# RST Parser

**Parminder Bhatia** **Pavan Kumar Ramnath**

## 1 Introduction

Discourse Parsing deals with the identification of elementary discourse units (EDUs) and the relations between them.

### 1.1 RST Parser

Rhetorical Structure Theory (RST) is a model to understand the coherence of a text. In this part of the project, features are added to an RST parser based on the enhancements mentioned in the paper Ji and Eisenstein [2014].

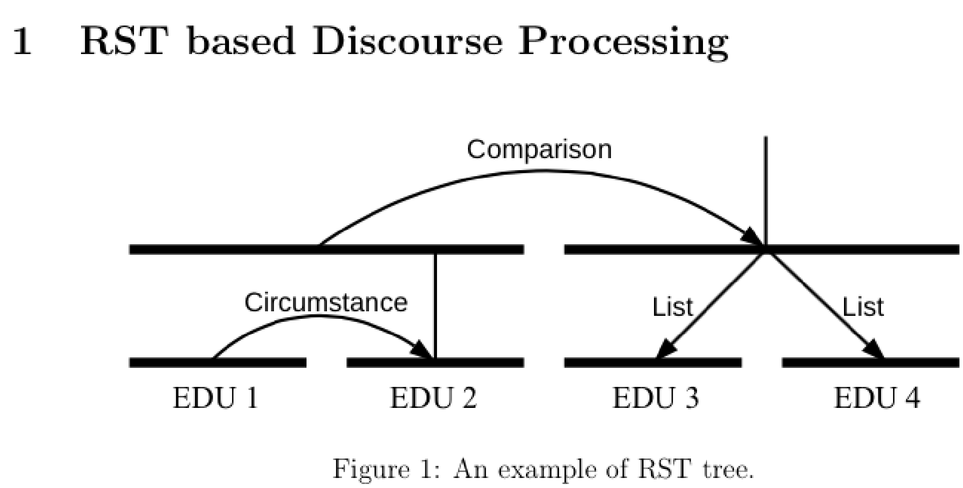



Figure 1: An example of RST tree.

## 2 Feature Space/Algorithm

Structural features such as distance of node from beginning or end are already provided as part of bare minimum implementation. We have added the following lexical features to improve the parsing:

### 2.1 Words

Words are an important part of the features and there are various features which are captured by the words.

- **First Word** - In this feature we capture the first word of each node. Idea is that reduction or shift depends on the first word. For example: 'but','and' at the beginning of Queue might suggest that it should be reduced with the element at top of the Stack. Similarly presence of a capital letter as in 'The' might suggest that shift is a better option. We have 3 type of features for this. **('StackSpan1','BEGIN-WORD-STACK1',word), ('QueueSpan1','BEGIN-WORD-QUEUE1',word), ('StackSpan2','BEGIN-WORD-STACK2',word)**
- **Last Word** - In this feature we capture the last word of each node for similar reasons. For example: '.', ',' at the end of the Stack might suggest that it should be reduced with the element in front of the Queue. We have 3 type of features for this. **('StackSpan1','END-WORD-STACK1',word), ('QueueSpan1','END-WORD-QUEUE1',word), ('StackSpan2','END-WORD-STACK2',word)**
- **First - Last Word** combination helps us identiy the structure within the span.

### 2.2 Part of Speech Tag

POS tags play a crucial role, as sentences have same general structure and thus POS tends to cluster words of same POS, which gives us good results.

- **First Word POS** - In this feature we capture POS of the first word of each node. Idea is that reduction or shift depends on the first word. For example - 'but', 'and' as conjunctions at the beginning of Queue might suggest that it should be reduced with the element at top of the Stack. We have 3 type of features for this. **('StackSpan1','BEGIN-POS-STACK1',word) , ('QueueSpan1','BEGIN-POS-WORD-QUEUE1',word)**
- **First Word - First Word POS** Combination of first word POS of element on top of stack and queue word helps us identify the general grammatical structures.

### 2.3 Dependency Tag

Dependency parsing of a sentence gives us the relation and dependencies of each word on other words in a sentence and this plays a crucial role for decision between shift and reduce. For each EDU we identify root words or words whose roots lie in other EDUs. We basically make two types of features depending on whether it is the root word or has dependency on a word belonging to other EDU. For example - we make **'StackSpan1','DEPENDENCY-BEGIN-END-WORD-STACK1',word** if it is root and has no dependency on words outside it's node and we generate **'StackSpan1','DEPENDENCY-BEGIN-OTHER-WORD-STACK1',word** for words having dependency outside the EDU.

