Intelligent Systems Program

University of Pittsburgh

Robust Parsing for Ungrammatical Sentences

Homa B. Hashemi

Dissertation Advisor: Dr. Rebecca Hwa

Parsing

- **NLP Goal**: understand and produce natural languages as humans do

As I remember , I have known her forever

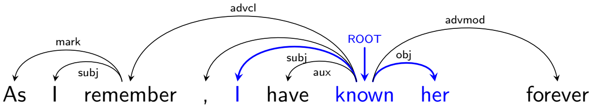
Parsing

- **NLP Goal**: understand and produce natural languages as humans do

As I remember , I have known her forever

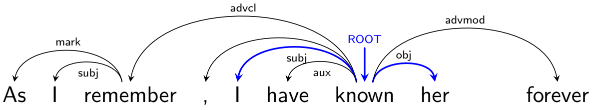
Parsing

- **NLP Goal**: understand and produce natural languages as humans do
- **Syntactic Parsing**: find relationship between individual words



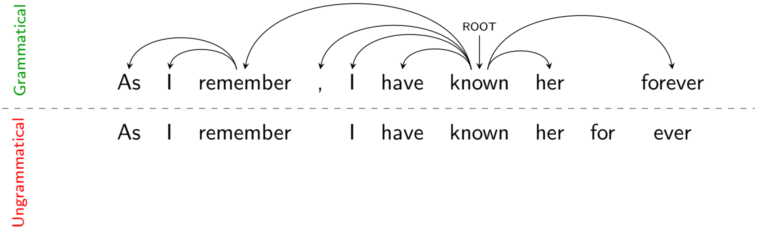
Parsing

- **NLP Goal**: understand and produce natural languages as humans do
- **Syntactic Parsing**: find relationship between individual words
- Parsing useful for many NLP applications, e.g: Question Answering, Machine Translation and Summarization
- If the parse is wrong, it would affect the downstream applications



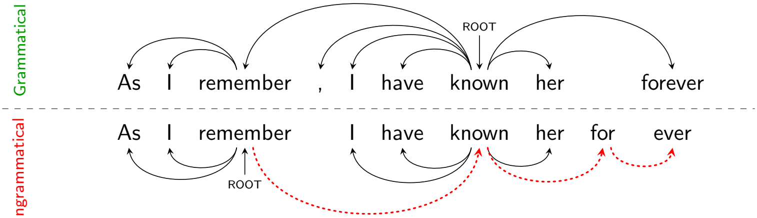
Parsing

- State-of-the-art parsers perform very well on grammatical sentences
- But even a small grammar error cause problems for them



Parsing

- State-of-the-art parsers perform very well on grammatical sentences
- But even a small grammar error cause problems for them

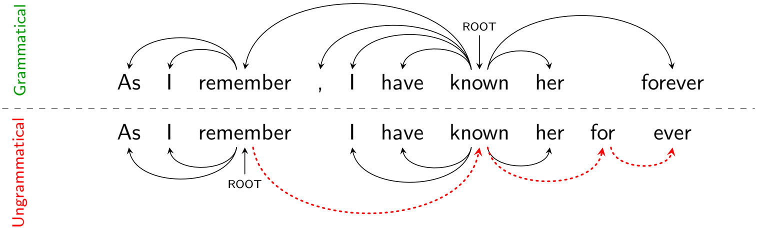


Parsing

- State-of-the-art parsers perform very well on grammatical sentences
- But even a small grammar error cause problems for them

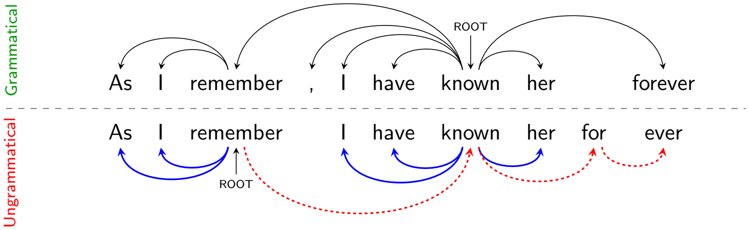
Question 1:

1. In what ways does a parser's performance degrade when dealing with ungrammatical sentences?



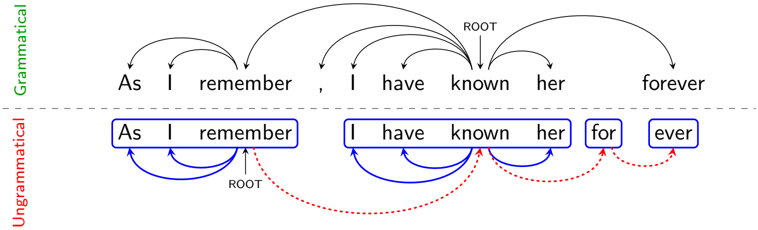
Parse Tree Fragments

- Parsers indeed have problems when sentences contain mistakes
- But there are still reliable parts in the parse tree unaffected by the mistakes



Parse Tree Fragments

- Parsers indeed have problems when sentences contain mistakes
- But there are still reliable parts in the parse tree unaffected by the mistakes $\Rightarrow$ **Tree Fragments**

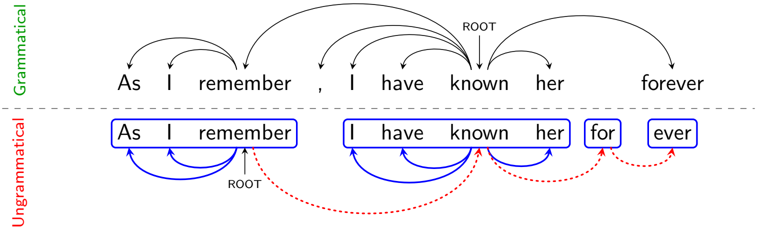


Parse Tree Fragments

- Parsers indeed have problems when sentences contain mistakes
- But there are still reliable parts in the parse tree unaffected by the mistakes $\Rightarrow$ **Tree Fragments**

Question 2:

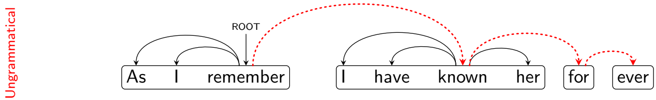
2. Is it feasible to automatically **identify parse tree fragments** that are plausible interpretations for the phrases they cover?



