CODRA: A Novel Discriminative Framework for Rhetorical Analysis

Shafiq Joty Giuseppe Carenini
Raymond T. Ng

Where are we now !!

- Phonology
- Morphology
- Syntax
- Semantics
- Pragmatics
- **Discourse**

Discourse Analysis

Analysis of language 'beyond the sentence'. This contrasts with types of analysis more typical of modern linguistics, which are chiefly concerned with the study of grammar: the study of smaller bits of language, such as sounds (phonetics and phonology), parts of words (morphology), meaning (semantics), and the order of words in sentences (syntax). Discourse analysts study larger chunks of language as they flow together.

Documents are more than individual meaning of each sentence unit.

Understanding the coherence of a text.

Rhetorical Structure Theory

proposed by Mann and Thompson (1988),

RST represents texts by labeled hierarchical structures, called Discourse Trees (DTs)

- leaves of a DT correspond to contiguous atomic text spans (EDU)
- EDUs are clause-like units that serve as building blocks.
- Adjacent EDUs are connected by coherence relations (e.g., Elaboration, Contrast), forming larger discourse units (represented by internal nodes), which in turn are also subject to this relation linking.

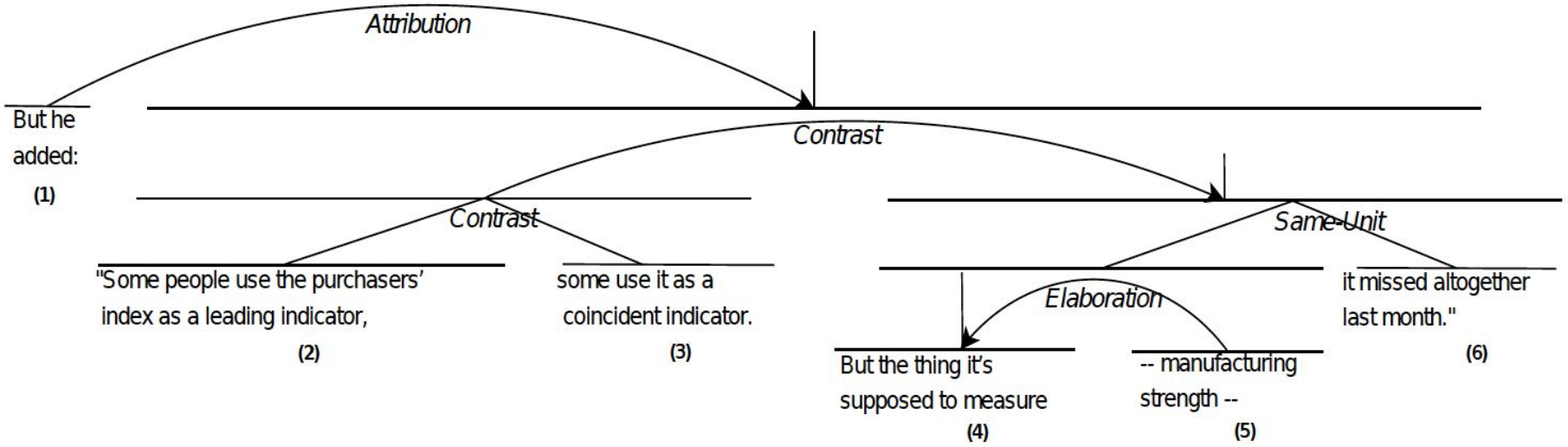


Figure 1
Discourse tree for two sentences in RST–DT. Each sentence contains three EDUs. Horizontal lines indicate text segments; satellites are connected to their nuclei by curved arrows and two nuclei are connected with straight lines.

RST contd.

- Discourse units are distinguished based on their relative importance in the text
- Nuclei are the core parts of the relation and satellites are peripheral or supportive ones.

Relation Name	Nucleus	Satellite
Background	text whose understanding is being facilitated	text for facilitating understanding
Elaboration	basic information	additional information
Preparation	text to be presented	text which prepares the reader to expect and interpret the text to be presented.
Contrast	one alternate	the other alternate (another nucleus)

RST contd.

rhetorical analysis => two subtasks: discourse segmentation, discourse parsing

- discourse segmentation is the task of breaking the text into a sequence of EDUs,
- discourse parsing is the task of linking the discourse units (EDUs and larger units) into a labeled tree.

Existing Discourse Parsers

Limitations=>

- Typically model the structure and the labels of a DT separately, and also do not take into account the sequential dependencies between the DT constituents.
- Apply greedy and sub-optimal parsing algorithms to build a DT.
- No discrimination between intrasentential parsing (i.e., building the DTs for the individual sentences) and multi sentential parsing (i.e., building the DT for the whole document).

CODRA

CODRA comprises a discourse segmenter and a discourse parser.

- Discourse segmenter, which is based on a binary classifier, identifies the elementary discourse units in a given text.
- Discourse parser builds a discourse tree by applying an optimal parsing algorithm to probabilities inferred from two Conditional Random Fields: one for intra-sentential parsing and the other for multi-sentential parsing.

CODRA Architecture

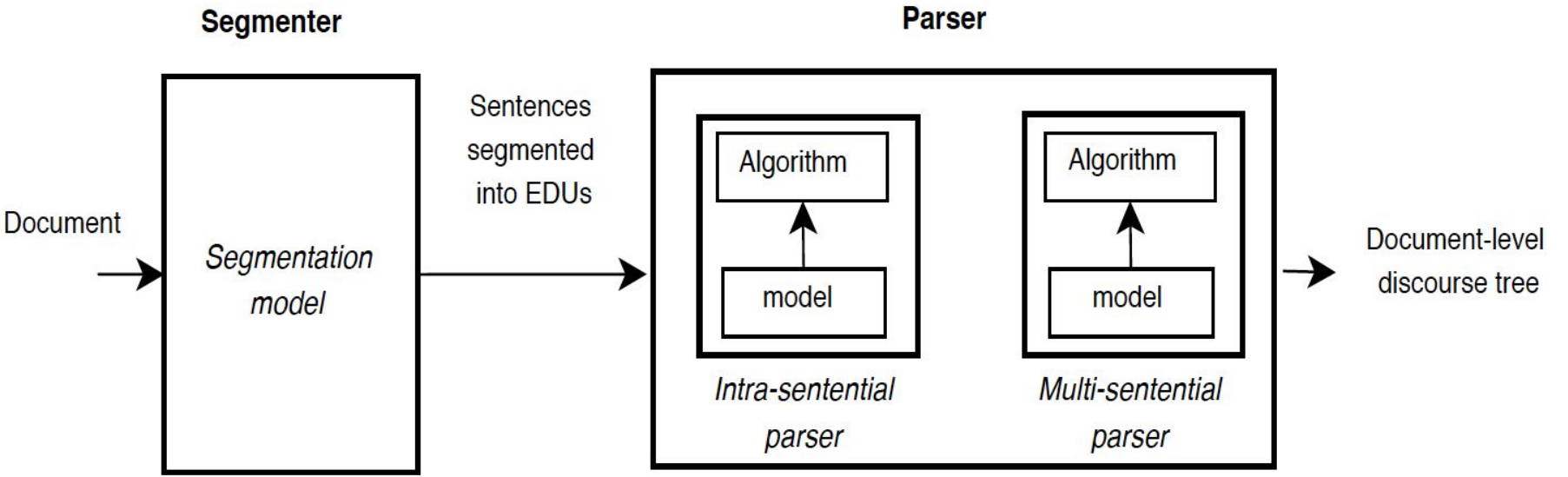


Figure 2
CODRA architecture.

CODRA Architecture contd.

