# Discourse Analysis of Asynchronous Conversations

by

Shafiq Rayhan Joty

B. Sc., Islamic University of Technology, 2005

M. Sc., University of Lethbridge, 2008

A THESIS SUBMITTED IN PARTIAL FULFILLMENT
OF THE REQUIREMENTS FOR THE DEGREE OF

**Doctor of Philosophy**

in

THE FACULTY OF GRADUATE AND POSTDOCTORAL STUDIES

(Computer Science)

The University Of British Columbia

(Vancouver)

December 2013

# Abstract

A well-written text is not merely a sequence of independent and isolated sentences, but instead a sequence of structured and related sentences. It addresses a particular topic, often covering multiple subtopics, and is organized in a coherent way that enables the reader to process the information. Discourse analysis seeks to uncover such underlying structures, which can support many applications including text summarization and information extraction.

This thesis focuses on building novel computational models of different discourse analysis tasks in asynchronous conversations; i.e., conversations where participants communicate with each other at different times (e.g., emails, blogs). Effective processing of these conversations can be of great strategic value for both organizations and individuals. We propose novel computational models for topic segmentation and labeling, rhetorical parsing and dialog act recognition in asynchronous conversation. Our approaches rely on two related computational methodologies: graph theory and probabilistic graphical models.

The topic segmentation and labeling models find the high-level discourse structure; i.e., the global topical structure of an asynchronous conversation. Our graph-based approach extends state-of-the-art methods by integrating a fine-grained conversational structure with other conversational features.

On the other hand, the rhetorical parser captures the coherence structure, a finer discourse structure, by identifying coherence relations between the discourse units within each comment of the conversation. Our parser applies an optimal parsing algorithm to probabilities inferred from a discriminative graphical model which allows us to represent the structure and the label of a discourse tree constituent jointly, and to capture the sequential and hierarchical dependencies between the

constituents.

Finally, the dialog act model allows us to uncover the underlying dialog structure of the conversation. We present unsupervised probabilistic graphical models that capture the sequential dependencies between the acts, and show how these models can be trained more effectively based on the fine-grained conversational structure.

Together, these structures provide a deep understanding of an asynchronous conversation that can be exploited in the above-mentioned applications. For each discourse processing task, we evaluate our approach on different datasets, and show that our models consistently outperform the state-of-the-art by a wide margin. Often our results are highly correlated with human annotations.

# Preface

Portions of this thesis are based on prior peer-reviewed publications by me (under the name Shafiq Joty) and my collaborators. In the following, I describe the publications and the relative contributions of all contributors.

Chapter 2 is based on the article *Topic Segmentation and Labeling in Asynchronous Conversations* by Shafiq Joty, Giuseppe Carenini and Raymond Ng, published in the Journal of Artificial Intelligence Research (JAIR), volume 47, June 2013. Portions of this work were previously published in two conference proceedings: (a) *Exploiting Conversation Structure in Unsupervised Topic Segmentation for Emails* by Shafiq Joty, Giuseppe Carenini, Gabriel Murray and Raymond Ng, in the proceedings of EMNLP, 2010; (b) *Supervised Topic Segmentation of Email Conversations* by Shafiq Joty, Giuseppe Carenini, Gabriel Murray and Raymond Ng, in the proceedings of ICWSM, 2011. My main contributions include: 1) designing and performing an user study to create two new corpora of asynchronous conversations (i.e., email and blog) annotated with topic information; 2) proposing and implementing novel topic segmentation and labeling models for asynchronous conversations; 3) implementing metrics to measure the performance of the computational models and inter-annotator agreement on the topic segmentation and labeling tasks; 4) carrying out the experiments on the two developed corpora; 5) preparing the manuscripts. Other collaborators played supervisory roles.

Chapter 3 is based on two conference papers: (a) *A Novel Discriminative Framework for Sentence-Level Discourse Analysis* by Shafiq Joty, Giuseppe Carenini and Raymond Ng, published in the proceedings of EMNLP-CoNLL, 2012; (b) *Combining Intra- and Multi-Sentential Rhetorical Parsing for Document-Level Discourse Analysis* by Shafiq Joty, Giuseppe Carenini, Raymond Ng and Yashar

Mehdad, published in the proceedings of ACL, 2013. My main contributions include: 1) preparing the datasets (i.e., RST-DT and Instructional) for experiments on discourse segmentation, sentence-level discourse parsing and document-level discourse parsing; 2) proposing and implementing the computational models for discourse segmentation, sentence-level discourse parsing and document-level discourse parsing; 3) implementing the metrics used to measure the performance of the computational models and inter-annotator agreement on the discourse segmentation and parsing tasks; 4) carrying out the experiments; 5) preparing the manuscripts. Other collaborators played supervisory roles throughout the project.

Chapter 4 is based on the conference paper *Unsupervised Modeling of Dialog Acts in Asynchronous Conversations* by Shafiq Joty, Giuseppe Carenini and Chin-Yew Lin, published in the proceedings of IJCAI, 2011. My main contributions include: 1) preparing the datasets (i.e., W3C email and TropAdvisor forum) by removing noise (e.g., unnecessary system messages); 2) proposing and implementing the computational models for dialog act recognition; 3) carrying out the experiments; 4) preparing the manuscript. Other collaborators advised me.

# Table of Contents

# List of Tables

# List of Figures

# Acknowledgments

First and foremost, I thank Allah, the almighty, for giving me the strength to carry on my post graduate studies and for blessing me with many great people who have been my greatest support in both my professional and personal life.

This thesis is the culmination of an exciting five-year adventure. For that I'd like to thank my advisor Dr. Giuseppe Carenini, whose door was always open for me. I'm eternally grateful to him for being a great teacher and mentor, for inspiring me to explore new ideas and for helping me refine my incoherent ideas. My co-advisor, Dr. Raymond Ng, shared his wealth of experience and guided me to be better at all aspects of being a researcher. I really enjoyed my Ph.D. experience at UBC.

I'd also like to thank Dr. Nando De Freitas for serving on my supervisory committee, my university examiners Dr. Rachel Pottinger and Dr. Carson Woo, and my external examiners Dr. Amanda Stent and Dr. Carolyn Rose for their insightful comments on my thesis.

I'm indebted to my friend Jackie Cheung from the University of Toronto and to the fellow members of the NLP group at UBC, Gabriel Murray, Yashar Mehdad, Shima Gerani and Shama Rashid. The conversations, ideas, feedback and support from them have been invaluable over the years.

I'm also very grateful to NSERC Canada Graduate Scholarship (CGS-D) and NSERC BIN Strategic Network for their funding support during my Ph.D. studies.

Finally, thanks to my parents Mr. and Mrs. Zaman and my sister Sonia Arju for their unconditional love and support. Many thanks to Sumaiya Sabira, my beloved wife, for always being there with me.

# Chapter 1

# Introduction

With the ever increasing popularity of Internet technologies, it is very common nowadays for people to discuss events, issues, tasks and personal experiences by writing in a growing number of social media including emails, blogs, Facebook and Twitter [16, 216]. These are examples of **written asynchronous conversations** where participants communicate with each other at *different* times.

The huge amount of textual data generated every day in these conversations calls for automated methods of conversational text analysis. Effective processing of these conversational texts can be of great strategic value for both organizations and individuals [36]. For instance, conversations in public blogging services like Twitter have become invaluable sources of information. During a natural disaster like Hurricane Sandy, affected people sent tweets to request food, shelter, medicine etc. In response to that, many people and organizations also tweeted to offer help. Humanitarian organizations (e.g., the United Nations, government officials) can mine these tweets to carry out aid activities more effectively [214].

In the very different scenario of business intelligence, corporate managers might find the information exchanged in their email conversations and company blogs to be extremely useful for decision auditing. If a decision turns out to be favorable, mining the relevant conversations may help in identifying effective communication patterns and sources. Similarly, conversations that led to ill-advised decisions could be mined to determine responsibility and accountability. To compete in markets, business users are now facing an unprecedented need to effectively process

the online conversations taking place in various social media [217]. They need to uncover not just keywords, but vital consumer sentiment and insights about their company, products, competitors and more.

Conversations in public blogs (e.g., Twitter, Slashdot[1]) often get very large, containing overall thousands of comments. During major events such as a political uprising in the Arab world, relevant messages are posted by the thousands or millions. It is simply not feasible to read all messages relevant to such an event, and so mining and summarization technologies can help to provide an overview of what people are saying and what positive or negative opinions are being expressed. Mining social media can reveal a nation's perspectives on national events and issues (e.g., presidential elections, abortion rights). Mining and summarizing also improve indexing and searching [119].

On a more personal level, an informative summary of a conversation could greatly support a new participant to get up to speed and join an already existing conversation. It could also help someone to quickly prepare for a follow-up discussion of a conversation she was already part of, but which occurred too long ago for her to remember the details.