Tree Fragments in NLP Applications

Question 3:

3. Do the resulting parse tree fragments provide some useful information for downstream NLP applications?
   - Fluency Judgment
   - Semantic Role Labeling (SRL)



Contributions

1. Investigating the impact of ungrammatical sentences on parsers
2. Introducing the new framework of **parse tree fragmentation**
3. Verifying utility of tree fragments for two NLP applications

Overview

- Ungrammatical Sentences
- Q1: Impact of Ungrammatical Sentences on Parsing
- Q2: Parse Tree Fragmentation Framework
  - Development of a Fragmentation Corpus
  - Fragmentation Methods
- Q3: Empirical Evaluation of Parse Tree Fragmentation
  - Intrinsic Evaluation
  - Extrinsic Evaluation: Fluency Judgment
  - Extrinsic Evaluation: Semantic Role Labeling

Overview

- **Ungrammatical Sentences**
  - English-as-a-Second Language (ESL)
  - Machine Translation (MT)
- Q1: Impact of Ungrammatical Sentences on Parsing
- Q2: Parse Tree Fragmentation Framework
  - Development of a Fragmentation Corpus
  - Fragmentation Methods
- Q3: Empirical Evaluation of Parse Tree Fragmentation
  - Intrinsic Evaluation
  - Extrinsic Evaluation: Fluency Judgment
  - Extrinsic Evaluation: Semantic Role Labeling

English-as-a-Second Language (ESL)

- English learners tend to make mistakes
- To study ESL mistakes, researchers have created learner corpora:
  - **ESL Sentence:** We live in changeable world.
  - **Corrections:** (Missing determiner "a" at position 3), (An adjective needs replacing with "changing" between positions 3 and 4)
  - **Corrected ESL Sentence:** We live in a changing world.

Machine Translation (MT)

- Machine translation systems are not perfect and make mistakes
- To improve MT systems, researchers have created MT corpora:
  - **MT Output:** For almost 18 years ago the Sunda space "Ulysses" flies in the area.
  - **Reference Sentence:** For almost 18 years, the probe "Ulysses" has been flying through space.
  - **Post-edited Sentence:** For almost 18 years the "Ulysses" space probe has been flying in space.

Overview

- Ungrammatical Sentences
- **Impact of Ungrammatical Sentences on Parsing**
- Parse Tree Fragmentation Framework
  - Development of a Fragmentation Corpus
  - Fragmentation Methods
- Empirical Evaluation of Parse Tree Fragmentation
  - Intrinsic Evaluation
  - Extrinsic Evaluation: Fluency Judgment
  - Extrinsic Evaluation: Semantic Role Labeling

Research Question

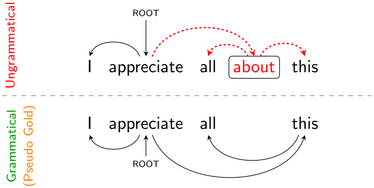
Question 1:

In what ways does a parser's performance degrade when dealing with ungrammatical sentences?

Impact of Ungrammatical Sentences on Parsing

1. To evaluate parsers we need manually annotated gold standards
   - But sizable ungrammatical treebanks are not available for ungrammatical domains
   - Also creating ungrammatical treebank is expensive and time-consuming
2. Gold standard free approach
   - We take the automatically produced parse tree of a grammatical sentence as **pseudo gold standard**
   - A parse is **robust** if the parse tree it produces for the ungrammatical sentence is similar to the tree of the corresponding grammatical sentence

Proposed Robustness Metric (Hashemi & Hwa, EMNLP 2016)



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

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

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

$$\text{Robustness } \mathrm{F}_1 = 2 \times \frac{Precision \times Recall}{Precision + Recall} = 0.66$$

Experiments

Compare 8 leading dependency parsers:

- Malt, Mate, MST, SNN, SyntaxNet, Turbo, Tweebo, Yara

Parser training data:

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

Robustness test data containing ungrammatical/grammatical sentences:

1. English-as-a-Second language writings (ESL): 10,000 sentences with 1+ errors
2. Machine translation outputs (MT): 10,000 sentences with 1+ errors

Overall Parsers Performance (Accuracy & Robustness)

- Trained on Penn Treebank:
  - All parsers have high accuracy on Penn Treebank
  - All parsers are comparably more robust on ESL than MT
- Trained on Tweebank (i.e. arguably more similar to test domains):
  - Parsers are more robust on ESL and even MT
  - Interestingly, Tweebo parser is as robust as others

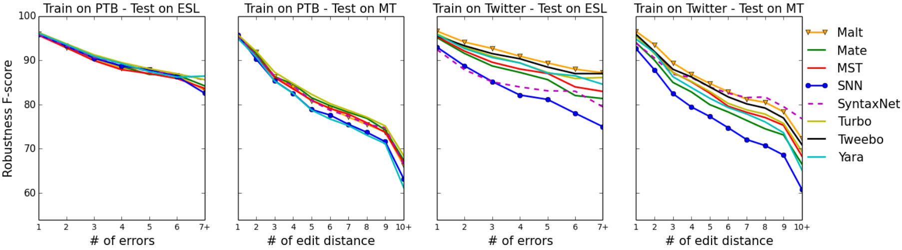
Parser	Train on PTB §1-21			Train on Tweebank_{train}		
	UAS	Robustness F_1		UAF_1	Robustness F_1	
	PTB §23	ESL	MT	Tweebank_{test}	ESL	MT
Malt	*89.58*	93.05	76.26	77.48	**94.36**	80.66
Mate	**93.16**	93.24	77.07	76.26	91.83	75.74
MST	91.17	*92.80*	76.51	73.99	92.37	77.71
SNN	90.70	93.15	74.18	*53.4*	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.

Parse Robustness by Number of Errors

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 help some parsers to be more robust against many errors



Impact of Grammatical Error Types on Parser Robustness

What types of grammatical errors are more problematic for parsers?

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

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.

Summary of Parser Robustness

- We have proposed a robustness metric without referring to a gold standard corpus
- We have presented a set of empirical analysis on the parser robustness of ungrammatical texts
- The results show that when ignoring erroneous parts of the ungrammatical sentences, parsers are doing reasonably well on finding syntactic structures of the remaining grammatical parts of the sentences
- Therefore, an alternative reasonable approach to parse ungrammatical sentences would be to omit the problematic structures

Overview

- Ungrammatical Sentences
- Impact of Ungrammatical Sentences on Parsing
- **Parse Tree Fragmentation Framework**
  - Development of a Fragmentation Corpus
  - Fragmentation Methods
- Empirical Evaluation of Parse Tree Fragmentation
  - Intrinsic Evaluation
  - Extrinsic Evaluation: Fluency Judgment
  - Extrinsic Evaluation: Semantic Role Labeling

Research Question

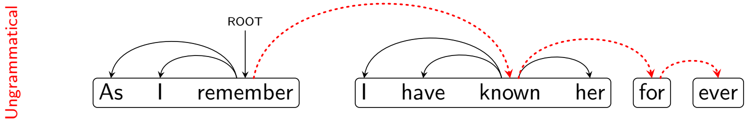
- There are reliable parts in the parse tree of ungrammatical sentences that are not affected by the mistakes

Question 2:

Is it feasible to automatically identify these unaffected areas of the parse tree and prune the problematic parts?