### 2.4 Same Sentence

This a boolean type of feature where we check if Stack Span1 and Queue Span1 belong to a same sentence or not. For example - EDUs belonging to same sentence would have higher chance of reduction as compared to those belonging to different sentences.

### 2.5 Number of words

Though not a major factor, having the knowledge of the length of words in a span helps in deciding between the shift or reduce step. For this, we have a feature as length of words(Stack1 ,Stack2), which would give us the reason why these two spans were not reduced. Similarly we have length of words(Stack1,Queue1) as a feature.

### 2.6 Coreference List

We have created a list of 5 - 7 words and check if stack span1 or queue span1 contains any of these words. Presence of coreference is an indicator or relationship between the stack span1 and queue span1. Thus, if we have ’he’,’it’, ’she’ in queue span1 , then there is high possibility of reduce step as these words basically refers to subject present in stack span1.

## 3 Code and Implementation

Implementation has been done in the following files:

**feature.py** : This file contains all the features generated in part 1 of the project.

**offlineDEP.py** : Responsible for creating dependency parse for documents and save the root word or words which have dependency on other EDU. Root is saved as (’warning’, ’ROOT’, ’R’) and dependency on other EDU word is stored as (’paid’, ’stick’, ’U’).

**offlinePOS4Sentence.py** : Responsible for creating POS tags for documents and saving them to file. Note that both (offlineDEP and offlinePOS4Sentence) are operated sentence at the time and then are broken down for respective EDUs.

**offlinePOS4ALL.py** : Same as above but creates POS based on EDUs (not sentences)

**offlineSentence.py** : Responsible for assigning the line to which particular EDU belongs.

In addition to these, other files have also been changed to enable loading of offline features as well as addition of these features into the SpanNode class.

## 4 Evaluation and Results

Using the above implementation , we got the following results. :
F1 score on nuclearity level is 0.644
F1 score on relation level is 0.522
F1 score on span level is 0.798
We can see that the results are quite similar to those mentioned in Sagae [2009].

In order to understand the impact of each of these broad features, we performed ablution tests on the dataset.

| DESCRIPTION | ALL | POS | DEPENDENCY | SAME SENTENCE | COREF | NUMBER OF WORDS |
|---|---|---|---|---|---|---|
| All Features | 52.2 | 50.5 | 49.2 | 51.6 | 51.5 | 49.4 |

Figure 2: Relation

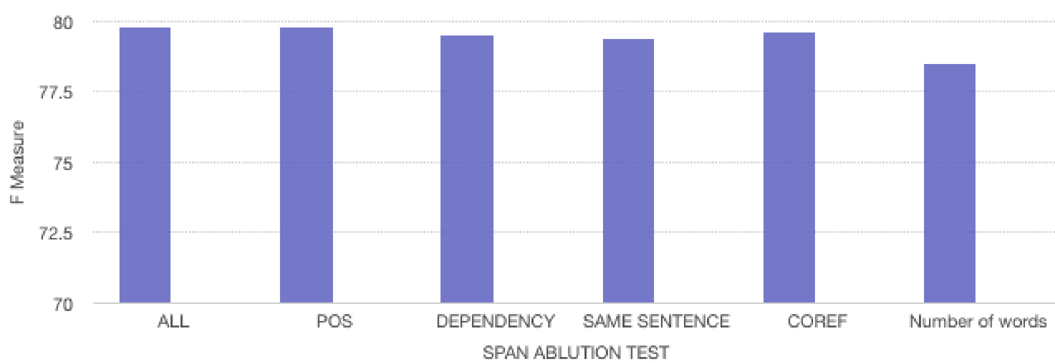
| DESCRIPTION | ALL | POS | DEPENDENCY | SAME SENTENCE | COREF | NUMBER OF WORDS |
|---|---|---|---|---|---|---|
| All Features | 64.4 | 62.5 | 61.9 | 63.5 | 63.8 | 61.2 |