Given a raw text, the first task in the rhetorical analysis pipeline is to break the text into a sequence of EDUs (i.e., discourse segmentation). Because it is taken for granted that sentence boundaries are also EDU boundaries (i.e., EDUs do not span across multiple sentences), the discourse segmentation task boils down to finding EDU boundaries inside sentences.

CODRA uses a maximum entropy model for discourse segmentation

CODRA Parser

First Component => A parsing model

To explore the search space of possible structures and labels for their nodes. A probabilistic parsing model like CRF assigns a probability to every possible DT.

Second Component => A parsing algorithm

To select the best parse tree(s) among the candidates.

CODRA Parser contd.

CODRA applies an optimal CKY parsing algorithm to the inferred posterior probabilities (obtained from the CRFs) of all possible DT constituents. Furthermore, the parsing algorithm allows CODRA to generate a list of k-best parse hypotheses for a given text

Intra-sentential parser and a Multi-sentential parser

CODRA comprises two separate modules: an intra-sentential parser and a multi-sentential parser, as shown in Figure 2.

First, the intra-sentential parser produces one or more discourse sub-trees for each sentence.

Then, the multi-sentential parser generates a full DT for the document from these

sub-trees.

Motivation behind 2 different Models

A simple and straightforward strategy would be to use a single unified parsing model for both intra- and multi-sentential parsing without distinguishing the two cases.

It appears that discourse relations are distributed differently intra- versus multi-sententially.

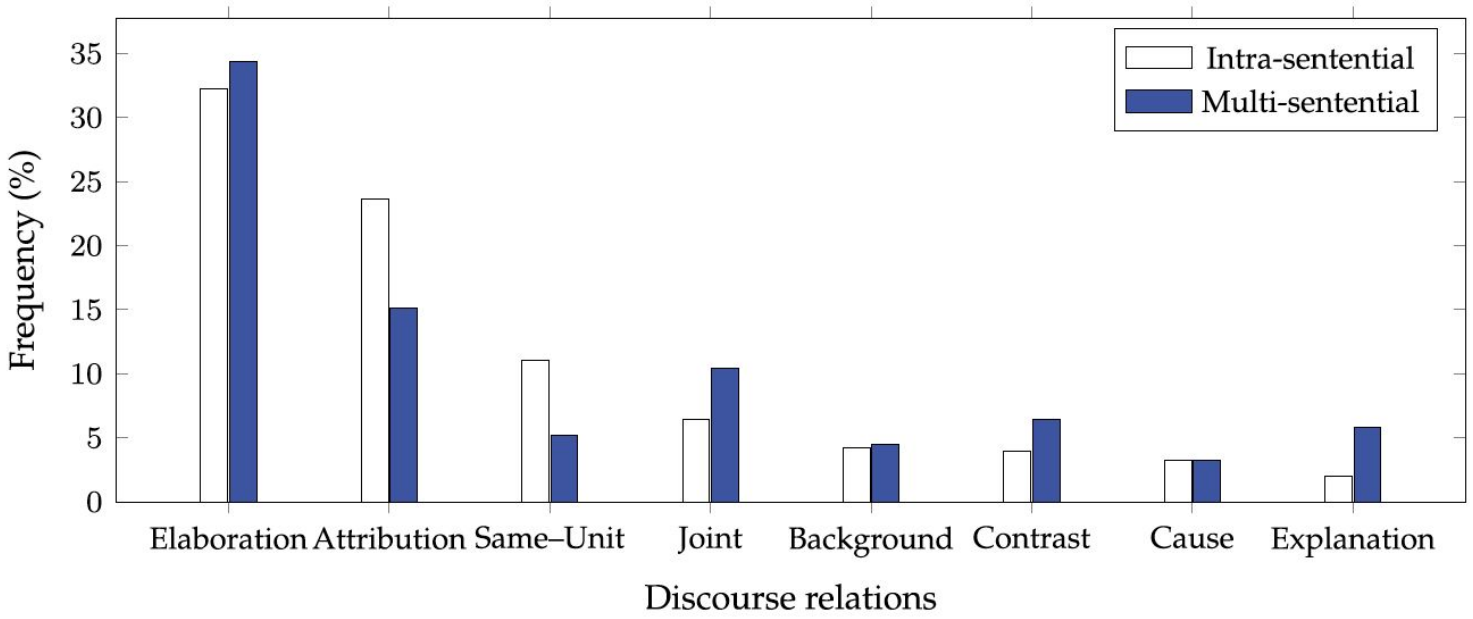


Figure 3
Distributions of the eight most frequent relations in intra-sentential and multi-sentential parsing scenarios on the RST–DT training set.

Motivation behind 2 different Models

different kinds of features are applicable and informative for intra versus multi-sentential parsing. For example, syntactic features like dominance sets (Soricut and Marcu 2003) are extremely useful for parsing at the sentence-level, but are not even applicable in the multi-sentential case. Likewise, lexical chain features (Sporleder and Lapata 2004), which are useful for multi-sentential parsing, are not applicable at the sentence level.

Necessary Terminologies

A DT can be formally represented as a set of constituents of the form R[i,m, j], where $i < m < j$. This refers to a rhetorical relation R between the discourse unit containing EDUs *i* through *m* and the discourse unit containing EDUs *m*+1 through j. For example, the DT for the second sentence in Figure 1 can be represented as Elaboration–NS[4,4,5], Same–Unit–NN[4,5,6]g.

Intra-Sentential Parser

The observed nodes Uj (at the bottom) in a sequence represent the discourse units (EDUs or larger units).

The first layer of hidden nodes are the structure nodes, where $S_j \in \{0,1\}$ denotes whether two adjacent discourse units U_j-1 and U_j should be connected or not.

The second layer of hidden nodes are the relation nodes, with $R_j \in \{1 \ldots M\}$denoting the relation between two adjacent units U_j-1 and U_j, where M is the total number of relations in the relation set.

The intra-sentential parsing model of CODRA

Sequential Dependencies

Structure and label jointly

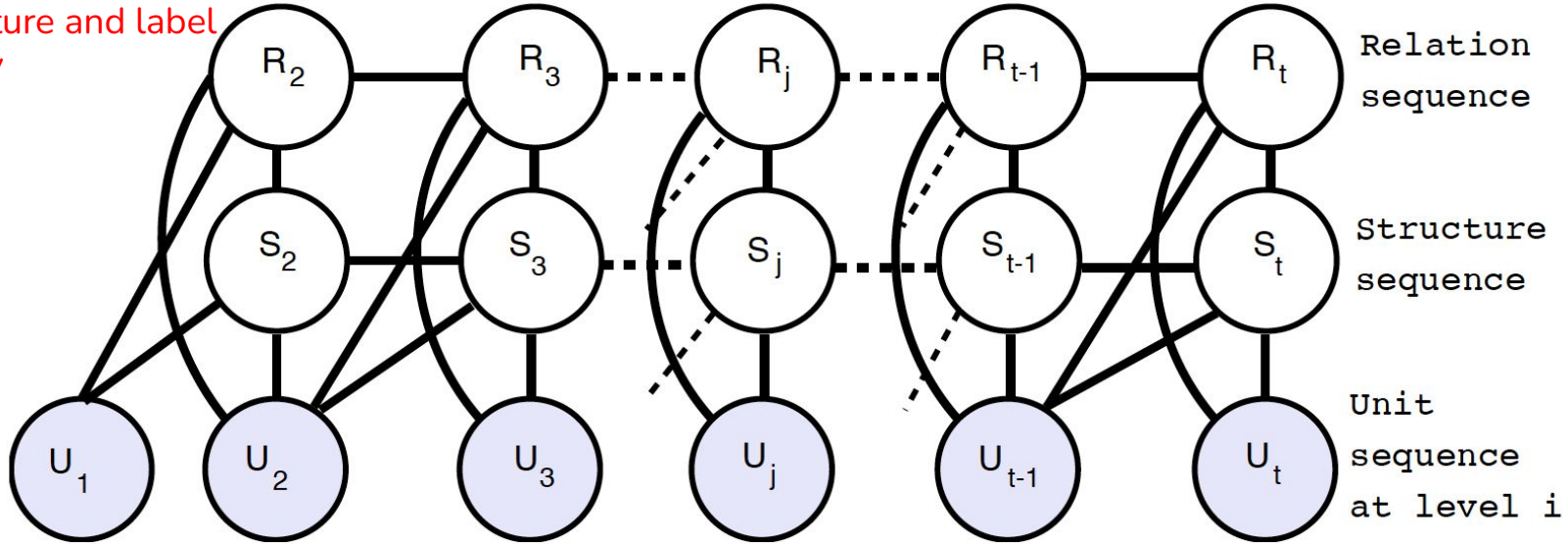


Figure 5
The intra-sentential parsing model of CODRA.