Parse Tree Fragmentation

- **Goal**: Identify and prune implausible dependency arcs
- **Tree fragments** are reasonable isolated parts of parse trees
- **Parse tree fragmentation** is the process of pruning the problematic parts of parse trees



Developing a Fragmentation Corpus

How to build gold fragments for ungrammatical sentences?

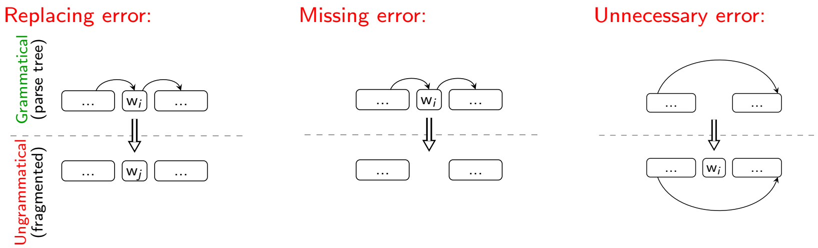
1. Manually annotate a fragmentation corpus
   - Annotation projects are expensive and time-consuming
   - Fragmentation may depend on the specific NLP application
2. Instead we leverage the existing corpora

Developing a Fragmentation Corpus: (1) PGold

(1) Pseudo Gold Fragmentation (PGold)

Reconstruct the ungrammatical sentence and its fragments using the parse tree of the grammatical sentence:

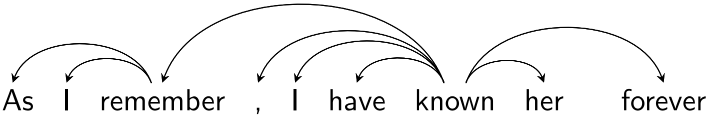
1. Prune the dependency arcs based on the type of the error



2. Prune arcs to or from the right or left words of the unaligned word that pass over it

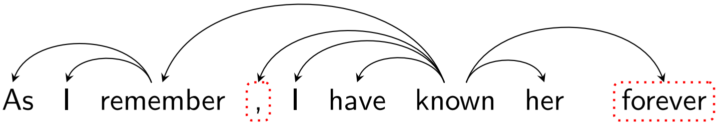
Developing a Fragmentation Corpus: (1) PGold example

- **Input**: Grammatical sentence and its parse tree



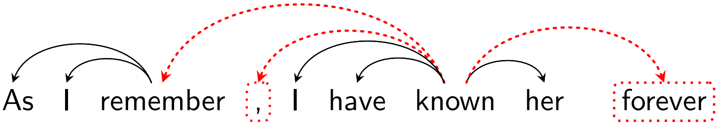
Developing a Fragmentation Corpus: (1) PGold example

- **Input**: Grammatical sentence and its parse tree
- The ungrammatical version has 2 errors: a missing comma and a phrase replacement error



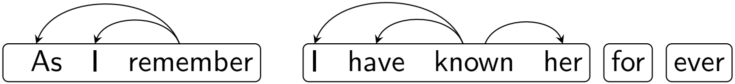
Developing a Fragmentation Corpus: (1) PGold example

- **Input**: Grammatical sentence and its parse tree
- The ungrammatical version has 2 errors: a missing comma and a phrase replacement error
- Reconstructing the ungrammatical sentence by applying:
  1. First error: missing comma
  2. Second error: replacement error



Developing a Fragmentation Corpus: (1) PGold example

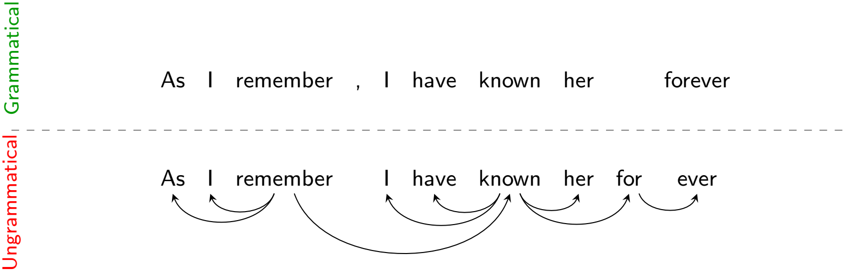
- **Input**: Grammatical sentence and its parse tree
- The ungrammatical version has 2 errors: a missing comma and a phrase replacement error
- Reconstructing the ungrammatical sentence by applying:
  1. First error: missing comma
  2. Second error: replacement error
- **Output**: PGold fragmentation of the ungrammatical sentence



(2) Reference Fragmentation (Reference)

Given an ungrammatical sentence and a grammatical version of the same sentence:

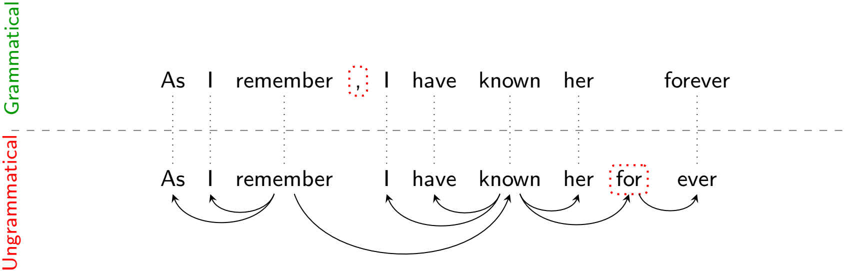
1. Parse ungrammatical sentence
2. Find alignments between grammatical/ungrammatical sentence
3. Prune arcs to and from the unaligned word
4. Prune arcs to or from the right or left words of the unaligned word that pass over it



(2) Reference Fragmentation (Reference)

Given an ungrammatical sentence and a grammatical version of the same sentence:

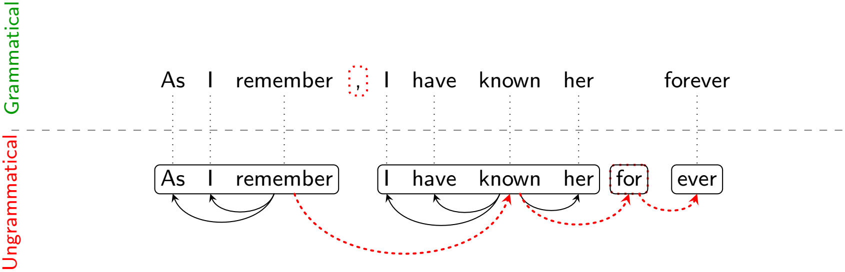
1. Parse ungrammatical sentence
2. Find alignments between grammatical/ungrammatical sentence
3. Prune arcs to and from the unaligned word
4. Prune arcs to or from the right or left words of the unaligned word that pass over it



