jha, Jacob Andreas, Kapil Thadani, Sara Rosenthal, and Kathleen McKeown. 2010. Corpus creation for new genres: A crowdsourced approach to pp attachment. In *Proceedings of the NAACL HLT 2010 Workshop on Creating Speech and Language Data with Amazon's Mechanical Turk*.

Mike Lewis, Kenton Lee, and Luke Zettlemoyer. 2016. Lstm ccg parsing. In *Proceedings of the Human Language Technology Conference of the North American Chapter of the Association of Computational Linguistics*.

Mitchell P Marcus, Mary Ann Marcinkiewicz, and Beatrice Santorini. 1993. Building a large annotated corpus of english: The penn treebank. *Computational Linguistics*.

Slav Petrov and Ryan McDonald. 2012. Overview of the 2012 Shared Task on Parsing the Web. Notes of the First Workshop on Syntactic Analysis of Non-Canonical Language (SANCL).

Sampo Pyysalo, Filip Ginter, Juho Heimonen, Jari Björne, Jorma Boberg, Jouni Järvinen, and Tapio Salakoski. 2007. Bioinfer: a corpus for information extraction in the biomedical domain. *BMC bioinformatics*.

Anders Søgaard and Christian Rishøj. 2010. Semi-supervised dependency parsing using generalized tri-training. In *Proceedings of the 23rd International Conference on Computational Linguistics*.

Mark Steedman. 2000. *The syntactic process*.

David Weiss, Chris Alberti, Michael Collins, and Slav Petrov. 2015. Structured training for neural network transition-based parsing. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing*.

Keenon Werling, Arun Tejasvi Chaganty, Percy S Liang, and Christopher D Manning. 2015. On-the-job learning with bayesian decision theory. In *Advances in Neural Information Processing Systems*.