Although the number of applications targeting conversations in social media is growing, most of these applications are not currently using sophisticated Natural Language Processing (NLP) techniques. There could be two reasons for this. First, although NLP technologies like syntactic parsing and part-of-speech (POS) tagging have attained performances close to that of humans, many others (e.g., discourse parsing, word sense disambiguation) are still far below the human standard, and are not sufficiently accurate to support downstream applications. Second, most of these technologies were originally developed for monologs (e.g., news articles) and are not as effective when applied directly to asynchronous conversations because the two types of genres are different in many aspects.

This thesis focuses on building novel computational models of several **discourse analysis** tasks in asynchronous conversations, which can support a wide range of NLP applications including conversation summarization, conversation visualization, sentiment analysis and information extraction [226]. In particular, we

[1] http://slashdot.org/

propose novel models for **topic segmentation and labeling, discourse (rhetorical) parsing** and **dialog act recognition** in asynchronous conversations. We do so by extending the monolog-based models to consider the conversation specific features of asynchronous conversations and by addressing the key limitations of the existing models to improve them further. Our approaches rely on two related computational methodologies: recent **graph-theoretic methods for NLP** [139] and advanced **probabilistic graphical models** [108].

Although discourse analysis has been well-studied in monolog and in synchronous dialog (e.g., meetings), a comprehensive study of discourse analysis in asynchronous conversation is still missing. For many discourse analysis tasks on asynchronous conversations (e.g., topic segmentation and labeling), there are no standard corpora, no annotation guidelines and no established evaluation or agreement metrics available. Also, since asynchronous conversations are quite different from monologs and synchronous dialogs, discourse analysis models that are successful in monologs or in synchronous dialogs may not be as effective when applied directly to asynchronous conversations. In this thesis, we overcome these limitations by developing new annotated corpora, and by proposing new computational models for discourse analysis in asynchronous conversation. Additionally, in the generic task of discourse parsing where the performance of existing systems is still far away from the human standard, we substantially reduce the performance gap by addressing their key limitations.

Like any other discourse, an asynchronous conversation discusses a common topic, often covering multiple subtopics. In addition, each message in a conversation locally constitutes a coherent monolog by connecting its sentences logically; i.e., the meaning of a sentence relates to the previous ones. Furthermore, being a conversation, asynchronous conversation exhibits a conversational structure.

The topic segmentation and labeling models find the high-level discourse structure; i.e., the global topical structure of an asynchronous conversation. Our unsupervised approach to topic segmentation extends state-of-the-art models by considering a fine-grained structure of the conversation. Our supervised topic segmentation model combines lexical, conversational and topic related features in a graph-theoretic framework. Our (unsupervised) topic labeling models capture conversation specific clues in a graph random walk framework.

On the other hand, the discourse parser captures the local coherence structure, a finer discourse structure, by identifying coherence relations between the discourse units within each comment. Our parser applies an optimal parsing algorithm to probabilities inferred from a discriminative undirected graphical model, which represents the structure and the label of a discourse tree constituent jointly and captures the sequential and hierarchical dependencies between the constituents.

Finally, the dialog act model allows us to uncover the underlying dialog structure of the conversation. We present unsupervised generative graphical models that capture the sequential dependencies between the dialog acts, and show how these models can be trained more effectively based on the fine-grained conversational structure.

Together, these structures provide a deep understanding of asynchronous conversations that can be effectively exploited in the above-mentioned NLP applications. For each analysis task, we evaluate our approach on different datasets, and show that our models consistently outperform the state-of-the-art by a wide margin. Remarkably, our results are often highly correlated with human annotations.

In the rest of this introduction, in Section 1.1 we give an overview of the discourse analysis tasks in different forms of discourse, and identifies the technical challenges in performing these tasks on asynchronous conversations. In Section 1.2 we discuss the computational methodologies used to tackle the technical challenges. Finally, in Section 1.3 we summarize our key contributions with an outline of the dissertation.

## 1.1 Discourse and Its Structures

Although many NLP tasks treat texts at the sentence level (e.g., syntactic or semantic parsing [100]), sentences rarely stand on their own in an actual discourse; it is simply not enough to gather some arbitrary sentences to obtain a discourse. Rather, the relationship between sentences carry important information which allows the discourse to express a meaning as a whole beyond the sum of its separate parts.

Two complementary aspects of discourse work together to make it interpretable as a whole. The first aspect is **coherence**, which logically binds the sentences together — the meaning of a sentence is connected to the meaning of the previous

and the following ones. Without this, a text is just a sequence of non-sequiturs. For instance, consider the following two examples from [87]:

- John took a train from Paris to Istanbul. He has family there.
- John took a train from Paris to Istanbul. He likes spinach.

While it is easy to process the first text, most readers will have difficulties in understanding the second one. The reader will either reject the second text simply calling it *"incoherent"* or spend some time to construct an explanation of what liking spinach has to do with taking a train from Paris to Istanbul. By asking this, the reader is actually questioning the coherence of the text. The second aspect is **cohesion**, which is the usage of linguistic glue to link the textual units [79]. Cohesion is achieved by using word repetitions and semantically similar words, such as synonyms, hypernyms and hyponyms (known as **lexical cohesion**), as well as by using other linguistic devices including coreferences (see the examples above).

Table 1.1 shows examples of different forms of discourse we encounter in our daily life. The thesis you are reading is an example of a **monolog** where a writer (speaker) writes (speaks) something to be read (heard) by a reader (hearer). The flow of information in monologs is unidirectional; i.e., from the writer to the reader. After you finish reading the thesis, you may have a face-to-face (or phone, email) conversation with your colleague about it. The conversation involves interchanging roles between being a speaker (writer) and hearer (reader). This type of discourse is called a **dialog (or conversation)**. Unlike monologs, the communication flow in dialogs is bidirectional and participants perform different types of communicative acts: asking questions, giving answers, requesting something, and so forth. A dialog involving more than two participants is called a **multi-party conversation**.

Conversations can be further categorized into two groups: **synchronous** and **asynchronous**. Synchronous conversations are those where participants communicate with each other at the same time (e.g., phone conversations). Turns in synchronous conversations are usually short, containing only a few utterances. Before the Internet revolution, this type of conversation was the dominant form of human conversation. However, with the rise of the Internet and web technologies, people now converse by writing in a growing number of social media including emails,

| | Monolog | Dialog (Conversation) | |
|---|---|---|---|
| | | Synchronous | Asynchronous |
| Written | articles, books | Instant messaging, Internet relay chat | emails, blogs fora, Facebook |
| Spoken | lectures, talks, speeches, news broadcast | face-to-face dialog, meetings, phone conversations, human-computer dialog | voicemail video logs |

**Table 1.1:** Examples of different forms of discourse.

blogs, fora and so on. These are called asynchronous conversations because the communication happens at different times. In contrast to synchronous conversations, replies in asynchronous conversations can be made days later. The length of the replies varies from one to a few hundred sentences. For example, while tweets are limited to only 140 characters, comments in political and technology-related blogs (e.g., AMERICAblog[2], Slashdot) are much longer.

What are the structures in a discourse? Speakers in a discourse discuss a common topic, often covering multiple subtopics. For example, an email conversation about *arranging a conference* may discuss *conference schedule*, *organizing committee*, *accommodation*, *attractions*, *registration* and so on. In other words, a discourse has a *topical structure*. We define topic more precisely in the next section. In a discourse, speakers refer to something they have talked about before; i.e., a discourse uses *coreference* (e.g., anaphora) to refer to the same thing. In addition, the sentences in a discourse are logically connected: the meaning of a sentence relates to that of the previous ones; that is, a discourse has a *coherence structure*. Furthermore, in a conversational discourse, participants interact with each other performing different dialog acts (e.g., asking or answering questions). A conversation exhibits a *conversational structure*, which comprises the dialog acts and the reply structure (i.e., who is talking to whom in multi-party dialogs).

Understanding a discourse implies uncovering these structures. Table 1.2 summarizes the analysis tasks for different forms of discourse. This thesis focuses on building computational models to uncover the topical structure, the rhetorical struc-

[2]http://americablog.com

| | Topic Seg. & Labeling | Discourse parsing | Reference resolution | Dialog acts & Reply structure extraction |
|---|---|---|---|---|
| Monolog | ✓ | ✓ | ✓ | N/A |
| Sync. Conv. | ✓ | ✓ | ✓ | ✓ |
| Async. Conv. | ✓ | ✓ | ✓ | ✓ |

**Table 1.2:** Discourse analysis tasks in different forms of discourse.

ture and the dialog act structure of an asynchronous conversation. In the following, we give a general overview of these three structures of discourse, the technical challenges and the previous work in uncovering these structures. Readers interested in knowing about *co-reference resolution* are encouraged to see [198].