(2) Reference Fragmentation (Reference)

Given an ungrammatical sentence and a grammatical version of the same sentence:

1. Parse ungrammatical sentence
2. Find alignments between grammatical/ungrammatical sentence
3. Prune arcs to and from the unaligned word
4. Prune arcs to or from the right or left words of the unaligned word that pass over it



Summary of Fragmentation Corpora

- Pseudo gold fragments (PGold)
  - Represent the most linguistically plausible interpretation of the ungrammatical sentence
  - Because PGold obtains fragments from parse trees of grammatical sentences
- Reference fragments (Reference)
  - May not be linguistically plausible
  - Because Reference fragments are formed from automatically parse trees of ungrammatical sentences
  - Thus, Reference represents an upperbound on what a real fragmentation algorithm could achieve

Overview

- Ungrammatical Sentences
- Impact of Ungrammatical Sentences on Parsing
- Parse Tree Fragmentation Framework
  - Development of a Fragmentation Corpus
  - **Fragmentation Methods**
    - Classification
    - Parser
    - sequence-to-sequence
- Empirical Evaluation of Parse Tree Fragmentation
  - Intrinsic Evaluation
  - Extrinsic Evaluation: Fluency Judgment
  - Extrinsic Evaluation: Semantic Role Labeling

Fragmentation methods: (1) Classification

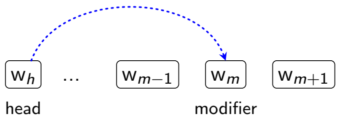
(1) Classification-based Parse Tree Fragmentation (Classification)

- Post-hoc process on generated parse trees of ungrammatical sentences
- Binary classification: Each arc is kept or cut

- **Input**: parse tree
- **Output**: fragmented tree

Features:

1. Depth & height of head, modifier
2. Part-of-speech tag of head,modifier
3. Word bigrams and trigrams



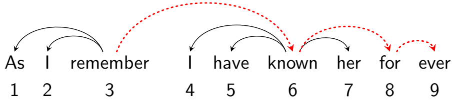
Training data: Parse trees fragments by Reference

Fragmentation methods: (2) Parser

(2) Parser Adaptation Parse Tree Fragmentation (Parser)

Jointly learns to parse a sentence and fragment it

- Build a treebank of ungrammatical sentences with their Reference fragments
- Train a state-of-the-art dependency parser
- **Input**: sentence
- **Output**: fragmented tree



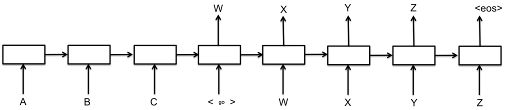
CoNLL format:

1	As	IN	3
2	I	PRP	3
3	remember	VB	0
4	I	PRP	6
5	have	VB	6
6	known	VB	0
7	her	PRP	6
8	for	IN	0
9	ever	RB	0

Fragmentation methods: (3) seq2seq

(3) Sequence-to-Sequence Parse Tree Fragmentation (seq2seq)

- Sequence-to-sequence Long Short-Term Memory (LSTM) model
  - Introduced by Sutskever et al. (2014) for translation



  - Used for parsing by Vinyals et al. (2015a)
    - **Input**: John has a dog
    - **Output**: (S (NP NNP $)_{NP}$ (VP VBZ (NP DT NN $)_{NP}$ $)_{VP}$.$)_{S}$

Fragmentation methods: (3) seq2seq

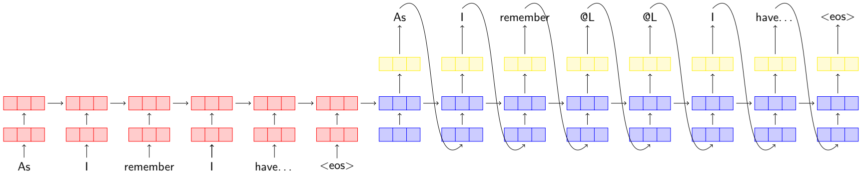
(3) Sequence-to-Sequence Parse Tree Fragmentation (seq2seq)

- seq2seq models require an effective representation for the input and the output to yield good performance
- We linearize dependency trees with **arc-standard transitions**:

Buffer	Stack	Action	Sequence
As I remember I have known her for ever			
I remember I have known her for ever	As	Shift	As
remember I have known her for ever	As I	Shift	I
I have known her for ever	As I remember	Shift	remember
I have known her for ever	As remember	Left-arc	@L
I have known her for ever	remember	Left-arc	@L
have known her for ever	remember I	Shift	I
known her for ever	remember I have	Shift	have
her for ever	remember I have known	Shift	known
her for ever	remember I known	Left-arc	@L
her for ever	remember known	Left-arc	@L
for ever	remember known her	Shift	her
for ever	remember known	Right-arc	@R
ever	remember known for	Shift	for
	remember known for ever	Shift	ever
	remember known for	Right-arc	@RCUT
	remember known	Right-arc	@RCUT
	remember	Right-arc	@RCUT

Example of Arc-Standard Actions

- Jointly parse and fragment sentences
- **Input**: As I remember I have known her for ever
- **Output**: As I remember @L @L I have known @L @L her @R for ever @RCUT @RCUT @RCUT



Summary of Fragmentation Methods

Method	Strength	Weakness
Classification	• A couple of thousand sentences is enough for training.	• It needs feature engineering. • It post-processes parser outputs, so parser's errors might propagate.
Parser retraining	• Jointly learns to parse and fragment. • Theoretically any dependency parser can be trained.	• It needs high quality or a huge amount of training data. • In practice, parsers' implementations matter. Because they perform differently even though they have the same underlying design.
seq2seq	• Jointly learns to parse and fragment. • No need for feature engineering. • No need for high quality annotated data, even noisy training data would be helpful.	• It needs a huge amount of parallel training data which might not be available for some ungrammatical domains.