Intra-Sentential Parser

The connections between adjacent nodes in a hidden layer encode sequential dependencies between the respective hidden nodes. The connections between the two hidden layers model the structure and the relation of DT constituents jointly.

$$P(R_{2:t}, S_{2:t}|\mathrm{x}, \Theta_s) = \frac{1}{Z(\mathrm{x}, \Theta_s)} \prod_{i=2}^{t-1} \phi(R_i, R_{i+1}|\mathrm{x}, \Theta_{s,r}) \psi(S_i, S_{i+1}|\mathrm{x}, \Theta_{s,s}) \omega(R_i, S_i|\mathrm{x}, \Theta_{s,c})$$

Intra-Sentential Parser

$$\phi(R_i, R_{i+1}|\mathrm{x}, \Theta_{s,r}) = \exp(\Theta_{s,r}^T f(R_i, R_{i+1}, \mathrm{x}))$$

$$\psi(S_i, S_{i+1}|\mathrm{x}, \Theta_{s,s}) = \exp(\Theta_{s,s}^T f(S_i, S_{i+1}, \mathrm{x}))$$

$$\omega(R_i, S_i|\mathrm{x}, \Theta_{s,c}) = \exp(\Theta_{s,c}^T f(R_i, S_i, \mathrm{x}))$$

Intra-Sentential Parser

$$\phi(R_i, R_{i+1}|\mathrm{x}, \Theta_{s,r}) = \exp(\Theta_{s,r}^T f(R_i, R_{i+1}, \mathrm{x}))$$

$$\psi(S_i, S_{i+1}|\mathrm{x}, \Theta_{s,s}) = \exp(\Theta_{s,s}^T f(S_i, S_{i+1}, \mathrm{x}))$$

$$\omega(R_i, S_i|\mathrm{x}, \Theta_{s,c}) = \exp(\Theta_{s,c}^T f(R_i, S_i, \mathrm{x}))$$

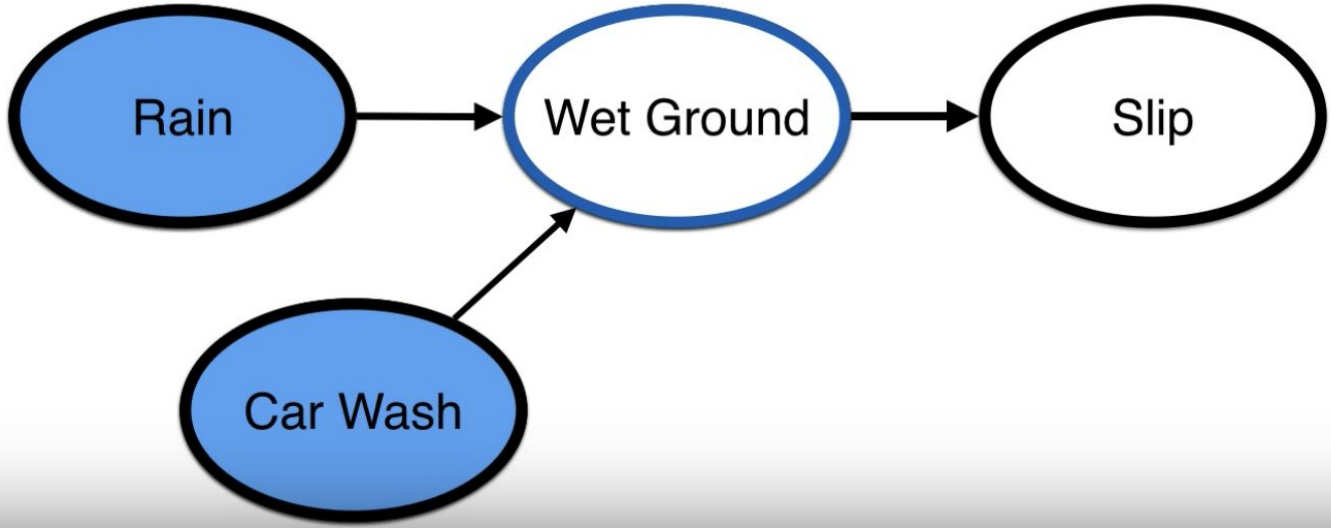
Q & A on CRF, HMM, DCRF, MRF

- Please ensure we do go over both DCRF and their “CKY-like bottom up parsing algorithm”.
- Can you clarify the difference between DCRFs and CRFs? It’s mentioned it’s a generalization but I’m having a hard time understanding it. Is it a CRF with more interconnected edges across generally not connect layers?
- Could you elaborate what is Dynamic Conditional Random Field (DCRF)?
- Could you please give more explanation of DCRF?
- Can you elaborate on DCRF?
- Could you explain the distinction between Markov Random Field and the Dynamic COnditional Random Field?
- Why are DCRF’s better than Hidden Markov Models?

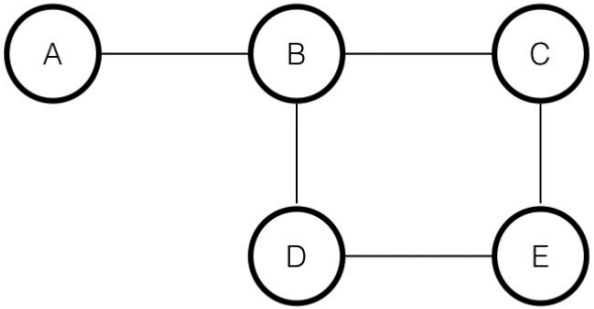
Review: Bayesian Networks

$P(R, W, S, C) = P(R)\, P(C)\, P(W \mid C, R)\, P(S \mid W)$

$P(X \mid \text{Parents}(X))$



Undirected Graphical Models

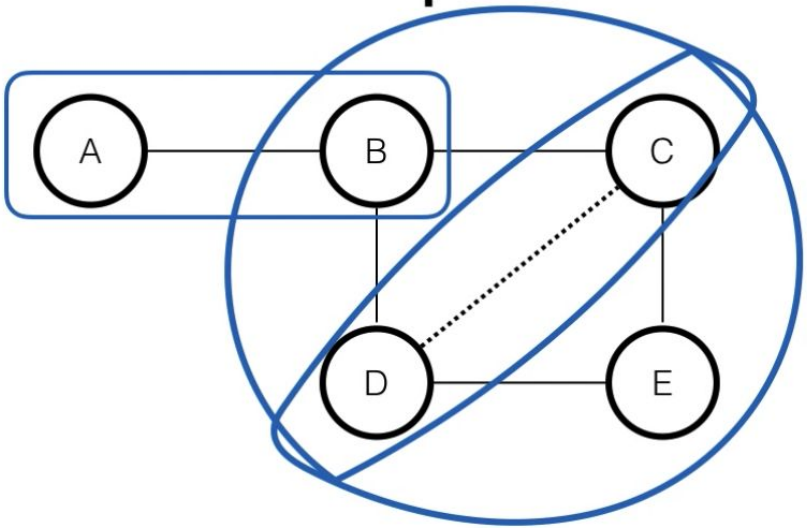


$$P(A, B, C, D, E) \propto \phi(A, B)\phi(B, C)\phi(B, D)\phi(C, E)\phi(D, E)$$

$$P(X) = \frac{1}{Z} \prod_{\text{cliques}(G)} \phi_C(x_c)$$

potential functions

Undirected Graphical Models



$$P(A, B, C, D, E) \propto \cancel{\phi(A, B)\phi(B, C)\phi(B, D)\phi(C, E)\phi(D, E)}$$