### 1.1.1 Topical Structure

Topical structure is the high-level structure of a discourse. According to Webber et al. [226], each topic comprises a set of entities and things being said about them. A topic can be characterized by the *subject matter* it addresses. For example, consider the following text from BBC[3].

- Mr. Obama faces a tough week of trying to persuade Congress to authorise military action in response to Syria's alleged use of chemical weapons. He will also seek public support for the action in a White House address on Tuesday before the Congress finally votes.

This paragraph addresses Mr. Obama's attempts to authorize a military action in Syria. Here the entities consist of Mr. Obama, Congress, military action, Syria, chemical weapon and White House. The set of entities usually changes from topic to topic. For example, consider the following paragraph from the same document but which appears later in the text.

- Russia restated its opposition to any strike at the G20 summit, with Mr Putin warning that military intervention would destabilise the region. Both Russia and China, which have refused to agree to a UN Security Council resolution against Syria, insist any military action without the UN would be illegal.

[3] www.bbc.co.uk/news/world-us-canada-23999066

This talks about the response from Russia and China on the army intervention. Here the entities are Russia, G20 summit, Mr. Putin, military intervention, China, UN Security Council resolution, Syria and UN.

Any discourse containing more than a few sentences is likely to cover multiple topics.[4] For example, Hearst [83] reports that about 40% of the paragraphs in her corpus of expository texts start with a new topic. Automatically determining the topic structure involves two subtasks: topic segmentation and topic labeling. In general, **topic segmentation** is the task of grouping the sentences of a discourse into a set of topical segments (or clusters). If the discourse is a monolog as in the above example or a synchronous dialog (e.g., meetings), then this task can be rephrased as: separating a discourse into a *sequence* of topical segments. However, we will see that because of the asynchronous nature, topics in *asynchronous conversations* often do not change in a sequential way.

Once we have the topical segments, **topic labeling** is the task of assigning a short informative description to each of the topical segments to facilitate interpretations of the topics [36]. For instance, ideal topic labels for the two texts above could be *Mr. Obama's attempts to authorize a military action in Syria* and *The response from Russia and China on the army intervention*.

Topic segmentation is often considered as an essential preprocessing step for other finer-level discourse analysis [14] and has been utilized in many NLP applications including text summarization [56, 104] and information extraction [3]. Topic labels provide a high-level concise summary of the discourse. However, both topic segmentation and labeling are considered to be difficult and unsolved problems. As mentioned before, as the topics become more fine-grained, topic segmentation and labeling could be a hard task even for humans. The difficulties also vary in different text domains. For example, dialogs pose a different set of challenges from monologs. Inside monologs, spoken monologs (e.g., talks) are more challenging than written edited monologs (e.g., articles), which typically come already organized in sections and paragraphs reflecting the topical structure [36].

[4] Considering that a discourse addresses a common topic, here 'topics' actually means 'subtopics'.

### Topic Segmentation

Previous work on topic segmentation targeted mainly monologs and synchronous dialogs [165]. In these two forms of discourse, topic segmentation refers to the task of separating the discourse into *sequential* topical segments. Several unsupervised and supervised methods have been proposed. Table 1.3 summarizes them. The unsupervised models exploit the strong correlation between topic and lexical usage. These models can be categorized into two broad classes based on their underlying intuitions: similarity-based models and probabilistic generative models.

| Category | Example models | Type | Features |
|---|---|---|---|
| Lexical similarity | TextTiling [83], C'01 [44] LCSeg [74], M&B [121] | Unsupervised | Lexical similarity |
| Generative models | LDAs [61, 167] HMMs [27, 233] | Unsupervised | Lexical distribution, cues, speaker |
| Feature based | Exponential [21] Decision tree [74] | Supervised | Lexical, cues, speaker, silence, etc. |

**Table 1.3:** Summary of existing work on topic segmentation.

The key intuition behind **similarity-based segmentation models** is that sentences in a segment are more lexically similar to each other than to sentences in the preceding or the following segment. The technical challenges addressed by this class of models are: (1) how to measure lexical similarity, and (2) how to use the similarity measures to perform topic segmentation.

A typical method to measure lexical similarity between two texts is to first represent each text as a vector, and then compute the cosine angle between the vectors. The existing approaches differ in how they represent the texts as vectors. For example, Hearst [83] uses term frequency (TF) in **TextTiling**, Malioutov and Barzilay [121] (**M&B** in Table 1.3) use TF and inverse document frequency (IDF) [178], Choi et al. [44] (**C'01** in the table) use latent semantic analysis (LSA), and Galley et al. [74] use a metric computed from lexical chains [143] in **LCSeg**.

Different models use different algorithms to perform segmentation using the lexical similarity scores. For example, Hearst [83] and Galley et al. [74] use a

cutoff threshold on the *similarity valley* (i.e., a plot of the lexical similarity scores), Choi et al. [44] use *divisive clustering* (also known as top-down clustering), and Malioutov and Barzilay [121] use minimum cut (graph-based) clustering.

Note that the similarity-based segmentation models only exploit the lexical cohesion phenomena of a discourse. However, a discourse can have other features that are useful for topic segmentation. For example, *cue phrases* provide domain-specific clues for topic shifts in broadcast news [21] (e.g., *coming up, joining us now*) and in meetings [74] (e.g., *anyway, so*). Later, we will see that *conversational structure* can be a very useful feature for segmenting asynchronous conversations.

**Probabilistic generative models** form another class of unsupervised segmentation models, which is based on the intuition that a discourse is a *hidden sequence* of topics, each of which has its own characteristic word distribution. The distribution changes with the change of a topic. Topic segmentation in these models is the task to infer the most likely sequence of topics given the observed words. Variants of **Hidden Markov Models (HMMs)** (e.g., [27, 233]) and **Latent Dirichlet Allocations (LDAs)** [26] (e.g., [61, 167]) are proposed. Although earlier generative models [26, 27, 167] are based on only lexical distributions, recent models incorporate some domain-specific features. For example, Eisenstein and Barzilay [62] incorporate cue phrases, and Nguyen et al. [153] incorporate speaker identity for segmenting meetings. As new forms of discourse (e.g., asynchronous conversation) are created and become popular, we face new challenges of incorporating their distinctive features (e.g., conversational structure) into these probabilistic models.

If we have enough labeled data for training (i.e., a corpus annotated with topic segments), then a **supervised approach** can be used to combine a large number of features and optimize their relative weights. One can use a binary classifier (e.g., Support Vector Machines (SVMs) [49]) or a sequence labeler (e.g., Conditional Random Fields (CRFs) [110]) to make a yes-no boundary decision between any two consecutive sentences. The feature set can include lexical similarity scores used in the unsupervised models, cue phrases and other domain-specific features. In the supervised framework, the challenge is to find the right set of features and the right way to model the problem for a particular discourse. For example, as described below, topic segmentation in asynchronous conversation cannot be reduced to a yes-no boundary decision and therefore requires a more sophisticated model.

**Topic Segmentation in Asynchronous Conversation**

As noted by Purver [165], it can be hard to define what exactly we mean by a topic. In some sense, the definition of topic and its granularity depend on the application at hand. For example, for categorization of news articles, topics could represent high-level distinctions among politics, culture, business, sports, science and so on. For meetings, topics could represent the items on the agenda. However, often it is not easy to define the right granularity of a topic. For example, Gruenstein et al. [77] reported that annotators, when asked to identify topic shifts in the ICSI meetings [92] — a collection of open-domain, less-structured meetings on subjects which the annotators were not familiar with —, did not agree at all as the notion of topic was too fine-grained. However, Galley et al. [74] found that annotators agreed reasonably well when topics are coarse-grained. Banerjee and Rudnicky [13] found that with more specific guidance (e.g., a list of candidate topics from which to choose), annotation agreement could be improved significantly.

One of our future goals is to automatically generate summaries of any asynchronous conversation (e.g., email, blog). Therefore, our notion of topic is similar to that of Galley et al. [74] for meeting summarization. In particular, we consider a topic to be something about which the participants discuss or argue or express their opinions. For example, a blog conversation in Slashdot[5], that begins with a discussion on *breaches of US army servers*, also covers *Iraq and Vietnam wars*, *hacker vs. cracker* and many others. An email conversation about an upcoming meeting may discuss *location* and *meeting agenda*. Our annotators were asked to identify as many topics as they feel most natural to convey the overall content structure of the conversation. For example, in the email conversation shown in Figure 1.1, our annotators found two different topics. The conversation starts with asking for *attendance via phone to a face to face meeting*, then also discusses the *time to call*.