Overview

- Ungrammatical Sentences
- Impact of Ungrammatical Sentences on Parsing
- Parse Tree Fragmentation Framework
  - Development of a Fragmentation Corpus
  - Fragmentation Methods
- Empirical Evaluation of Parse Tree Fragmentation
  - Intrinsic Evaluation
  - Extrinsic Evaluation: Fluency Judgment
  - Extrinsic Evaluation: Semantic Role Labeling

Empirical Evaluation of Parse Tree Fragmentation

- **Intrinsic Evaluation**:
  - Compare fragments against gold standard fragments
- **Extrinsic Evaluation**:
  - Evaluate potential uses of tree fragments in downstream applications:
    1. **Fluency Judgment**
    2. **Semantic Role Labeling**

Experimental Setup: Datasets

1. **English as a Second Language corpus (ESL)**
    - 5000 sentences with 1+ errors to train Classification
    - 576,000/30,000 sentences as train/development of Parser and seq2seq
    - 7000 sentences with 0+ errors to test

2. **Machine Translation outputs (MT)**

    Fluency score calculated by edit rates (HTER)

    - 4000 sentences with HTER score > 0.1 to train Classification
    - 9000/2000 sentences as train/development of Parser
    - 6000 sentences with HTER scores $\geqslant 0$ to test

    * No sizable parallel MT data to train seq2seq, so we use ESL seq2seq model and test it on MT

Experimental Setup: Tools

1. **Classification**
   - Use standard Gradient Boosting Classifier (Friedman, 2001)
2. **Parser**
   - Train the SyntaxNet parser (Andor, 2016), a transition-based neural network parser
3. **seq2seq**
   - Use OpenNMT (klein, 2017) package, a neural machine translation system on the Torch mathematical toolkit
   - 2-layer LSTMs with 750 dimensional hidden states

Intrinsic Evaluation: Performance of Each Fragmentation Method

Comparing resulting tree fragments against Reference fragments:

- **Unlabeled Attachment Score (UAS)**: percentage of words with correct head
- **Accuracy of Cut Arcs**: percentage of correct pruned dependency arcs

dataset	method	UAS	Accuracy of cut arcs		
			Precision_{cut}	Recall_{cut}	F-score_{cut}
ESL	Classification	61.36	0.35	0.79	0.48
	Parser	63	0.35	0.53	0.42
	seq2seq	82.4	0.71	0.57	**0.63**
MT	Classification	60.67	0.49	0.66	**0.56**
	Parser	50.55	0.43	0.70	0.54
	seq2seq (trained on ESL)	58.82	0.68	0.16	0.26
	Classification (trained on ESL)	62.23	0.51	0.52	0.51

Intrinsic Evaluation: Performance of Each Fragmentation Method

- In ESL, seq2seq method is more similar to the Reference

dataset	method	UAS	Accuracy of cut arcs		
			Precision_{cut}	Recall_{cut}	F-score_{cut}
ESL	Classification	61.36	0.35	0.79	0.48
	Parser	63	0.35	0.53	0.42
	seq2seq	82.4	0.71	0.57	**0.63**
MT	Classification	60.67	0.49	0.66	**0.56**
	Parser	50.55	0.43	0.70	0.54
	seq2seq (trained on ESL)	58.82	0.68	0.16	0.26
	Classification (trained on ESL)	62.23	0.51	0.52	0.51

Intrinsic Evaluation: Performance of Each Fragmentation Method

- In ESL, seq2seq method is more similar to the Reference
- In MT, Classification method is more similar to the Reference

dataset	method	UAS	Accuracy of cut arcs		
			Precision_{cut}	Recall_{cut}	F-score_{cut}
ESL	Classification	61.36	0.35	0.79	0.48
	Parser	63	0.35	0.53	0.42
	seq2seq	82.4	0.71	0.57	**0.63**
MT	Classification	60.67	0.49	0.66	**0.56**
	Parser	50.55	0.43	0.70	0.54
	seq2seq (trained on ESL)	58.82	0.68	0.16	0.26
	Classification (trained on ESL)	62.23	0.51	0.52	0.51

Intrinsic Evaluation: Performance of Each Fragmentation Method

- In ESL, seq2seq method is more similar to the Reference
- In MT, Classification method is more similar to the Reference
- Cross-domain model: Classification cuts more arcs, thus performs better on MT

dataset	method	UAS	Accuracy of cut arcs		
			Precision_{cut}	Recall_{cut}	F-score_{cut}
ESL	Classification	61.36	0.35	0.79	0.48
	Parser	63	0.35	0.53	0.42
	seq2seq	82.4	0.71	0.57	**0.63**
MT	Classification	60.67	0.49	0.66	**0.56**
	Parser	50.55	0.43	0.70	0.54
	seq2seq (trained on ESL)	58.82	0.68	0.16	0.26
	Classification (trained on ESL)	62.23	0.51	0.52	0.51

Intrinsic Evaluation: Evaluation of Tree Fragmentation Methods

Comparing resulting tree fragments against Reference fragments:

- **set-2-set P/R/F1**: percentage of shared arcs after mapping two fragment sets

dataset	method	Avg. #of Fragments	Avg. Size of Fragments	set-2-set P/R/F_1 to Reference
ESL	PGold	3.51	8.61	-
	Reference	3.51	8.60	0.97/0.97/0.97 (to PGold)
	Classification	7.29	2.40	0.90/0.57/0.67
	Parser	1.8	13.62	0.77/0.82/0.77
	seq2seq	2.92	9.36	0.85/0.85/**0.83**
MT	Reference	9.66	5.36	-
	Classification	12.96	2.09	0.71/0.57/0.60
	Parser	15.61	2.38	0.63/0.37/0.41
	seq2seq (trained on ESL)	2.29	18.70	0.54/0.72/0.59
	Classification (trained on ESL)	9.80	2.88	0.67/0.64/**0.62**

Intrinsic Evaluation: Evaluation of Tree Fragmentation Methods

Comparing resulting tree fragments against Reference fragments:

- **set-2-set P/R/F1**: percentage of shared arcs after mapping two fragment sets
- Reference fragments are the most similar to PGold

dataset	method	Avg. #of Fragments	Avg. Size of Fragments	set-2-set P/R/F_1 to Reference
ESL	PGold	3.51	8.61	-
	Reference	3.51	8.60	0.97/0.97/0.97 (to PGold)
	Classification	7.29	2.40	0.90/0.57/0.67
	Parser	1.8	13.62	0.77/0.82/0.77
	seq2seq	2.92	9.36	0.85/0.85/**0.83**
MT	Reference	9.66	5.36	-
	Classification	12.96	2.09	0.71/0.57/0.60
	Parser	15.61	2.38	0.63/0.37/0.41
	seq2seq (trained on ESL)	2.29	18.70	0.54/0.72/0.59
	Classification (trained on ESL)	9.80	2.88	0.67/0.64/**0.62**

Intrinsic Evaluation: Evaluation of Tree Fragmentation Methods

Comparing resulting tree fragments against Reference fragments:

- **set-2-set P/R/F1**: percentage of shared arcs after mapping two fragment sets
- Reference fragments are the most similar to PGold
- Reference produces more fragments in MT

dataset	method	Avg. #of Fragments	Avg. Size of Fragments	set-2-set P/R/F_1 to Reference
ESL	PGold	3.51	8.61	-
	Reference	3.51	8.60	0.97/0.97/0.97 (to PGold)
	Classification	7.29	2.40	0.90/0.57/0.67
	Parser	1.8	13.62	0.77/0.82/0.77
	seq2seq	2.92	9.36	0.85/0.85/**0.83**
MT	Reference	9.66	5.36	-
	Classification	12.96	2.09	0.71/0.57/0.60
	Parser	15.61	2.38	0.63/0.37/0.41
	seq2seq (trained on ESL)	2.29	18.70	0.54/0.72/0.59
	Classification (trained on ESL)	9.80	2.88	0.67/0.64/**0.62**

Overview

- Ungrammatical Sentences
- Impact of Ungrammatical Sentences on Parsing
- Parse Tree Fragmentation Framework
  - Development of a Fragmentation Corpus
  - Fragmentation Methods
- Empirical Evaluation of Parse Tree Fragmentation
  - Intrinsic Evaluation
  - Extrinsic Evaluation: Fluency Judgment
  - Extrinsic Evaluation: Semantic Role Labeling

Research Question

Question 3:

Do the resulting parse tree fragments provide some useful information for downstream NLP applications?