$$\phi(A, B)\phi(B, C, D)\phi(C, D, E)$$

$$P(X) = \frac{1}{Z} \prod_{c \in \text{cliques}(G)} \phi_C(x_c)$$

potential functions

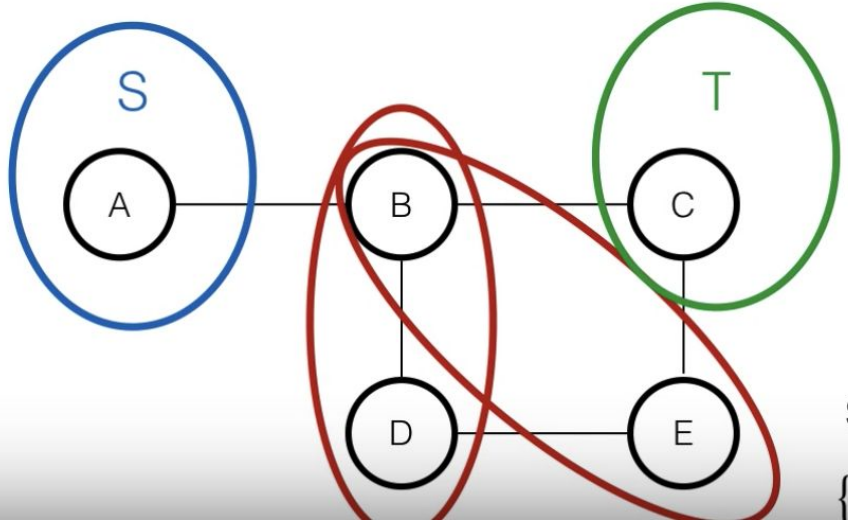
Markov Random Fields

- Any two subsets S and T of variables are conditionally independent given a **separating subset**
  - All paths between S and T must travel through the separating subset

paths:

A-B-C

A-B-D-E-C



separating subsets

{B,D}, {B,E}, {B,D,E}

Independence Corollaries

- Any two non-adjacent variables are conditionally independent given all other variables
- Any variable is conditionally independent of the other variables given its neighbors

Conditional Random Fields

Useful in many NLP and Vision tasks like POS tagging, Handwritten Text Recognition, Image Classification, Image Segmentation

Sequential natures of inputs need to be considered.

CRF in POS Tagging =>

In POS tagging, the goal is to label a sentence (a sequence of words or tokens) with tags like ADJECTIVE, NOUN, PREPOSITION, VERB, ADVERB, ARTICLE.

For example, given the sentence “Bob drank coffee at Starbucks”, the labeling might be “Bob (NOUN) drank (VERB) coffee (NOUN) at (PREPOSITION) Starbucks (NOUN)”.

Feature Functions in a CRF

In a CRF, each **feature function** is a function that takes in as input:

- a sentence s
- the position i of a word in the sentence
- the label l_i of the current word
- the label l_{i-1} of the previous word

and outputs a real-valued number (though the numbers are often just either 0 or 1).

(Note: by restricting our features to depend on only the current and previous labels, rather than arbitrary labels throughout the sentence, I'm actually building the special case of a **linear-chain CRF**. For simplicity, I'm going to ignore general CRFs in this post.)

For example, one possible feature function could measure how much we suspect that the current word should be labeled as an adjective given that the previous word is "very".

Features to Probabilities

Next, assign each feature function f_j a weight λ_j (I'll talk below about how to learn these weights from the data). Given a sentence s, we can now score a labeling l of s by adding up the weighted features over all words in the sentence:

$$score(l|s) = \sum_{j=1}^{m} \sum_{i=1}^{n} \lambda_j f_j(s, i, l_i, l_{i-1})$$

(The first sum runs over each feature function j, and the inner sum runs over each position i of the sentence.)

Finally, we can transform these scores into probabilities $p(l|s)$ between 0 and 1 by exponentiating and normalizing:

$$p(l|s) = \frac{exp[score(l|s)]}{\sum_{l'} exp[score(l'|s)]} = \frac{exp[\sum_{j=1}^{m} \sum_{i=1}^{n} \lambda_j f_j(s,i,l_i,l_{i-1})]}{\sum_{l'} exp[\sum_{j=1}^{m} \sum_{i=1}^{n} \lambda_j f_j(s,i,l'_i,l'_{i-1})]}$$

Example Feature Functions

So what do these feature functions look like? Examples of POS tagging features could include:

- $f_1(s, i, l_i, l_{i-1}) = 1$ if l_i = ADVERB and the ith word ends in "-ly"; 0 otherwise. ** If the weight λ_1 associated with this feature is large and positive, then this feature is essentially saying that we prefer labelings where words ending in -ly get labeled as ADVERB.

- $f_2(s, i, l_i, l_{i-1}) = 1$ if $i = 1$, l_i = VERB, and the sentence ends in a question mark; 0 otherwise. ** Again, if the weight λ_2 associated with this feature is large and positive, then labelings that assign VERB to the first word in a question (e.g., "Is this a sentence beginning with a verb?") are preferred.

- $f_3(s, i, l_i, l_{i-1}) = 1$ if l_{i-1} = ADJECTIVE and l_i = NOUN; 0 otherwise. ** Again, a positive weight for this feature means that adjectives tend to be followed by nouns.

- $f_4(s, i, l_i, l_{i-1}) = 1$ if l_{i-1} = PREPOSITION and l_i = PREPOSITION. ** A negative weight λ_4 for this function would mean that prepositions don't tend to follow prepositions, so we should avoid labelings where this happens.

And that's it! To sum up: to build a conditional random field, you just define a bunch of feature functions (which can depend on the entire sentence, a current position, and nearby labels), assign them weights, and add them all together, transforming at the end to a probability if necessary.

Recall that Hidden Markov Models are another model for part-of-speech tagging (and sequential labeling in general). Whereas CRFs throw any bunch of functions together to get a label score, HMMs take a generative approach to labeling, defining

$$p(l, s) = p(l_1) \prod_i p(l_i|l_{i-1}) p(w_i|l_i)$$

where

- $p(l_i|l_{i-1})$ are transition probabilities (e.g., the probability that a preposition is followed by a noun);
- $p(w_i|l_i)$ are emission probabilities (e.g., the probability that a noun emits the word "dad").

So how do HMMs compare to CRFs? CRFs are more powerful – they can model everything HMMs can and more. One way of seeing this is as follows.

Note that the log of the HMM probability is $\log p(l, s) = \log p(l_0) + \sum_i \log p(l_i|l_{i-1}) + \sum_i \log p(w_i|l_i)$. This has exactly the log-linear form of a CRF if we consider these log-probabilities to be the weights associated to binary transition and emission indicator features.