In topic segmentation, we are interested in identifying what portions of the conversation are about the same topic, i.e., clustering the sentences based on their topics. Figure 1.2 shows the same email conversation of Figure1.1 annotated with topic information by our annotators. The right most column specifies a particular topic segmentation by assigning the same topic ID to sentences belonging to the

[5]http://it.slashdot.org/story/09/05/28/1952214/hackers-breached-us-army-servers

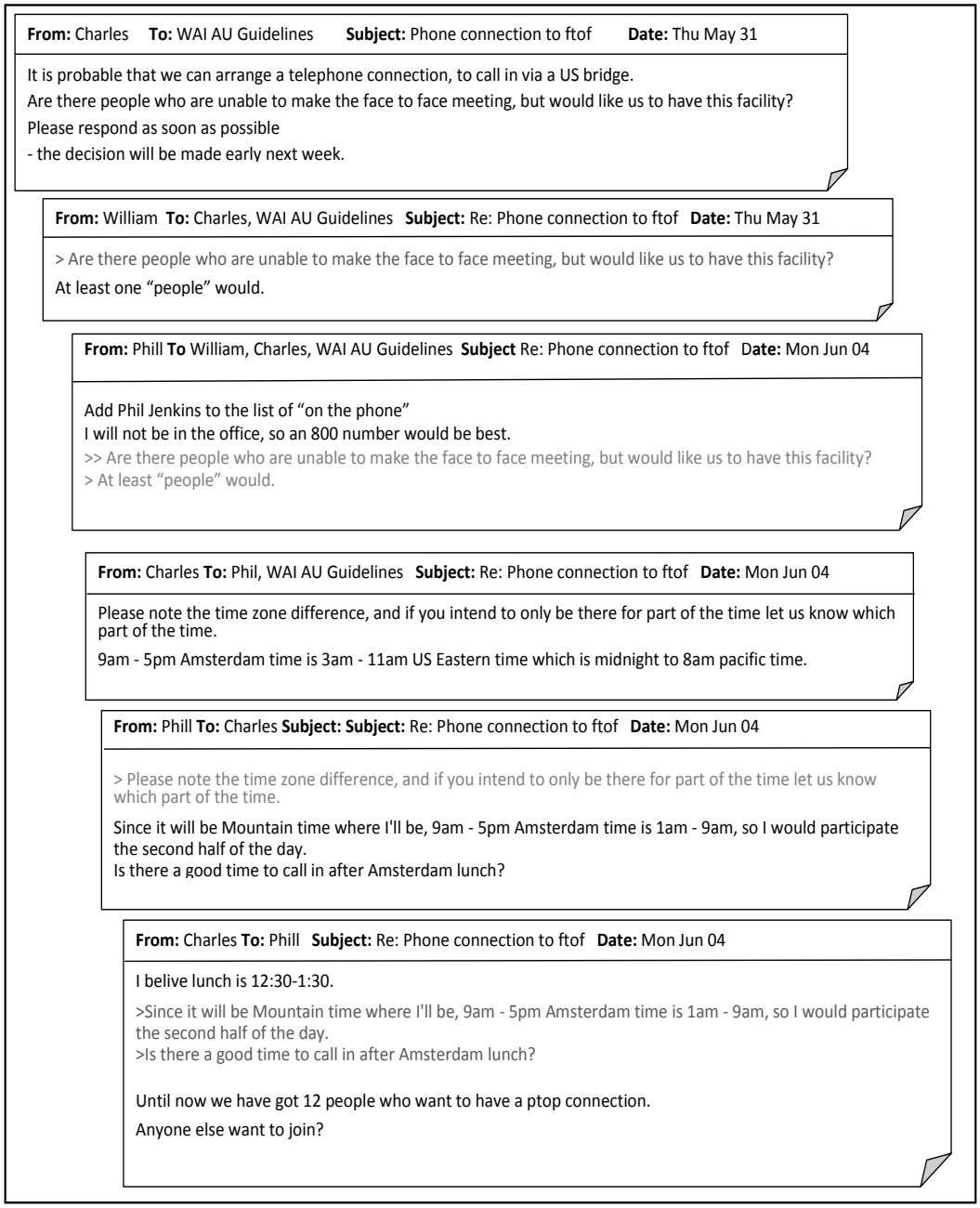



**Figure 1.1:** Sample email conversation from the BC3 email corpus [213].

same topic. We also use different colors to differentiate the topics.

To our knowledge, no-one has studied this problem before. Therefore, there are no standard corpora and evaluation metrics available. Also, because of its asynchronous nature, topics in these conversations often do not change sequentially. Notice that the conversation in Figure 1.2 starts with a discussion on topic 1, then shifts to topic 2, again revisits topic 1 in the last email. In other words, topics in asynchronous conversation are often interleaved, and the sequentiality constraint of topic segmentation in monolog and synchronous dialog does not hold anymore. As a result, the models that are successful in monolog or synchronous dialog may not be as successful, when they are directly applied to asynchronous conversation.

Furthermore, as can be noticed in Figure 1.2, writing style varies among participants, and many people tend to use informal, short and ungrammatical sentences, thus making the discourse much less structured compared to written monologs.

One unique aspect of asynchronous conversation that, at first glance, may appear to carry all the necessary topic information is its subject headers. However, subject headers are often not enough and could sometimes even be misleading. For example, in the email conversation shown in Figure 1.2, participants keep talking about different topics using the same subject (i.e., *Phone connection to ftof*). This problem is more evident in more informal interactions like public blogs.

Asynchronous conversations have their own distinctive features that could be useful for topic segmentation. One of the most important indicators that we hypothesize to be very informative is conversation structure. As can be seen in our email conversation in Figure 1.2, participants often reply to a post and/or use quotations to talk about the same topic. Notice also that the use of quotations can express a conversational structure that is at a finer level of granularity than the one revealed by reply-to relations. To leverage this key information, the first challenge we face is capturing the conversation structure at the quotation (i.e., text fragment) level. In Section 1.1.3, we describe a method to extract this finer-grained conversational structure. Beside conversation structure, other conversation-specific features like *sender, recipient, mentioning names* could be also useful for topic segmentation.

Now that we have identified the key distinctive features, the next challenge we face is incorporating these features into our topic segmentation models in a principled way. As described before, **LCSeg** [74] and **LDA** [26] are the two