1. **Fluency Judgment**: Predict how natural a sentence might sound
2. **Semantic Role Labeling**: Discover semantic role of terms

Extrinsic Evaluation: Fluency Judgment

An automatic fluency judge can be used to:

- Decide whether an MT output needs to be post-processed
- Help grading student writings

Binary classification: a sentence has virtually no error or many errors

Regression: Predict number of errors in ESL dataset or edit rates in MT dataset

Our feature set:

1. Number of fragments
2. Average size of fragments
3. Minimum size of fragments
4. Maximum size of fragments

Extrinsic Evaluation: Fluency Judgment Results

ESL		
Feature Set	Binary Acc.(%)	Regression Pearson's r
Chance	76.1	
length	77.3	0.304
C&J	76.3	0.318
TSG	77.3	0.285
PGold	100	0.889
Reference	100	0.879
Classification	80.7	0.411
Parser Retraining	77.6	0.3
seq2seq	81.3	0.377

MT		
Feature Set	Binary Acc.(%)	Regression Pearson's r
Chance	72.2	
length	72	0.018
C&J	68.3	0.136
TSG	69.8	0.105
Reference	98.8	0.865
Classification	73.3	0.228
Parser Retraining	71.8	0.077
seq2seq (trained on ESL)	71.9	0.06
Classification (trained on ESL)	72.4	0.207

Experiments using 10-fold cross validation with Gradient Boosting Classifier
C&J: Charniak&Johnson, "Coarse-to-fine n-best parsing and MaxEnt discriminative reranking", ACL 2005.
TSG: Post, "Judging grammaticality with tree substitution grammar derivations", ACL 2011.

Extrinsic Evaluation: Fluency Judgment Results

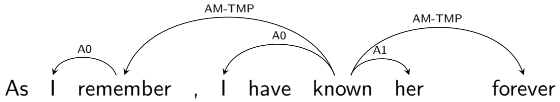
ESL		
Feature Set	Binary Acc.(%)	Regression Pearson's r
Chance	76.1	
length	77.3	0.304
C&J	76.3	0.318
TSG	77.3	0.285
PGold	100	0.889
Reference	100	0.879
Classification	**80.7**	**0.411**
Parser Retraining	77.6	0.3
seq2seq	**81.3**	0.377

MT		
Feature Set	Binary Acc.(%)	Regression Pearson's r
Chance	72.2	
length	72	0.018
C&J	68.3	0.136
TSG	69.8	0.105
Reference	98.8	0.865
Classification	**73.3**	**0.228**
Parser Retraining	71.8	0.077
seq2seq (trained on ESL)	71.9	0.06
Classification (trained on ESL)	72.4	0.207

Experiments using 10-fold cross validation with Gradient Boosting Classifier
C&J: Charniak&Johnson, "Coarse-to-fine n-best parsing and MaxEnt discriminative reranking", ACL 2005.
TSG: Post, "Judging grammaticality with tree substitution grammar derivations", ACL 2011.

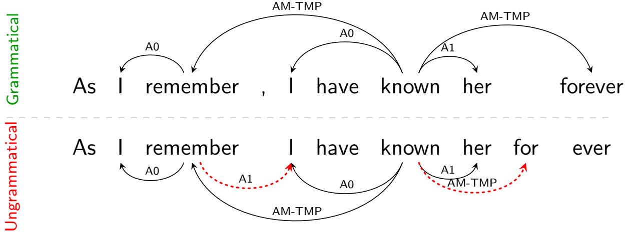
Extrinsic Evaluation: Semantic Role Labeling (SRL)

- SRL identifies relations between group of words with respect to a verb



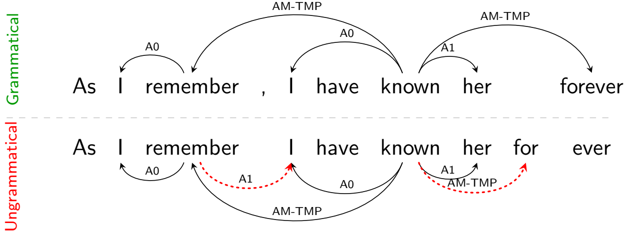
Extrinsic Evaluation: Semantic Role Labeling (SRL)

- SRL identifies relations between group of words with respect to a verb
- Grammatical mistakes have also impacts on semantic of the sentences



Extrinsic Evaluation: Semantic Role Labeling (SRL)

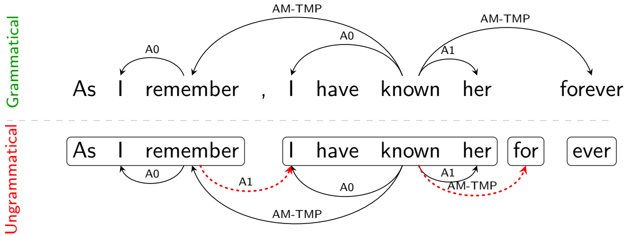
- SRL identifies relations between group of words with respect to a verb
- Grammatical mistakes have also impacts on semantic of the sentences



- Detecting *incorrect semantic dependencies* is crucial for applications that require high accuracy
  - e.g. Building accurate knowledge bases for question answering systems

Extrinsic Evaluation: Semantic Role Labeling (SRL)

- SRL identifies relations between group of words with respect to a verb
- Grammatical mistakes have also impacts on semantic of the sentences



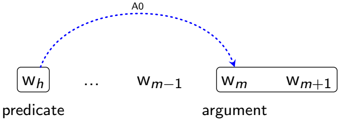
We hypothesize that through **parse tree fragmentation**, major syntactic problems can be identified; thus, tree fragments should be useful to detect *incorrect dependencies* of semantic role labeling

Detecting incorrect semantic dependencies

We introduce a binary classifier: indicate whether the semantic dependency is correct or incorrect