That is, we can build a CRF equivalent to any HMM by…

- For each HMM transition probability $p(l_i = y|l_{i-1} = x)$, define a set of CRF transition features of the form $f_{x,y}(s, i, l_i, l_{i-1}) = 1$ if $l_i = y$ and $l_{i-1} = x$. Give each feature a weight of $w_{x,y} = \log p(l_i = y|l_{i-1} = x)$.
- Similarly, for each HMM emission probability $p(w_i = z|l_i = x)$, define a set of CRF emission features of the form $g_{x,y}(s, i, l_i, l_{i-1}) = 1$ if $w_i = z$ and $l_i = x$. Give each feature a weight of $w_{x,z} = \log p(w_i = z|l_i = x)$.

Thus, the score $p(l|s)$ computed by a CRF using these feature functions is precisely proportional to the score computed by the associated HMM, and so every HMM is equivalent to some CRF.

However, CRFs can model a much richer set of label distributions as well, for two main reasons:

However, CRFs can model a much richer set of label distributions as well, for two main reasons:

- **CRFs can define a much larger set of features.** Whereas HMMs are necessarily local in nature (because they're constrained to binary transition and emission feature functions, which force each word to depend only on the current label and each label to depend only on the previous label), CRFs can use more global features. For example, one of the features in our POS tagger above increased the probability of labelings that tagged the first word of a sentence as a VERB if the end of the sentence contained a question mark.
- **CRFs can have arbitrary weights.** Whereas the probabilities of an HMM must satisfy certain constraints (e.g., $0 <= p(w_i|l_i) <= 1, \sum_w p(w_i = w|l_1) = 1$), the weights of a CRF are unrestricted (e.g., $\log p(w_i|l_i)$ can be anything it wants).

CRF vs DCRF

Dynamic CRFs (DCRFs), which are a generalization of linear-chain CRFs that repeat structure and parameters over a sequence of state vectors.

This allows to both represent distributed hidden state and complex interaction among labels, and to use rich, overlapping feature sets, as in conditional models.

Algorithm 1 Generating unit sequences for a sentence with n EDUs.

Input: Sequence of EDUs: $(e_1, e_2, \cdots, e_n)$
Output: List of sequences: L

```
for i = 1 → n − 1 do                                   // all possible starting positions for the subsequence
    for j = i + 1 → n do                               // all possible ending positions for the subsequence
        if j == n then                                 // sequences at top and bottom levels
            for k = i → j − 1 do                       // all possible cut points within the subsequence
                L.append ((e_1, ··· , e_{i−1}, e_{i:k}, e_{k+1:j}, e_{j+1}, ··· , e_n))
            end
        else                                           // sequences at intermediate levels
            for k = i + 1 → j − 1 do                   // cut points excluding duplicate sequences
                L.append ((e_1, ··· , e_{i−1}, e_{i:k}, e_{k+1:j}, e_{j+1}, ··· , e_n))
            end
        end
    end
end
```

TRAINING

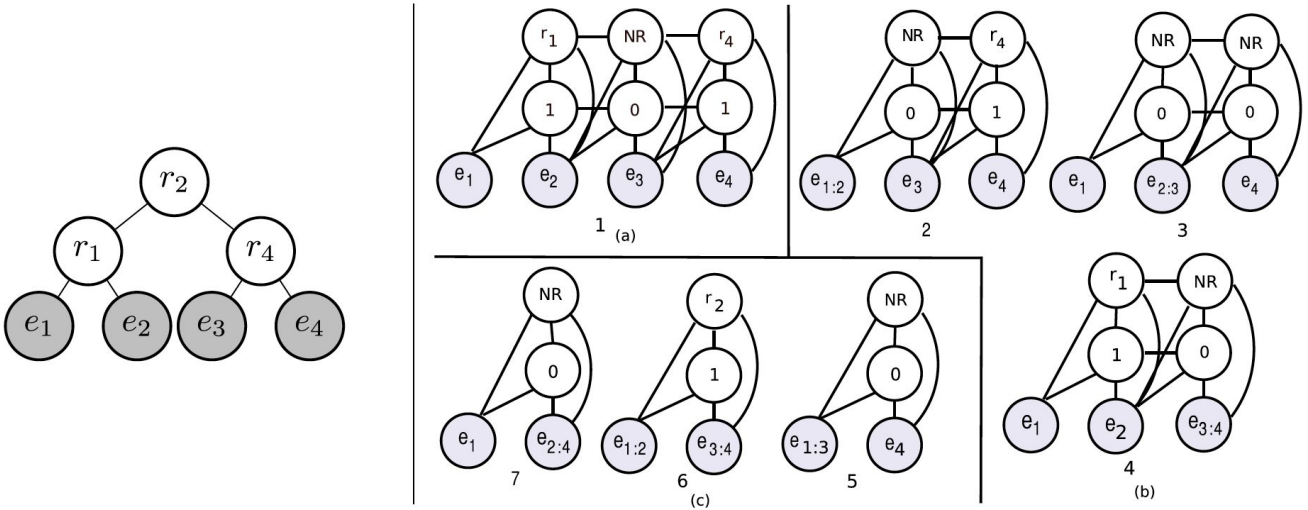


Figure 7
A gold discourse tree (left), and the 7 training instances it generates (right). NR = No Relation.

Training contd.

- From the gold DT, check if two discourse units are connected by a relation r
- train the model by maximizing the conditional likelihood of the labels in each of these training examples

Multi-Sentential Parsing Model

Simple Approach:

Given the discourse units (sub-trees) for all the individual sentences in a document, we can apply a new DCRF model, similar to the one to all the possible sequences generated from these units by Algorithm 1 to infer the probability of all possible higher-order (multi-sentential) constituents.

Multi-Sentential Parsing Model

- Algorithm 1 generates $O(n^3)$ sequences.
- Forward–backward on a sequence containing T units costs $O(TM^2)$ time, where M is the number of relations in relation set.
- learning requires running inference on every training sequence with an overall time complexity of $O(TM^2n^3)$ per document

Multi Sentential Parse Model

The two observed nodes U_t-1 and Ut are two adjacent (multi-sentential) discourse units.

The (hidden) structure node S ϵ {0, 1} denotes whether the two discourse units should be linked or not.

The other hidden node R ϵ {1,.....,M} represents the relation between the two units.

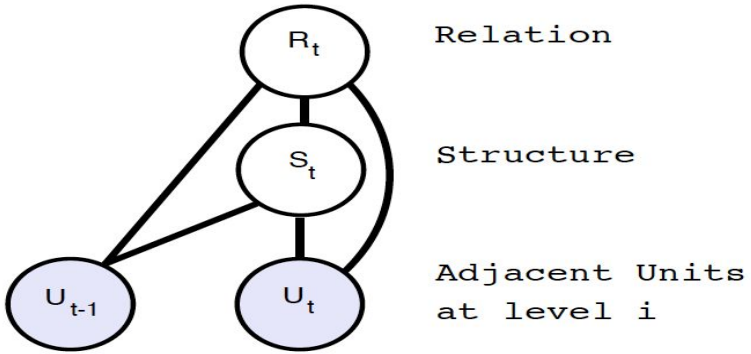


Figure 8
The multi-sentential parsing model of CODRA.

Feature Set

Organizational Features => the length of the discourse units as the number of EDUs and tokens in it using relative number

Text structural features => capture the correlation between text structure and rhetorical structure by counting the number of sentence and paragraph boundaries in the discourse units

Feature Set

N gram Feature Set =>

A lexical N-gram dictionary empirically from the training corpus,

first and last N tokens (1, 2, 3) of each discourse unit and rank them

according to their mutual information with the two labels, Structure (S) and Relation (R).

Table 1
Features used in our intra- and multi-sentential parsing models.