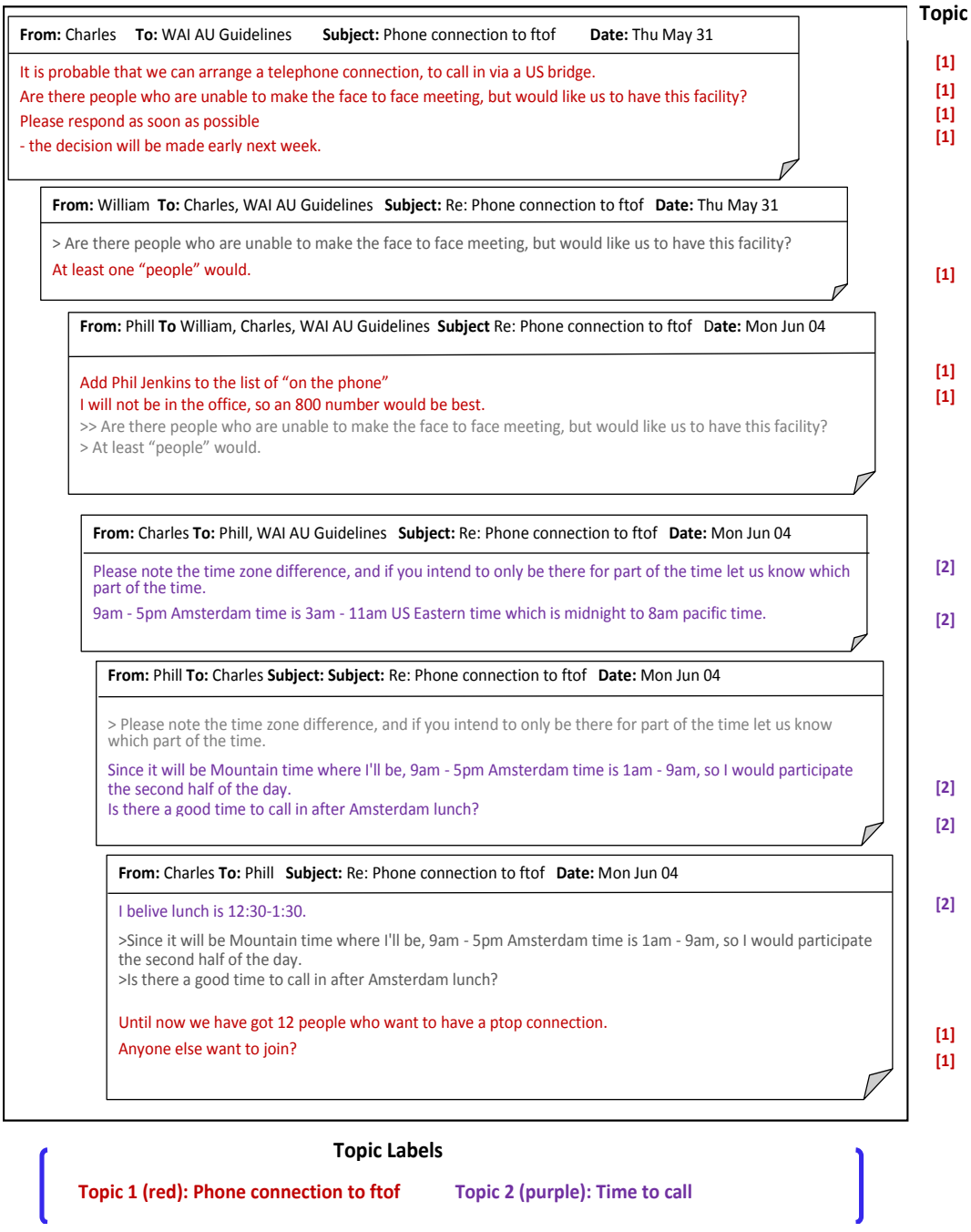



**Figure 1.2:** The email conversation of Figure 1.1 with topic annotations.

state-of-the-art unsupervised topic segmentation models developed for monolog and synchronous dialog from two different viewpoints. In this thesis, we propose a graph-theoretic clustering framework to incorporate the fine-grained conversational structure (i.e., a graph) into the LCSeg model. On the other hand, to incorporate the conversational structure into the LDA model, we propose to replace its standard Dirichlet prior by an informative Dirichlet Tree prior.

Although the unsupervised models have the key advantage of not requiring any labeled data, they can be limited in their ability to learn domain-specific knowledge from a possible large and diverse set of features [62]. In contrast, the supervised framework serves as a viable option to combine a large number of features and optimize their relative weights for decision making, but relies on labeled data for training. The amount of labeled data required to achieve an acceptable performance is always an important factor to consider for choosing supervised vs. unsupervised. In this thesis, we propose a supervised topic segmentation model that outperforms all the unsupervised models, even when it is trained on a small number of labeled conversations. Our model uses a discriminative classifier to combine all the important features in a graph-theoretic clustering framework.

### Topic Labeling

A topic label should give a brief high level overview of the topic discussed in the topical segment. For example, the labels for the two topic segments in Figure 1.2 are *phone connection to ftof* and *time to call*. The task of topic labeling can be framed as a **keyphrase generation (or extraction)** problem, where our goal is to generate (or extract) phrases that are representative of the given text. In keyphrase extraction, we select the phrases verbatim from the given text, whereas in keyphrase generation, we generate novel phrases based on the extracted information. Given that the human-authored topic labels are abstractive in nature (see the examples above), keyphrase generation has more potential than keyphrase extraction to get closer to human-like topic labels. However, generation is computationally much more complex and challenging than extraction since it requires inference, aggregation and abstraction. Therefore, most existing approaches are extractive in nature.

Several supervised and unsupervised approaches to keyphrase extraction have been proposed focusing only on monologs (e.g., paper abstracts). Generally, these methods operate in two steps: first, they find the candidate keyphrases from the text, then they rank (or filter) them based on the their relevancy to the text. Candidate keyphrases can be extracted using either a NLP chunker or *n*-gram sequences [91, 135, 137]. Supervised methods use a number of features (e.g., TF.IDF, position) in a classifier to filter these candidates [91, 135]. Unsupervised methods use different heuristics (e.g., TF.IDF, graph centrality) to rank the candidates [139].

Mei et al. [137] find that using a NLP chunker to select the candidates leads to poor results due to its inaccuracies, especially when it is applied to a new domain. Finding the right value of *n* in the *n*-gram sequences is also an issue, and Mihalcea and Tarau [140] claims that including all possible *n*-gram sequences excessively increases the search space. Instead, they follow a different approach, where they first select the words of a certain POS and rank them using a ranking method, then in the post-processing step, construct the keyphrases based on the co-occurrences of the top ranked words in the text. Their approach with a graph-based (unsupervised) ranking method (also known as *random walk*) for ranking the words achieves the state-of-the-art performance outperforming supervised methods [139].

#### Topic Labeling in Asynchronous Conversation

A direct application of the ranking method proposed by Mihalcea and Tarau [140] to label a topical segment in an asynchronous conversation would consider the words in the segment as nodes in a graph, define the edges between the nodes based on the **co-occurrence** of the respective words in the text, and then run the PageRank algorithm [155] on the graph. Our hypothesis is that better topic labels can be identified if, in addition to co-occurrence relations, we also consider aspects that are specific to asynchronous conversations.

Our first finding is that the *leading sentences* of a topical segment carry informative clues for the topic labels, since this is where the speakers will most likely try to signal a topic shift and introduce the new topic. For example, in Figure 1.2, notice that in almost every case, the leading sentences of the topical segments cover the information conveyed by the respective topic labels.

Our second clue is again the finer-grained *conversational structure* (i.e., a graph) used for topic segmentation. Carenini et al. [35] successfully applied the PageRank algorithm to this graph to measure the importance of a sentence. Their finding implies that an important node in the conversation graph is likely to cover an important aspect of the topics discussed in the conversation. Our intuition is that, to be in the topic label, a keyword should not only co-occur with other keywords, but it should also come from an important fragment in the graph.

Now, the challenge is how to incorporate these two conversation-specific features into the graph-based ranking (i.e., random walk) model. In this thesis, we propose and evaluate a *biased random walk* model [139] to incorporate the clues from the leading sentences of a topical segment, and a *co-ranking* model [238] to incorporate the fine-grained conversation structure.

### 1.1.2 Rhetorical Structure

In addition to the high-level topical structure, a discourse exhibits a finer-level structure called *rhetorical (or coherence) structure*, which logically binds its units (i.e., clauses and sentences) together. The meaning of a discourse unit relates to the previous and the following ones.

Several formal theories of discourse have been proposed to describe the coherence structure of a text (e.g., [10, 122, 225]). **Rhetorical Structure Theory (RST)** [122], one of the most influential of them, represents texts by labeled hierarchical structures, called Discourse Trees (DTs). For example, consider the DT shown in Figure 1.3 for the following text taken from the RST-DT corpus [38]:

- But he added: "Some people use the purchasers' index as a leading indicator, some use it as a coincident indicator. But the thing it's supposed to measure — manufacturing strength — it missed altogether last month."

The leaves of a DT correspond to contiguous atomic text spans, called *elementary discourse units* (EDUs). EDUs are *clause-like units* that serve as building blocks [38]. Adjacent EDUs are then related by coherence relations (e.g., *Elaboration, Contrast*), thereby forming larger units (represented by internal nodes), which in turn are also linked by coherence relations. Discourse units linked by a relation

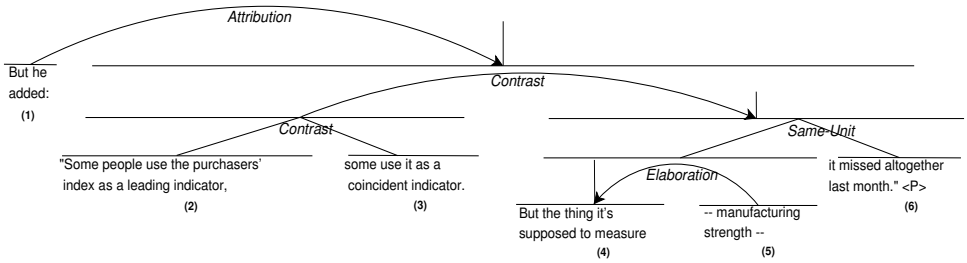



**Figure 1.3:** Discourse tree for two sentences. Each sentence contains three EDUs. Horizontal lines indicate text segments; satellites are connected to their nuclei by curved arrows.

are further distinguished based on their relative importance in the text: the **nucleus** being the central part, whereas **satellites** are peripheral ones. For example, in Figure 1.3, *Elaboration* is a relation between a nucleus (EDU 4) and a satellite (EDU 5), and *Contrast* is a relation between two nuclei (EDUs 2 and 3).

As mentioned before, messages in asynchronous conversations can be long unless there is a length constraint (e.g., Twitter). For example, the average length of a comment in our blog corpus containing conversations from Slashdot is 11.7 sentences. Each message in an asynchronous conversation locally forms a coherent monolog, i.e., it exhibits a rhetorical structure. For example, consider the DT shown in Figure 1.4 for the first sentence of the fifth email in Figure 1.1.[6] The second EDU *where I'll be* Elaborates the first EDU *Since it will be Mountain time*, and the resultant span containing the first and the second EDUs is Elaborated by the third EDU *9am - 5pm Amsterdam time is 1am - 9am*. Finally, the span containing the first three EDUs Explains the fourth EDU *so I would participate the second half of the day*. Rhetorical structure provides a useful discourse structure that has been shown to be beneficial for a range of NLP applications including text summarization and compression [54, 118, 126, 194], text generation [163], sentiment analysis [112, 189], information extraction [130, 209] and question answering [215]
Rhetorical analysis in RST involves two subtasks: **discourse segmentation** is the task of breaking the text into a sequence of EDUs, and **discourse parsing** is the task of linking the discourse units into a labeled tree. Both discourse segmentation

[6] We do not show the discourse tree for the entire email post to avoid visual clutter.

