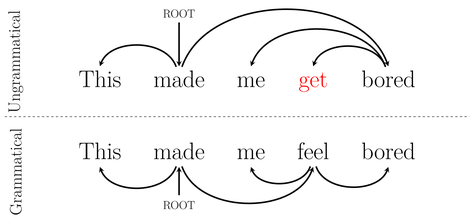
An Evaluation of Parser Robustness for Ungrammatical Sentences

Homa B. Hashemi, Rebecca Hwa

Intelligent Systems Program, Computer Science Department, University of Pittsburgh

Parsing Ungrammatical Sentences

Performance of parsers degrades on sentences that have even small grammatical errors:



Robust Parser

If the parser can overlook problems such as grammar mistakes and produce a parse tree that closely resembles the correct analysis for the intended sentence, we say that the parser is robust.

Questions

- Are some parsers more robust than others against sentences that are not well-formed?
- In what ways does a parser's performance degrade when dealing with ungrammatical sentences?

Evaluation of Parser Robustness

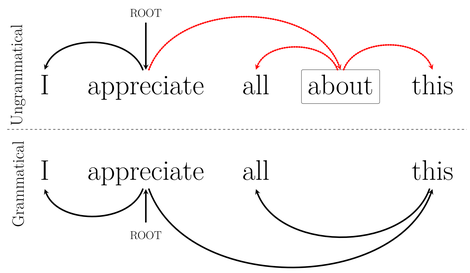
1. Manually annotated gold standards
   - Ungrammatical treebank is not available for all domains
   - Creating a treebank is expensive and time-consuming
2. **Gold standard free** approach
   - Compare parse tree of problematic sentence against parse tree of well-formed sentence as **gold standard**
   - We cannot use standard metrics of comparing trees, because
     - Words of ungrammatical sentence and its grammatical counterpart do not necessarily match
   - We do not want to unfairly penalize parsers when there are extra or missing words

ESL Sentence: *I appreciate all **about** this.*

Corrected ESL Sentence: *I appreciate all this.*

Proposed Evaluation Methodology

- **Error-related dependency**: dependency connected to an extra word
- **Shared dependency**: mutual dependency between two trees



$$Precision = \frac{\#\text{ of shared dependencies}}{\#\text{ of dependencies - }\#\text{ of error-related dependencies of ungrammatical sentence}} = \frac{2}{5-3} = 1$$

$$Recall = \frac{\#\text{ of shared dependencies}}{\#\text{ of dependencies - }\#\text{ of error-related dependencies of grammatical sentence}} = \frac{2}{4-0} = 0.5$$

$$\text{Robustness } F_1 = 2 \times \frac{Precision \times Recall}{Precision + Recall} = 0.66$$

Experiments

Parser training data:

1. Penn Treebank (News data)
2. Tweebank (Twitter data)

Robustness test data:

1. English-as-a-Second Language writings (ESL)
2. Machine translation outputs (MT)

How do parsers perform on erroneous sentences?

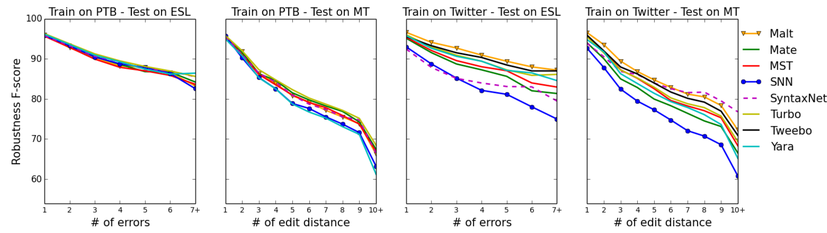
- All parsers are comparably robust on ESL, while they exhibit more differences on MT
- Training conditions matter, Malt prforms better when trained on Tweebank than PTB
- Training on Tweebank, Tweebo parser is as robust as others

Parser	Train on PTB §1-21			Train on Tweebank$_{train}$		
	UAS	Robustness F_1		UAF_1	Robustness F_1	
	PTB §23	ESL	MT	Tweebank$_{test}$	ESL	MT
Malt	*89.58*	93.05	76.26	77.48	**94.36**	80.66
Mate	**93.16**	93.24	77.07	76.26	91.83	75.74
MST	91.17	*92.80*	76.51	73.99	92.37	77.71
SNN	90.70	93.15	74.18	*53.4*	88.90	*71.54*
SyntaxNet	93.04	93.24	76.39	75.75	*88.78*	**81.87**
Turbo	92.84	**93.72**	**77.79**	79.42	93.28	78.26
Tweebo	-	-	-	**80.91**	93.39	79.47
Yara	93.09	93.52	*73.15*	78.06	93.04	75.83

Tweebo parser is not trained on Penn Treebank, because it is a specialization of Turbo parser to parse tweets.

To what extent is each parser impacted by the increase in number of errors?

- Robustness degrades faster with the increase of errors for MT than ESL
- Training on Tweebank helps some parsers to be more robust against many errors



What types of grammatical errors are more problematic for parsers?

- Replacement errors are the least problematic errors for all the parsers
- Missing errors are the most difficult errors

Parser	Train on PTB §1-21						Train on Tweebank$_{train}$					
	ESL			MT			ESL			MT		
	Repl.	Miss.	Unnec.	Repl.	Miss.	Unnec.	Repl.	Miss.	Unnec.	Repl.	Miss.	Unnec.
min	93.7 (MST)			92.8 (Yara)			89.4 (SyntaxNet)			87.8 (SNN)		
Malt												
Mate												
MST												
SNN												
SyntaxNet												
Turbo												
Tweebo												
Yara												
max	96.9 (Turbo)			97.2 (SNN)			97.8 (Malt)			97.6 (Malt)		

Each bar represents the level of robustness of each parser scaled to the lowest score (empty bar) and highest score (filled bar).

Conclusion

- Introducing a robustness metric without referring to a gold standard corpus
- Presenting a set of empirical analysis on the robustness of leading parsers
- Recommending practitioners to examine the range of ungrammaticality of input:
  - If it is more similar to tweets, Malt or Turbo parser may be good choices
  - If it is more similar to MT, SyntaxNet, Malt and Turbo parser are good choices
- The results suggest some preprocessing steps may be necessary for ungrammatical sentences, such as handling redundant and missing word errors