8 Organizational features	***Intra & Multi-Sentential***
Number of EDUs in *unit 1* (or *unit 2*).	
Number of tokens in *unit 1* (or *unit 2*).	
Distance of unit 1 in EDUs to the *beginning* (or to the *end*).	
Distance of unit 2 in EDUs to the *beginning* (or to the *end*).	
4 Text structural features	***Multi-Sentential***
Number of sentences in *unit 1* (or *unit 2*).	
Number of paragraphs in *unit 1* (or *unit 2*).	
8 N-gram features $N \in \{1, 2, 3\}$	***Intra & Multi-Sentential***
Beginning (or *end*) lexical N-grams in unit 1.	
Beginning (or *end*) lexical N-grams in unit 2.	
Beginning (or *end*) POS N-grams in unit 1.	
Beginning (or *end*) POS N-grams in unit 2.	

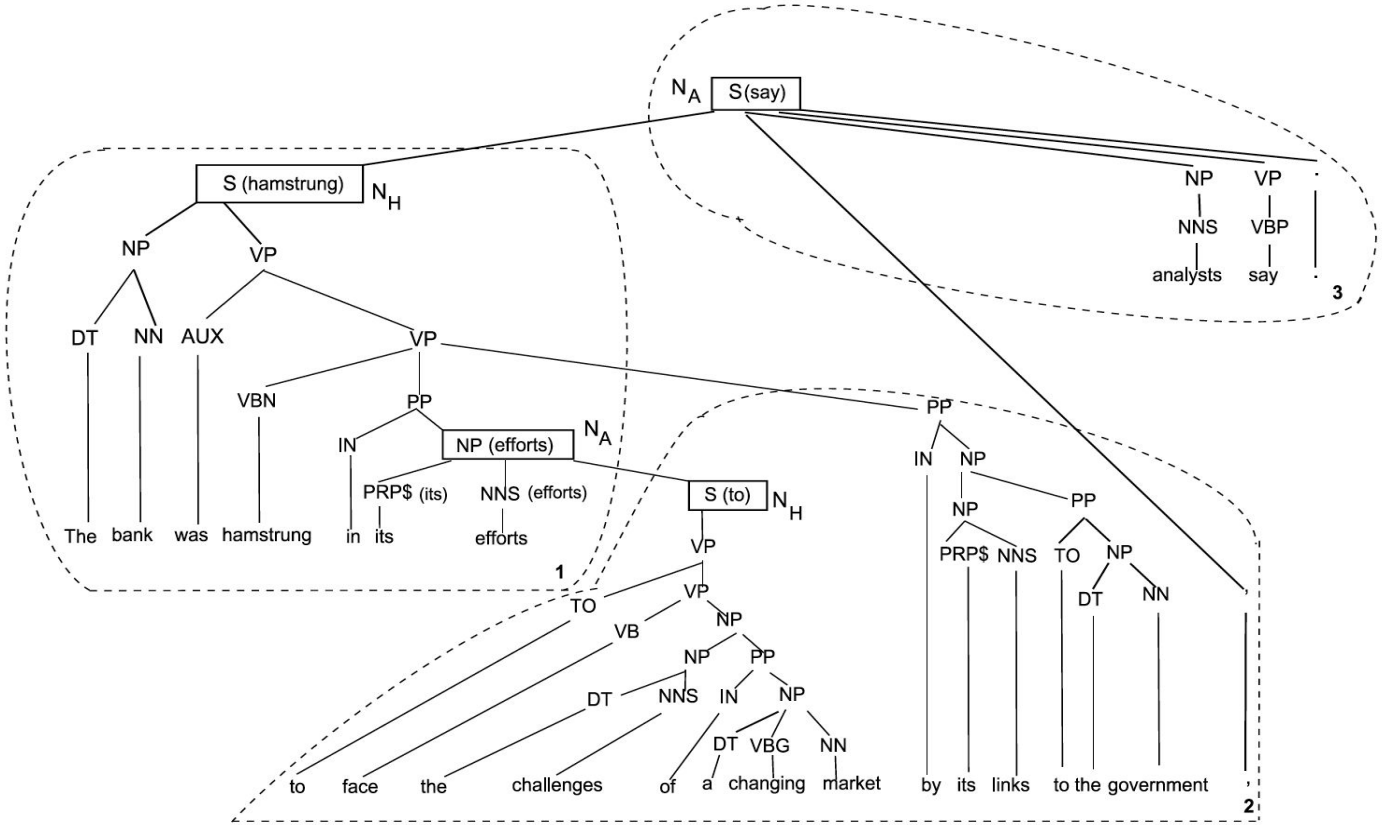
5 Dominance set features	***Intra-Sentential***
Syntactic labels of the *head* node and the *attachment* node. Lexical heads of the *head* node and the *attachment* node. *Dominance relationship* between the two units.	
9 Lexical chain features	***Multi-Sentential***
Number of chains spanning unit 1 and unit 2. Number of chains start in unit 1 and end in unit 2. Number of chains *start* (or *end*) in *unit 1* (or in *unit 2*). Number of chains skipping both unit 1 and unit 2. Number of chains skipping *unit 1* (or *unit 2*).	
2 Contextual features	***Intra & Multi-Sentential***
Previous and *next* feature vectors.	
2 Sub-structural features	***Intra & Multi-Sentential***
Root nodes of the *left* and *right* rhetorical sub-trees.	

Feature Set

Lexico-syntactic features =>

the Discourse Segmented Lexicalized Syntactic Tree (DS-LST)

- In a DS-LST, each EDU except the one with the root node must have a head node NH that is attached to an attachment node NA residing in a separate EDU.
- A dominance set D contains these attachment points of the EDUs in a DS-LST.
- In addition to the syntactic and lexical information of the head and attachment nodes, each element in the dominance set also includes a dominance relationship between the EDUs involved; the EDU with the attachment node dominates (represented by “> ”) the EDU with the head node.



D = { ((1, efforts/NP) > (2, to/S)), ((3, say/S) > (1, hamstrung/S)) }

(a) The discourse segmented lexicalized syntactic tree (DS-LST) for a sentence in RST–DT. Boxed nodes form the dominance set D as shown at the bottom.

Feature Set

Lexical Chain => sequences of semantically related words that can indicate topical boundaries in a text.

Useful for =>

- Semantic Analysis
- Topic Modeling
- Text Summarization

Feature Set

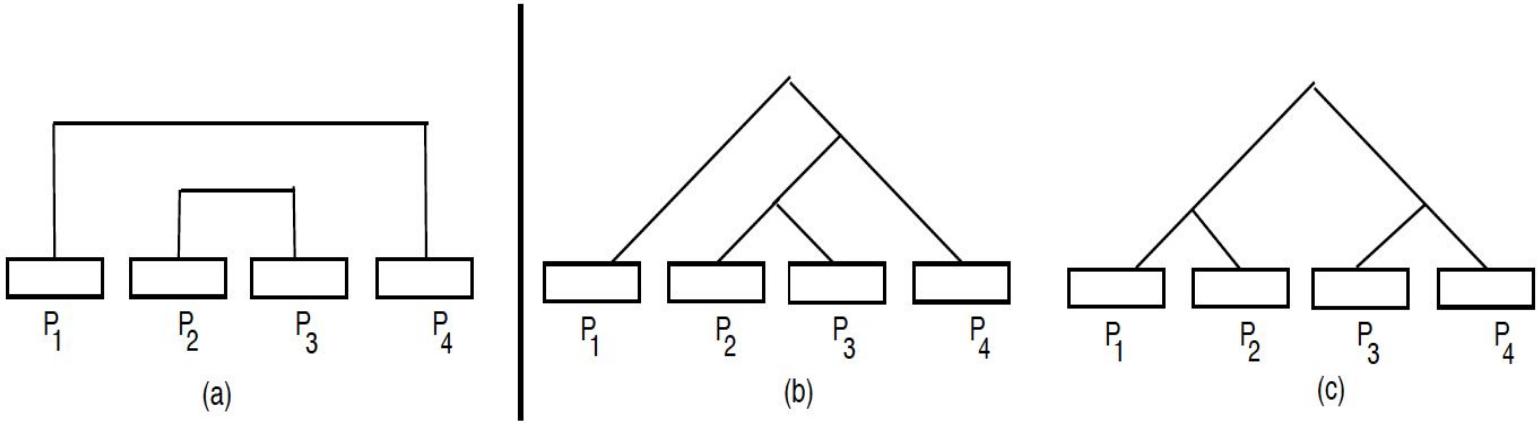


Figure 10
Correlation between lexical chains and discourse structure. (a) Lexical chains spanning paragraphs. (b) and (c) Two possible DT structures.