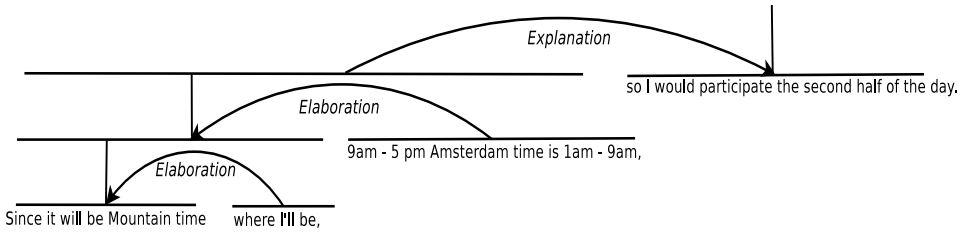



**Figure 1.4:** Discourse tree for the first sentence of the fifth email in Figure 1.1

and parsing are considered to be challenging tasks as we describe below.

### Discourse Segmentation

For written texts, it is taken for granted that sentence boundaries are also EDU boundaries. So, discourse segmentation is the task of deciding whether there should be an EDU boundary after each word in a sentence except the last one. Discourse segmentation has a strong influence on the accuracy of a discourse parser. For example, Soricut and Marcu [191] found a 29% error reduction in parsing when a perfect segmentation is provided to the parser. However, finding EDUs is challenging because there is no general agreement as to what constitutes the EDUs or what their properties are [198]. For example, consider the following sentence:

- John said that the journal paper won the best paper award.

How many EDUs are there? Since it contains two verbs, it is reasonable to think that it has two: *John said* and *that the journal paper won the best paper award*. But, at the same time we would expect an EDU to be structurally complete. Here, *John said* is neither syntactically nor semantically complete. There is disagreement among researchers on this issue [198].

Existing approaches to discourse segmentation can be divided into two types: rule-based and supervised learning. **Rule-based** approaches use hand-crafted rules that are based on syntactic categories and POS tags [113, 210]. On the other hand, **supervised** approaches use a handful of lexical and syntactic features to learn a segmentation model from the labeled data. For example, Soricut and Marcu [191] learn a generative model, Sporleder and Lapata [194] learn a boosting (ensemble) model, Fisher and Roark [72] learn a log-linear model, and Hernault et al. [84]

learn a SVM. In general, supervised segmenters perform better than rule-based segmenters. However, the challenge is to come up with the right set of features. In this thesis, we propose a discriminative model that uses mainly syntactic features including rules extracted from syntactic parse trees, POS tags and chunk tags.

**Discourse Parsing**

Once the EDUs are identified, the discourse parsing problem is to determine which discourse units (EDUs or larger units) to relate (i.e., the structure), and what relations (i.e., the labels) to use in the process of building the DT. Specifically, it requires: *a parsing model* to explore the search space of possible structures and labels for their nodes; and *a parsing algorithm* for deciding among the candidates.

Often discourse connectives like *but, because, although* convey clear information on the kind of relation linking the two text segments. Earlier work developed rule-based (unsupervised) systems based on discourse connectives (e.g., [125]). However, this approach faces at least three serious problems. First, identifying discourse connectives is a difficult task of its own, because depending on the usage, the same phrase may or may not signal a coherence relation. Second, often coherence relations are not explicitly signaled by discourse cues [127, 196]. Third, discourse connectives could be ambiguous in signaling relations [198].

Another line of research uses unambiguous discourse cues to automatically label a large corpus with coherence relations (e.g., *although* for *Contrast*) that could be then used to train a supervised classifier [127, 195]. To make the classifier work for implicit cases, they remove the connectives from the training instances. However, later studies confirm that classifiers trained on instances by stripping off the original cue phrases do not generalize well to implicit cases [24, 196]. Furthermore, this approach does not attempt to solve the actual *parsing* (i.e., building hierarchical tree) problem, rather it attempts to solve a *tagging* (i.e., flat) problem. To perform an effective and complete rhetorical parsing, one needs to employ supervised learning techniques based on human annotated data.

Supervised approaches to discourse parsing can be judged based on two criteria: the type of model it uses as the *parsing model*, and the type of *parsing algorithm* it uses to build the discourse tree. Marcu [124] uses a *C*4.5 decision tree

classifier as the parsing model in a shift-reduce (bottom-up) parsing algorithm. In the sentence-level discourse parser SPADE[7], Soricut and Marcu [191] use a generative probabilistic parsing model with a CKY-like bottom-up parsing algorithm. Subba and Di-Eugenio [201] use an ILP-based[8] classifier as the parsing model in a shift-reduce parsing algorithm. In the HILDA system[9], Hernault et al. [84] and Feng and Hirst [69] use two SVM classifiers — one for the structure and one for the label — in a cascade as the parsing model. They employ this cascaded parsing model iteratively to build the discourse tree in a bottom-up fashion.

The existing parsing models mentioned above disregard the structural inter-dependencies between the tree constituents. However, we hypothesize that like syntactic parsing, discourse parsing is also a structured prediction problem, which involves predicting multiple variables (i.e., the structure and the relation labels) that depend on each other [188]. Recently, Feng and Hirst [69] also found these inter-dependencies to be critical for parsing performance. In this thesis, we use undirected conditional graphical models (i.e., CRFs) to capture structural dependencies between the tree constituents.

Another technical question related to parsing models is whether to employ a single unified model or two different models for parsing at the sentence-level (i.e., intra-sentential) and at the document-level (i.e., multi-sentential). Existing discourse parsers use a single model and do not discriminate between intra- and multi-sentential parsings. This approach has the advantages that it makes the parsing process easier, and the model gets more data to learn from. However, the feature set used in the parsing model may not generalize well to these two different parsing conditions, and there is empirical evidence that coherence relations are distributed differently intra-sententially vs. multi-sententially [38].

Our hypothesis is that a more effective approach could be to use two separate models that would also allow us to exploit the strong correlation observed between the text structure and the DT structure. However, this poses an addition challenge of combining the two stages of parsing effectively and efficiently. Although, most sentences have a well-formed discourse sub-tree in the full DT (e.g., the second

[7] http://www.isi.edu/licensed-sw/spade/

[8] ILP stands for Inductive Logic Programming

[9] http://www.cs.toronto.edu/ weifeng/software.html

sentence in Figure 1.3), there are few cases where rhetorical structures violate sentence boundaries. For example, the first sentence in Figure 1.3 does not have a well-formed discourse sub-tree because the unit containing EDUs 2 and 3 merges with the next sentence and only then is the resulting unit merged with EDU 1.

In this thesis, we combine our intra-sentential and multi-sentential parsers in two different ways. Since most sentences have a well-formed discourse sub-tree in the full DT, our first approach constructs a DT for every sentence using our intra-sentential parser, and then runs the multi-sentential parser on the resulting sentence-level DTs. Clearly, this approach disregards those cases where rhetorical structures violate sentence boundaries. To deal with those cases, our second approach builds sentence-level sub-trees by applying the intra-sentential parser on a sliding window covering two adjacent sentences and consolidating the results produced by overlapping windows. Then, the multi-sentential parser takes all these sentence-level sub-trees and builds a full rhetorical parse for the whole document.

Most of the existing discourse parsers apply greedy and sub-optimal parsing algorithms to build the DT. These algorithms offer a simple and efficient solution, but are not very effective in terms of accuracy. Therefore, a challenge we address in this thesis is to develop an optimal parsing algorithm, which is also efficient.

### 1.1.3 Dialog Acts and Conversational Structure

As mentioned before, a conversation is a joint activity between two or more participants. The participants take *turns*, each of which consists of one or more utterances [177]. The utterances in a turn perform certain actions (e.g., asking a question, requesting something), which are called **dialog acts** (DAs) [12]. For instance, in the second email shown in Figure 1.5, the sentence *At least one "people" would* answers the question posed in the first email. Two-part structures connecting two DAs (e.g., *Question-Answer, Request-Accept*) are called **adjacency pairs** [180]. Dialog acts and adjacency pairs provide useful structures for conversation analysis, that have been shown to be beneficial for a range of applications including conversation summarization [148, 151] and conversational agents [5].

Now consider the following multi-party (synchronous) exchange:

- John: Anyone up for lunch?