Features:

1. Binary feature denotes whether the semantic dependency crosses between parse tree fragments
2. Label of semantic dependency (e.g. A0).
3. Depth & height of predicate, argument
4. Part-of-speech tag of predicate, argument
5. Word bigrams and trigrams



Creating pseudo gold semantic dependencies

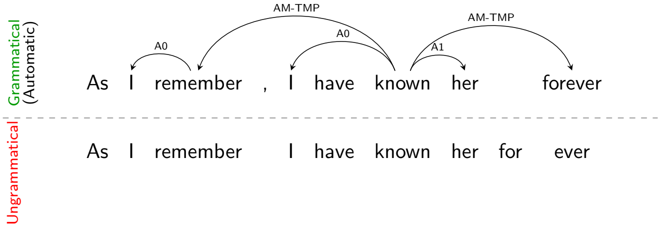
- We need ungrammatical sentences with annotated semantic dependencies

Ungrammatical

As I remember I have known her for ever

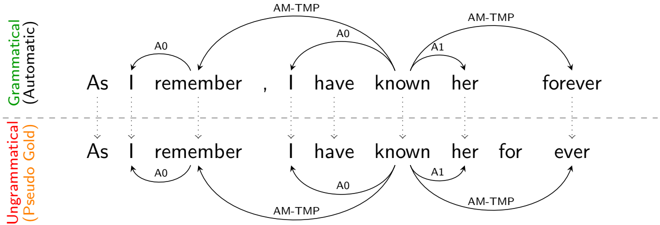
Creating pseudo gold semantic dependencies

- We need ungrammatical sentences with annotated semantic dependencies
- Similar to syntactic dependencies:
  - We take automatically produced semantic relations of corresponding grammatical sentence as gold standard



Creating pseudo gold semantic dependencies

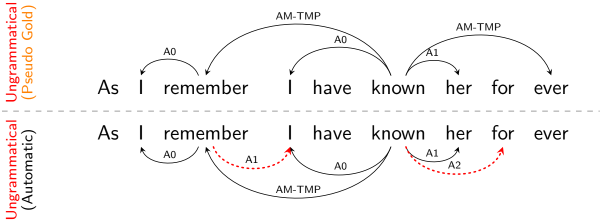
- We need ungrammatical sentences with annotated semantic dependencies
- Similar to syntactic dependencies:
    - We take automatically produced semantic relations of corresponding grammatical sentence as gold standard



Evaluating SRL Annotations of Ungrammatical Sentences

- Use CoNLL-2009 evaluation script to compare semantic dependencies
- **True Positive (TP)**: # of correct semantic dependencies
- **False Positives (FP)**: # of incorrect semantic dependencies (Type *I* error)
- Monitoring False Positives is crucial to evaluate helpfulness of fragmentation

$$\textbf{False Discovery Rate (FDR)} = \frac{\text{False Positive}}{\text{False Positive} + \text{True Positive}} = \frac{2}{2+4} \approx 33\%$$



Overall False Discovery Rates

Do parse tree fragments help detecting incorrect semantic dependencies?

ESL	
method	FDR (↓)
Basic	12.81
Reference	3.65
Classification	7.40
Parser	7.88
seq2seq	7.32

MT	
method	FDR (↓)
Basic	33.51
Reference	16.16
Classification	26.96
Parser	26.72
seq2seq (trained on ESL)	26.43
Classification (trained on ESL)	26.84

Overall False Discovery Rates

Do parse tree fragments help detecting incorrect semantic dependencies?

- **Basic** compares automatic semantic dependencies of ungrammatical sentences with pseudo gold dependencies

ESL	
method	FDR (↓)
Basic	12.81
Reference	3.65
Classification	7.40
Parser	7.88
seq2seq	7.32

MT	
method	FDR (↓)
Basic	33.51
Reference	16.16
Classification	26.96
Parser	26.72
seq2seq (trained on ESL)	26.43
Classification (trained on ESL)	26.84

Overall False Discovery Rates

Do parse tree fragments help detecting incorrect semantic dependencies?

- **Basic** compares automatic semantic dependencies of ungrammatical sentences with pseudo gold dependencies
- Applying fragmentation methods significantly helps

ESL	
method	FDR (↓)
Basic	12.81
Reference	3.65
Classification	7.40
Parser	7.88
seq2seq	7.32

MT	
method	FDR (↓)
Basic	33.51
Reference	16.16
Classification	26.96
Parser	26.72
seq2seq (trained on ESL)	26.43
Classification (trained on ESL)	26.84

Overall False Discovery Rates

Do parse tree fragments help detecting incorrect semantic dependencies?

- **Basic** compares automatic semantic dependencies of ungrammatical sentences with pseudo gold dependencies
- Applying fragmentation methods significantly helps
- seq2seq outperforms even though it learns both to parse and fragment

ESL	
method	FDR (↓)
Basic	12.81
Reference	3.65
Classification	7.40
Parser	7.88
seq2seq	**7.32**

MT	
method	FDR (↓)
Basic	33.51
Reference	16.16
Classification	26.96
Parser	26.72
seq2seq (trained on ESL)	**26.43**
Classification (trained on ESL)	26.84

Are some error types more challenging for SRL system?

- An error can be either in a verb role, an argument role, or no semantic role

ESL			
Method	Verb	Argument	No role
min	3.05 (Reference)		
Basic			
Reference			
Classification			
Parser			
seq2seq			
max	18.09 (Parser)		

MT			
Method	Verb	Argument	No role
min	7.71 (Reference)		
Basic			
Reference			
Classification			
Parser			
seq2seq (trained on ESL)			
Classification (trained on ESL)			
max	20.1 (Classification)		

Impact of error semantic role on Detecting Incorrect Semantic Dependencies

Are some error types more challenging for SRL system?