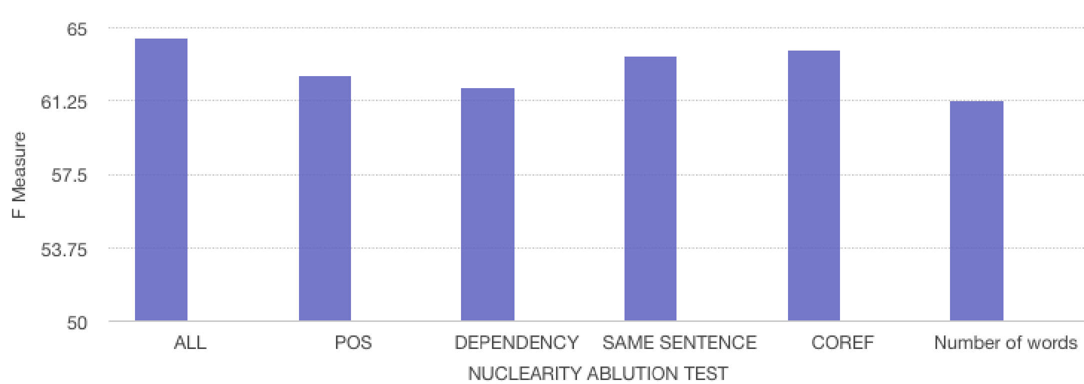
Figure 3: Nuclearity

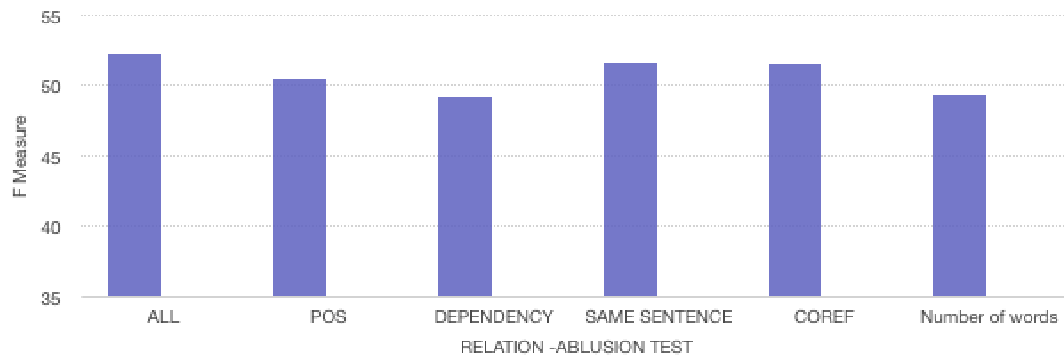
| DESCRIPTION | ALL | POS | DEPENDENCY | SAME SENTENCE | COREF | NUMBER OF WORDS |
|---|---|---|---|---|---|---|
| All Features | 79.8 | 79.8 | 79.5 | 79.4 | 79.6 | 78.5 |

Figure 4: Span

These results seem to indicate that words have a major impact. Hence we perform evaluation of individual features within words using absolution graphs of individual features.





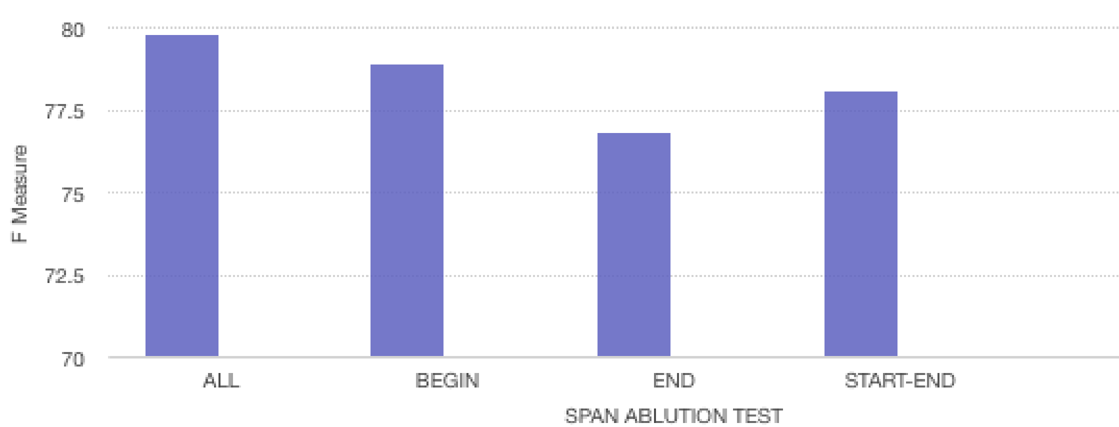

We then perform ablution of specific features within words. We have 3 cases of ablution: first word, last word and first-last word pair. Evaluation using just the specific features for words are as follows.

| DESCRIPTION | ALL | BEGIN | END | START-END |
|---|---|---|---|---|
| All Features | 64.4 | 62.3 | 58.2 | 60 |

| DESCRIPTION | ALL | BEGIN | END | START-END |
|---|---|---|---|---|
| All Features | 52.2 | 50 | 44.4 | 48.1 |

| DESCRIPTION | ALL | BEGIN | END | START-END |
|---|---|---|---|---|
| All Features | 79.8 | 78.9 | 76.8 | 78 |