It's 12:10

- Ale: I'm in
  who else is joining?
- Carla: John, are you done with the report?
  You were supposed to finish it before lunch
- Kevin: I'll also join
- John: I'm almost done

Clearly, in this exchange, two simultaneous conversations are going on: one is about *going for lunch*, and the other is about *the report*. Notice that in multi-party conversations, it is always not obvious who is talking to whom. For example, a straightforward reading would mistakenly consider Kevin's utterance as a response to Carla, and John's last utterance as a response to Kevin. Identifying which utterance contributes to which conversation is known as **disentanglement** [63] (also called **conversation structure extraction** [1, 131] and **thread detection** [184]). It is considered as an essential prerequisite for other higher-level conversation analysis (e.g., dialog act recognition). In the following, we give an overview of the two tasks: dialog act recognition and conversational structure extraction.

**Dialog Act Recognition**

Most of the previous work on dialog act modeling has mainly focused on synchronous conversations, e.g., [102, 232] for chats, [57, 107] for meetings, [14, 200] for phone conversations. The dominant approaches are mostly supervised, and use either simple classifiers (binary or multi-class) or more structured models like Hidden Markov Models (HMMs), Maximum Entropy Markov Models (MEMMs), and Conditional Random Fields (CRFs). Since turns in synchronous conversations occur one after the other with minimal delay, the conversation flow in these conversations exhibits sequential dependencies between adjacency pairs (e.g., *question* followed by *answer*, *request* followed by *grant*). Sequence labelers like HMMs and CRFs, which are capable of capturing these inter-dependencies between the dialog acts, generally perform better than the simple classifiers (e.g., MaxEnt, SVMs).

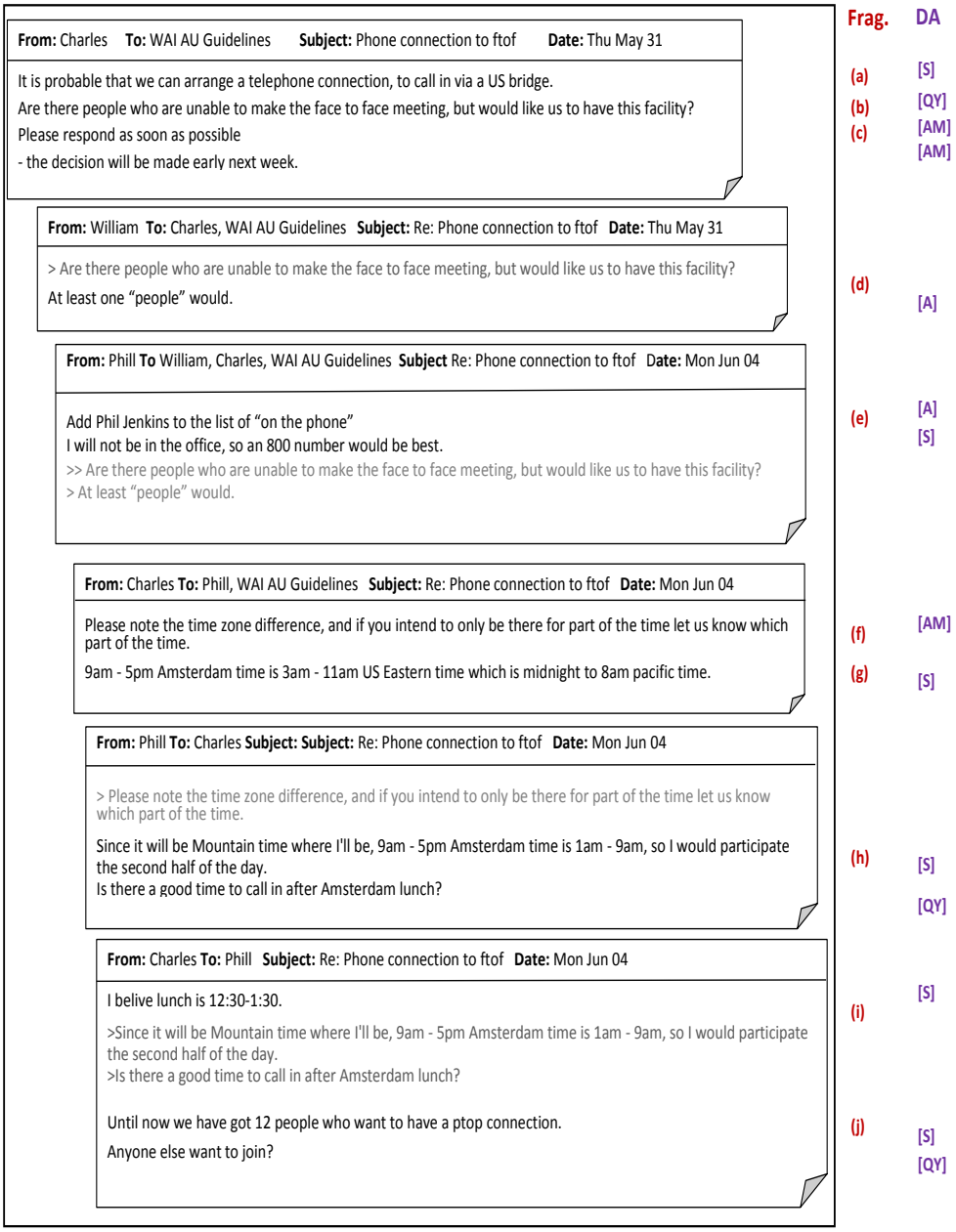



**Figure 1.5:** The email conversation of Figure 1.1, now annotated with dialog acts (DAs). The right most (DA) column specifies the dialog acts for the sentences: *S* stands for *Statement*, *QY* stands for *Yes-no question*, *A* stands for *Accept response*, and *AM* stands for *Action motivator*. The Frag. column specifies the fragments in the Fragment Quotation Graph (FQG) described in Section 1.1.3.

**Dialog Act Recognition in Asynchronous Conversation**

Unlike synchronous conversations, consecutive turns in asynchronous conversations can be far apart in time, and multiple turns can largely overlap [36]. Among the supervised approaches, Cohen et al. [46] [39] first use the term *email speech act* for classifying *emails* (not sentences) based on the writers' intentions (e.g., deliver, meeting). Jeong et al. [93] propose semi-supervised boosting to tag the *sentences* in email and forum conversations with DAs by adapting knowledge from annotated spoken conversations, i.e., meeting and phone conversations. However, their approach does not consider the sequential dependencies between the DA types.

Among recent attempts at unsupervised DA modeling, Ritter et al. [174] propose HMMs to cluster the Twitter posts into DAs. Their simple HMM model tends to find some undesired topical clusters in addition to the DA clusters. Without labeled data, distinguishing DAs from topics is in fact a common challenge, because many of features used for modeling DAs are also indicators of topics. Ritter et al. [174] propose a HMM+Topic model that tries to separate the DA indicators from the topic words. Since Twitter messages are short, HMMs are able to learn meaningful sequence dependencies from the reply-to structure of the conversation.

When messages are long, capturing sequential dependencies between the act types for sentence-level DA tagging adds further challenges because the two components of adjacency pairs could be far apart in the sequence. Because of the asynchronous nature, the temporal order of the utterances often lacks these dependencies. This is true even when the reply-to structure between messages is considered. For instance, notice in Figure 1.5 that there are two other sentences in between the question asked in the first email (i.e., *Are there ..*) and the reply made in the second email (i.e., *At least ..*). This example also demonstrates that the use of quotations (see the second email) could help us in putting the components of adjacency pairs close to each other. However, this comes with another challenge of capturing the conversational structure at the quotation level. In the next section, we will present a method to capture the conversational structure at the quotation level.

### Conversational Structure Extraction

The fact that multi-party conversations are often made up of interwoven threads was first acknowledged by Rosé et al. [176]. Aoki et al. [7] report an average of 1.76 active conversations in a exchange involving 8 to 10 speakers. Elsner and Charniak [63] report an average of 2.75 conversations active at a time in multi-party chat. Several methods have been proposed to disentangle multi-party synchronous conversations (e.g., [63, 131, 184, 223]). We describe them briefly in Section 2.2.3.

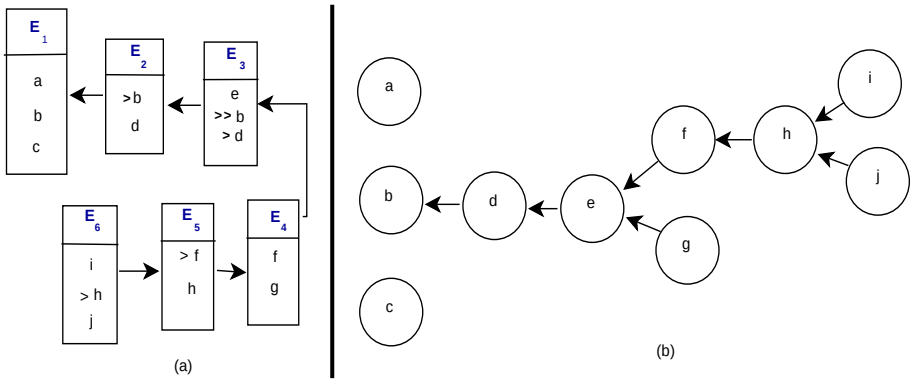



**Figure 1.6:** (a) The email conversation of Figure 1.5 with its fragments. The real contents are abbreviated as a sequence of symbols. Arrows indicate reply-to links. (b) The corresponding Fragment Quotation Graph.