Feature Set (Lexical Chain)

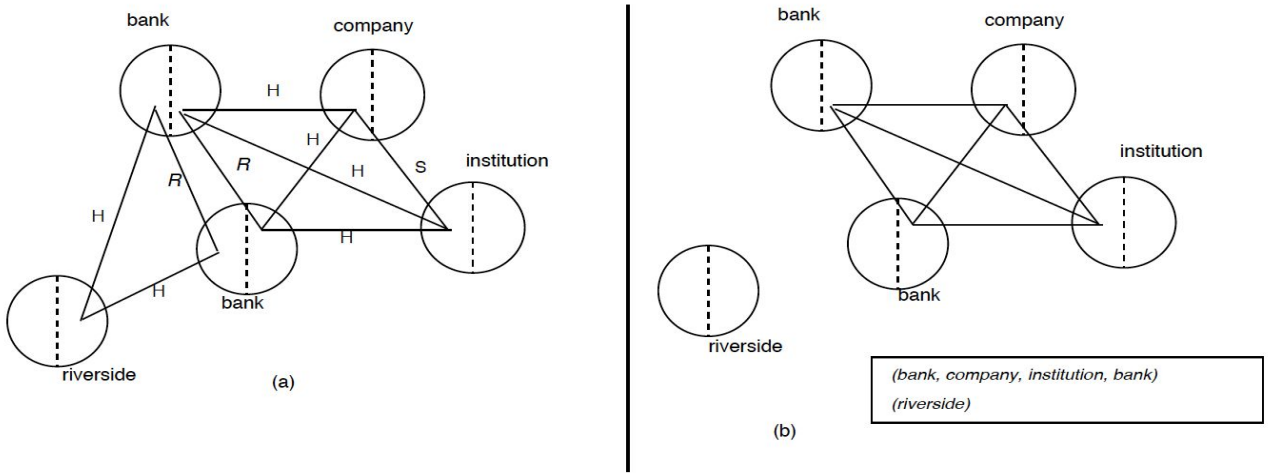


Figure 11
Extracting lexical chains. (a) A Lexical Semantic Relatedness Graph (LSRG) for five noun-tokens. (b) Resultant graph after performing WSD. The box at the bottom shows the lexical chains.

Parsing Algorithm

Explanation of CKY Algorithm => Roger Levy Slide

$$D[i,j] = P(r^*[U_i(0), U_{m^*}(1), U_j(1)])$$

where $U_x(0)$ and $U_x(1)$ are the start and end EDU Ids of discourse unit U_x, and

$$(m^*, r^*) = \underset{i \leq m < j \; ; \; R \in \{1 \cdots M\}}{\operatorname{argmax}} P(R[U_i(0), U_m(1), U_j(1)]) \times D[i,m] \times D[m+1,j]$$

Parsing Algorithm

Two Extra Tables =>

S and R for storing the structure (U_m (*)) and the relations (i.e., r*) of the corresponding DT constituents, respectively.

Parsing Algorithm

	1	1	2
		2	2
			3

S

	r_1	r_3	r_2
		r_2	r_3
			r_4

R

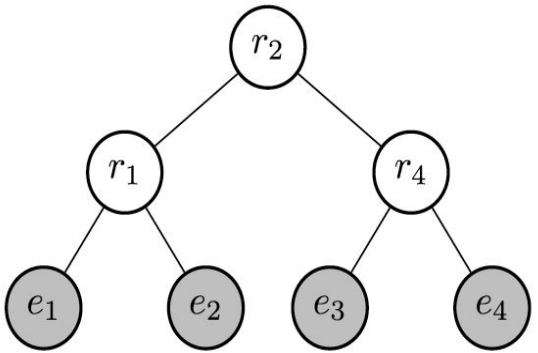


Figure 12
The S and R dynamic programming tables (left), and the corresponding discourse tree (right).

Document Level Parsing Approaches

1S–1S (1 Sentence–1 Sub-tree) =>

- constructs a DT for every sentence using intra-sentential parser
- and then it provides multi-sentential parser with the sentence-level DTs to build the rhetorical parse for the whole document.

1s-1s Approach

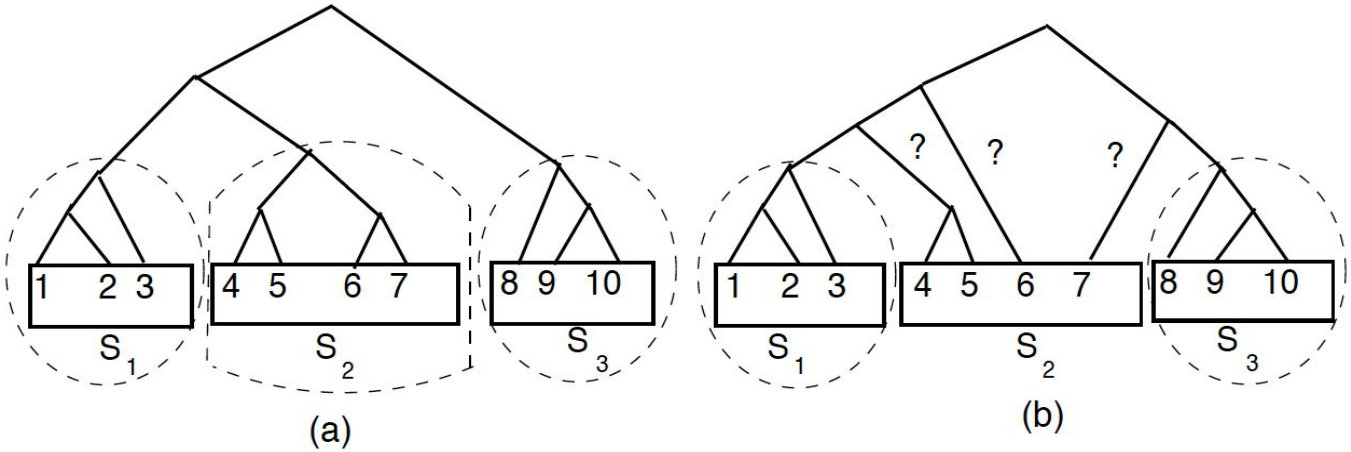


Figure 13
Two possible DTs for three sentences.

Sliding Window Approach

5% of the sentences in the RST–DT have leaky boundaries, In most cases (> 75%) where DT structures violate sentence boundaries, its units are merged with the units of its adjacent sentences.

Intra-sentential parser works with a window of two consecutive sentences, and builds a DT for the two sentences. For example, given the three sentences in Figure 13, our intra-sentential parser constructs a DT for S1–S2 and a DT for S2–S3.