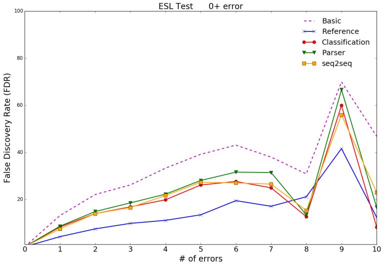
- An error can be either in a verb role, an argument role, or no semantic role
- Sentences with argument errors are more challenging

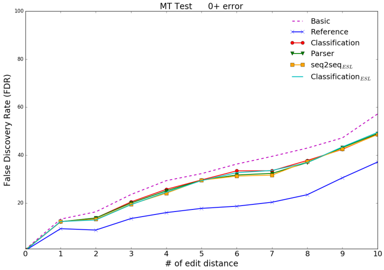
ESL			
Method	Verb	Argument	No role
min	3.05 (Reference)		
Basic			
Reference			
Classification			
Parser			
seq2seq			
max	18.09 (Parser)		

MT			
Method	Verb	Argument	No role
min	7.71 (Reference)		
Basic			
Reference			
Classification			
Parser			
seq2seq (trained on ESL)			
Classification (trained on ESL)			
max	20.1 (Classification)		

Incorrect Semantic Dependencies by Number of Errors

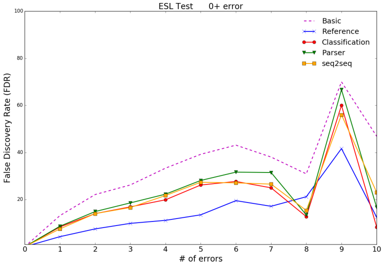
- To what extent parse tree fragmentation helps by increasing number of errors?
  - FDR score is increasing more rapidly for the Basic than Reference

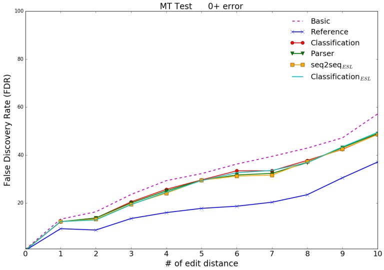




Incorrect Semantic Dependencies by Number of Errors

- To what extent parse tree fragmentation helps by increasing number of errors?
  - FDR score is increasing more rapidly for the Basic than Reference





- Fragmentation features are useful to detect some of incorrect semantic dependencies
- Reference significantly helps SRL as the upper bound approach

Conclusion

Examining the problems of parsing ungrammatical sentences:

- Analyzing the negative impact of ungrammatical sentences on
  - State-of-the-art statistical parsers
- Introducing the new framework of **parse tree fragmentation**
  - By pruning implausible dependency arcs of parse trees
- Empirical studies shows that fragmenting trees is helpful for NLP applications
  - Sentence-level fluency judgment
  - Semantic role labeling

Publications and Future Work

Publications:

- Hashemi & Hwa, An Evaluation of Parser Robustness for Ungrammatical Sentences, EMNLP, 2016.
- Hashemi & Hwa, Parse Tree Fragmentation of Ungrammatical Sentences, IJCAI, 2016.
- Hashemi & Hwa, Jointly Parse and Fragment Ungrammatical Sentences, AAAI, 2018.

Future Work:

- Expanding parser robustness evaluation on various domains
- Applying fragmentation on a wider set of applications
- Building specialized parsers to handle ungrammatical sentences, e.g by adding new actions to transition-based dependency parsers

obj

Thank You

References

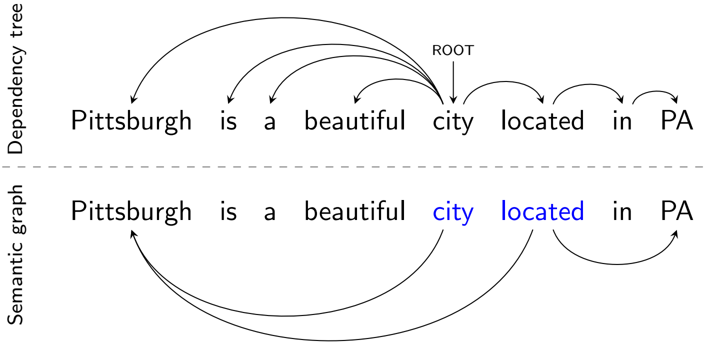
- Andor, D., Alberti, C., Weiss, D., Severyn, A., Presta, A., Ganchev, K., Petrov, S., and Collins, M. (2016). Globally normalized transition-based neural networks. arXiv.
- Friedman, J. H. (2001). Greedy function approximation: a gradient boosting machine. Annals of statistics.
- Klein, G., Kim, Y., Deng, Y., Senellart, J., and Rush, A. M. (2017). OpenNMT: Open-source toolkit for neural machine translation. arXiv.
- Sutskever, I., Vinyals, O., and Le, Q. V. (2014). Sequence to sequence learning with neural networks. NIPS.
- Vinyals, O., Kaiser, Å, Koo, T., Petrov, S., Sutskever, I., and Hinton, G. (2015a). Grammar as a foreign language. NIPS.
- Vinyals, O., Bengio, S., and Kudlur, M. (2015b). Order matters: Sequence to sequence for sets. arXiv.

Evaluation of Classification-based Parse Tree Fragmentation

- Classification runs a binary prediction to decide to keep an edge or cut it
- Unbalanced data (few edges are cut)
- Never cutting any edge results in high accuracy: 84% on ESL, 65% on MT
- Thus, we evaluate classifiers with AUC measure

method	**ESL**	**MT**
No cut baseline	0.5	0.5
Classification	0.75	0.63

Relation of Syntactic and Semantic Dependencies



Relationships between Fragments Statistics

ESL dataset

Method	# of Fragments		size of Fragments	
	Pearson r	RMSE (↓)	Pearson r	RMSE (↓)
Classification	**0.453**	5.086	**0.299**	0.543
Parser	0.092	3.946	0.076	0.545
seq2seq	0.407	**3.068**	0.281	**0.444**

MT dataset

Method	# of Fragments		size of Fragments	
	Pearson r	RMSE (↓)	Pearson r	RMSE (↓)
Classification	**0.646**	7.433	**0.377**	0.335
Parser	0.527	11.135	0.223	0.364
seq2seq (trained on ESL)	0.012	10.212	-0.011	0.654
Classification (trained on ESL)	0.589	**6.169**	0.326	**0.327**

Correlation between 4 fluency features

ESL dataset

Method	# of fragments	Avg. size	Min size	Max size
Reference	*0.842*	*-0.822*	*-0.765*	*-0.766*
Classification	**0.409**	**-0.317**	-0.178	**-0.241**
Parser	0.099	-0.093	-0.084	-0.063
seq2seq	0.285	-0.241	**-0.215**	-0.177

MT dataset

Method	# of fragments	Avg. size	Min size	Max size
Reference	*0.662*	*-0.608*	*-0.476*	*-0.77*
Classification	0.155	-0.122	-0.047	-0.171
Parser	0.081	-0.056	-0.042	-0.082
seq2seq (trained on ESL)	0.076	-0.077	**-0.073**	-0.058
Classification (trained on ESL)	**0.191**	**-0.148**	-0.06	**-0.179**

Mapping each fragment of the first set S_1 with a fragment of the second set S_2 that have the maximum number of shared edges:

$$Precision = \frac{\text{number of shared edges between all mapped fragments}}{\text{total number of edges of } S_1}$$

$$Recall = \frac{\text{number of shared edges between all mapped fragments}}{\text{total number of edges of } S_2}$$

$$F_1(S_1, S_2) = 2 \times \frac{Precision \times Recall}{Precision + Recall}$$