Figure 5: Relation Ablution Tests



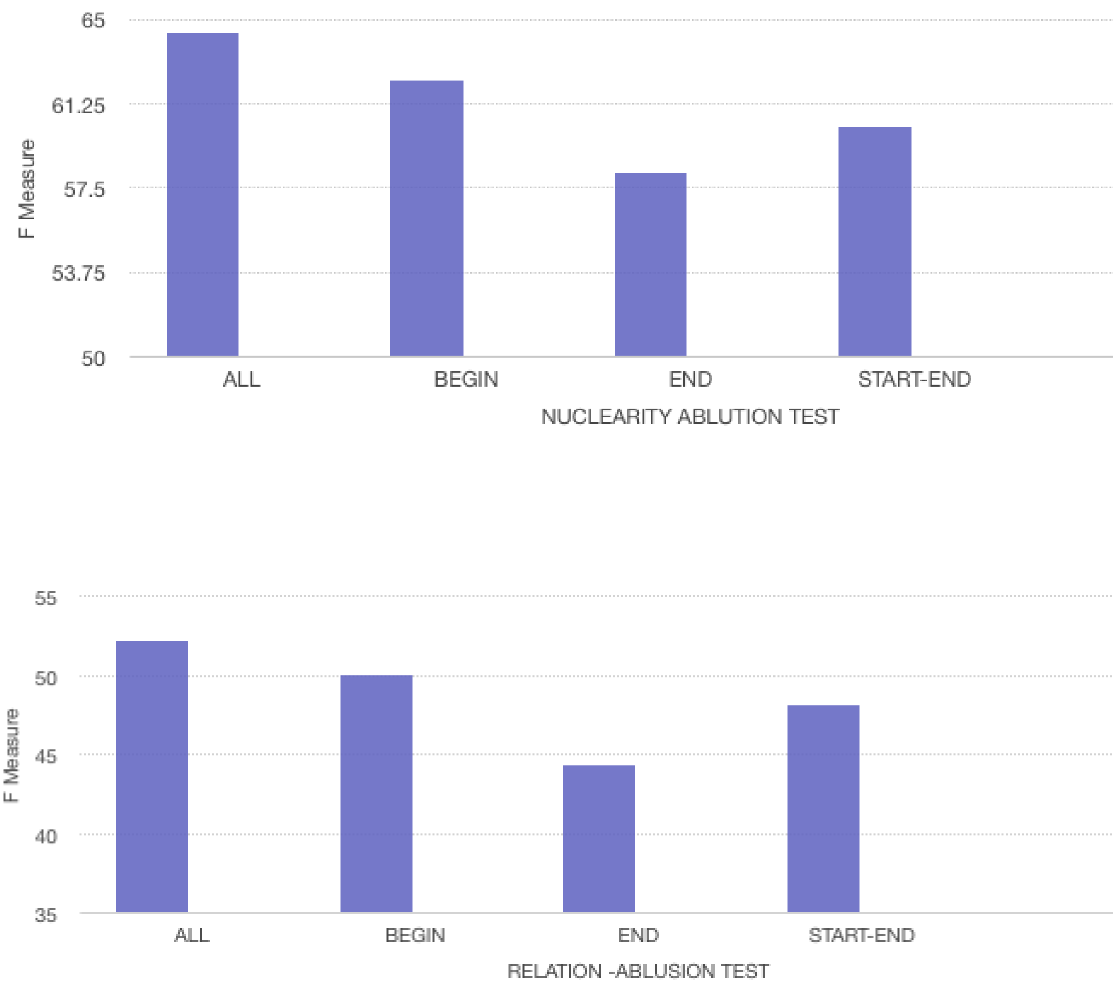



Figure 6: Relation Ablution Tests

## 5 Discussion

From the above graphs we can see that all the above features play critical role in some parameter or the other. Clearly, words have the most impact, whereas other features such as dependency and POS are critical from relation and nuclearity perspectives.

Among the words features, ending word has maximum impact especially in relation and nuclearity, as it provides information about the end of sentence which is important for deciding between shift and reduce. This is followed by a pair of start-end word.

## References

Yangfeng Ji and Jacob Eisenstein. Representation learning for text-level discourse parsing. 2014.

Kenji Sagae. Analysis of discourse structure with syntactic dependencies and data-driven shift-reduce parsing. 2009.