Sliding Window Approach

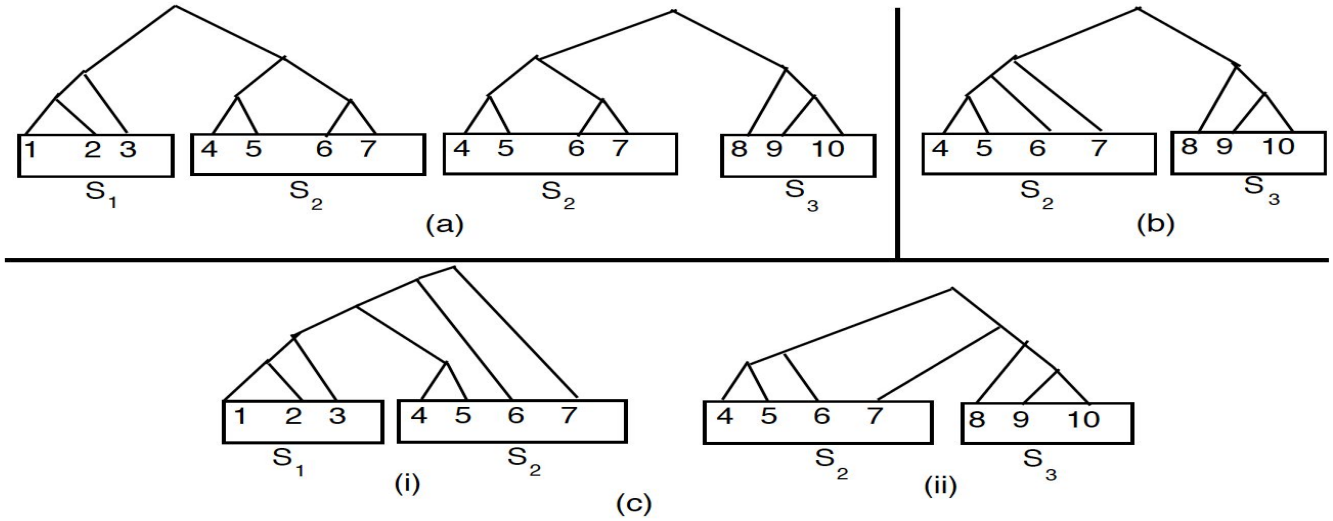


Figure 14
Extracting sub-trees for S_2.

Performances

Table 4
Intra-sentential parsing results based on manual discourse segmentation. Performances significantly superior to SPADE are denoted by *.

	RST–DT					Instructional	
	Standard Test Set			10-fold	Doubly	Reported	10-fold
Scores	SPADE	CRF-NC	PAR-S	PAR-S	Human	ILP	PAR-S
Span	93.5	95.1*	**96.5***	95.4	95.7	92.9	**98.3**
Nuclearity	85.8	87.7*	**89.4***	88.6	90.4	71.8	**89.4**
Relation	67.6	76.6*	**79.8***	78.9	83.0	63.0	**75.8**

Performances

Table 5
Intra-sentential parsing results using automatic discourse segmentation. Performances significantly superior to SPADE are denoted by *.

	RST–DT									Instructional		
	Test set						10-fold			10-fold		
Scores	SPADE			PAR-S			PAR-S			PAR-S		
	P	R	F	P	R	F	P	R	F	P	R	F
Span	75.9	77.4	76.7	**80.8***	**84.0***	**82.4***	79.6	80.7	80.1	73.5	80.7	76.9
Nuclearity	69.8	70.5	70.2	**75.2***	**78.1***	**76.6***	73.9	76.5	75.2	64.6	71.0	67.6
Relation	57.4	58.5	58.0	**66.1***	**68.8***	**67.5***	65.2	67.4	66.8	54.8	60.4	57.5

Performances

Table 6
Parsing results of document-level parsers using manual segmentation. Performances significantly superior to HILDA (p-value <0.0001) are denoted by *. Significant differences between TSP 1-1 and TSP SW (p-value <0.01) are denoted by †.

	RST–DT						Instructional		
Metrics	HILDA	CRF-O	CRF-T	TSP 1-1	TSP SW	Human	ILP	TSP 1-1	TSP SW
Span	74.68	77.02*	81.34*	**82.56***	**83.84*†**	88.70	70.35	**80.67**	**82.88†**
Nuc.	58.99	63.84*	66.52*	**68.32***	**68.90***	77.72	49.47	**63.03**	**64.13**
Rel.	44.32	48.46*	53.01*	**55.83***	**55.87***	65.75	35.44	**43.52**	**44.20**

Performances

Table 7
Oracle scores as a function of k of k-best sentence-level parses on RST–DT.

k	1	2	3	4	5	10	15	20	25	30
PAR-S	79.77	84.42	86.55	87.68	88.09	90.37	91.74	92.57	92.95	93.22

Performances

Table 8
Oracle scores as a function of k of k-best document-level parses on RST–DT.

k	1	2	3	4	5	10	15	20	25	30
TSP 1S-1S	55.83	56.52	56.67	56.80	56.91	57.23	57.54	57.65	57.67	57.74

Q & A

1. In terms of performance, how scalable is the work to much larger documents? The paper mentions that by taking into account both the "intra" and the "inter" sentence information, the system is scalable, but it does not go into significant detail about this.

2. The paper mentions the usage of conditional random fields for the parsing and inference part - and that the forward-backward algorithm was used for exact inference. This might be potentially computationally expensive - would neural models be useful here?

3. The paper mentions that segmentation and parsing are two separate stages. Would it be possible to combine them in some way? If the incoming text is not given as a solid document but in an "online" way, we might not be able to do both of these tasks separately.

Q & A

- Can we go over how the multi-sentential parsing is able to break the O(M^2) complexity?
- Can we touch on performance? I don't think it's part of the assigned sections, but I'm curious
- Why do they use "two" hidden layers in DCRF, not other deep belief networks or deep neural networks?
- In the feature part, it seems they only used the "beginning/end" N-grams in a unit, would they miss information about a sentence?
- Are there any other application of lexical chains?
- Just curious, what's the difference between the term "rhetorical analysis" and "discourse analysis"?

Q & A

1. What is CKY parsing algorithm?
2. Just curious to know what are state-of-the-arts model.
3. Do we have a neural model for discourse parsing? Does it do better ?

Q & A

- Can you explain why the parser uses separate parsing models for intra-sentential and inter-sentimental parsers but the same parsing algorithm unit? Also, on topic of the parsing models, could we leverage neural architectures to learn better parameters and potentially improve performance?
- Is the binary discourse tree structure assumption a valid assumption? Could there be cases where this is not true and we cannot fix it by a cross-over onto a multi-nested binary tree?
- Based on personal experience, EDU generation of sentences with in proper conjugations (such as .., or improper parentheses) often fails using CODRA. While this is a problem that lies on the EDU parser, how do you think this problem could be potentially tackled?

Q & A

2. How do CRFs surmount the label bias problem of the Maximum Entropy Markov Model?

3. What is the overall time complexity of this CODRA framework?

4. Could you please explain how to apply rhetorical structure on text summarization?

5. Is there any neural model related to this task?

Q & A

I wonder if we could formulate building a full DT for a whole document such that we could use the HAN architecture from last week's paper. Since the authors approach seems to be using a hierarchical structure (Intra-sentential parser à multi0sentential parser à DT for whole document.)

How were rhetorical relations (53 mononuclear and 25 multinuclear) originally determined? Would it be possible to now try and learn these from some sort of bottom up data driven approach?

Q & A

- Could we use a single parser for both intra-sentential and inter-sentimental parsing?
- What is the current state-of-the-art method on discourse analysis?
- What is the runtime of the proposed method?
- What is the label bias problem?

Q & A

1. Why did existing discourse parsers use a sub-optimal parsing algorithm to be the discourse tree?
2. Did those approaches possibly, trade optimality for speed?
3. Could explain what are dominance sets and their role in this paper?