While disentanglement is necessary for many multi-party synchronous conversations, asynchronous media like email and blog services (e.g., Gmail, Slashdot, Twitter) generally organize comments into tree-structured threads using reply-to relations. In absence of the reply-to relations, automatic methods to uncover the thread structure have also been proposed (e.g., [220, 224]). However, the use of quotations in asynchronous conversations can express a conversational structure that is finer grained and can be more informative than the one revealed by reply-to relations [36]. For example, consider the relation between the new and quoted (marked with '>') text fragments in the second and fifth emails in Figure 1.5. The proximity between quoted and new text fragments can represent a conversational link that is more specific than the actual reply-to link. Carenini et al. [34] presented a method to capture this finer level conversational structure in the form a

graph called Fragment Quotation Graph (FQG). For example, Figure 1.6 shows the FQG for our email conversation in Figure 1.5. The nodes in the FQG represent text fragments and the edges represent reply relations between the fragments.

Carenini et al. [34, 35] show the benefits of using a FQG in email summarization. In this thesis, we generalize the FQG to any asynchronous conversation, and demonstrate how topic models and dialog act models can benefit significantly from this fine conversational structure of asynchronous conversation.

## 1.2 Computational Methodologies

Table 1.4 lists the computational methods used for different discourse analysis tasks in this thesis. These methods can be classified into two broad classes: **graph-based methods** and **probabilistic graphical models**.

| Tasks | Computational methods | Comments |
|---|---|---|
| Topic Segmentation | Graph-based clustering | N-cut model |
| | LDA with Dirichlet Tree prior | Generative topic model |
| Topic Labeling | Graph-based ranking | Random walk model |
| Discourse Segmentation | Binary classifier (MaxEnt) | Discriminative model |
| Discourse Parsing | Multi-layered CRFs | Discriminative UGM |
| Dialog act modeling | HMMs (unsupervised) | Generative DGM |

**Table 1.4:** Computational methods used for different discourse analysis tasks. N-cut stands for Normalized cut, UGM stands for Undirected Graphical Model, and DGM stands for Directed Graphical Model.

### 1.2.1 Graph-based Methods for NLP

The adoption of a graph-theoretic framework allows us to represent linguistic units as diverse as words, sentences and documents as nodes in a graph, and relations between them as edges in the graph. It then allows us to apply a variety of efficient algorithms on the graph to solve a wide range of NLP applications. Table 1.5 shows some widely used graph-based algorithms and their applications to various NLP problems. In this thesis, we use graph-based clustering and ranking models

for topic segmentation and labeling in asynchronous conversation.

| Tasks | Methods | Applications |
|---|---|---|
| Clustering | Min cut/ Max flow | Topic segmentation [121], Reference resolution [190] Chat disentanglement [63], Subjectivity detection [157] |
| Ranking | PageRank, HITS, ArcRank | Summarization [65], Keyphrase extraction [140] Passage retrieval [154], WSD [138], Lexical acquisition [228] |
| Learning | Semi-supervised, Manifold | Text categorization [208], Sentiment analysis [208], Dialog act tagging [202], POS tagging [203], MT [2] |
| Matching | Min-cost match | Textual entailment [78] |

**Table 1.5:** Graph-based methods applied to NLP tasks. WSD stands for Word Sense Disambiguation and MT stands for Machine Translation.

### Graph-based Clustering and Ranking

Let $G = (V,E)$ be a weighted undirected graph, where the nodes $V$ represent the linguistic units (e.g., sentences, words) and the edge weights $w(x,y)$ represent some form of "similarity" between units $x$ and $y$. The term "similarity" can refer to a number of things. For example, it can be a score of lexical similarity between two sentences $x$ and $y$, it can be a confidence score of a classifier that determines how likely is that sentences $x$ and $y$ belong to the same class, and so on.

**Graph-based clustering** (also called **correlation clustering** [15]) aims to partition the nodes of the graph into disjoint groups, where, by some measure, the similarity among the nodes in a group is high and the similarity across different groups is low. Several objectives and efficient algorithms to compute the globally optimal partition based on those objectives have been proposed, e.g., min-cut, n-cut [185].

Graph-based clustering has been used to solve a number of problems in discourse (see Table 1.5). One can encode different discourse-related information in terms of the edge weights. For example, Pang and Lee [157] encode contextual dependencies for subjectivity detection. Malioutov and Barzilay [121] use cosine similarity to encode lexical cohesion into their unsupervised topic segmentation

model. In this thesis, we encode lexical cohesion as well as conversational structure into our unsupervised topic segmentation model. Furthermore, by defining the edge weights based on the confidence score of a classifier, we encode a large number of domain-specific features into our supervised topic segmentation model.

**Graph-based ranking** algorithms (e.g., PageRank [31], HITS [105]) are ways of measuring the importance of a node within a graph, by taking into account global information computed from the entire graph, rather than relying only on local node-specific information [139]. TextRank [140] (or LexRank [65]), a version of PageRank for textual units, is the most popular graph-based ranking algorithm in NLP. A probabilistic interpretation of this algorithm can be given from the concept of a **random walk** on a graph, where the graph represents the transition matrix of a Markov chain, and the ranking gives the stationary distribution of the chain.

The random walk framework allows us to incorporate knowledge from multiple sources as priors [208], biases [154] and co-ranking [238]. This ability enables us to incorporate different discourse phenomena in the ranking process. In this thesis, we use a biased random walk and a co-ranking framework to incorporate clues from the leading sentences and the FQG respectively for the task of topic labeling.

### 1.2.2 Probabilistic Graphical Models

Uncertainty is inevitable in many real-world scenarios because we never observe the world state fully, and even those aspects that we observe are sometimes noisy. As a consequence, we often employ probabilistic reasoning to design a real-world system. **Probabilistic Graphical Models (PGMs)** constitute a general class of probabilistic models with the following key properties [108]:

- A compact graph *representation* which is semantically intuitive.
- Efficient probabilistic *reasoning* in a general framework.
- Separation of *reasoning* from *representation*.
- A general *learning* framework.

Complex systems often involve multiple interrelated aspects (i.e., random variables), some of which are observed while some are not, and some that interact with

each other directly while others do not. Probabilistic graphical models define joint distributions over the random variables that then allow us to reason about some query variables, possibly given observations about some others. The graph representation allows us to define arbitrary joint distributions intuitively and compactly by exploiting the interactions among the variables. The key insight of graphical modeling is that a distribution over many variables can often be represented as a product of local functions that each depend on a much smaller subset of variables.

The reasoning algorithms work directly on the graph structure and are generally faster than working on the joint distribution explicitly. Separating reasoning from representation allows us to develop a general class of algorithms that can be applied to any model within a broad category, conversely, our model can be improved for a specific application without modifying the algorithms.

The attributes, their interdependencies and the parameters in a graphical model can be either defined by domain experts or learned automatically from data. PGMs provide a general learning framework that is very effective in practice.

Table 1.6 shows examples of the two families of PGMs: **Directed Graphical Models (DGMs)** and **Undirected Graphical Models (UGMs)**. These models can be further distinguished based on how they are trained. **Generative** models define a joint distribution $p(\mathbf{y}, \mathbf{x}|\Theta)$ over inputs and outputs, then use the (learned) model $\Theta$ to infer the conditional $p(\mathbf{y}|\mathbf{x}, \Theta)$. This approach has several advantages including efficient training, unsupervised modeling and handling missing data [145]. However, it also has crucial limitations. The dimensionality of $\mathbf{x}$ can be very large with complex dependencies, so modeling $\mathbf{x}$ can be difficult and may lead to intractable models, but ignoring the dependencies can lead to inaccurate models [204].

| | Generative | Discriminative |
|---|---|---|
| Directed | LDAs, HMMs, State space models | Maximum Entropy Markov Model |
| Undirected | Markov Random Fields | Conditional Random Fields |

**Table 1.6:** Families of probabilistic graphical models.

A solution to this problem is a **discriminative** approach, which models the conditional distribution $p(\mathbf{y}|\mathbf{x}, \Theta)$ directly. The key advantage of this approach is that dependencies involving only variables in $\mathbf{x}$ play no role in the conditional model.

In general, discriminative methods are more accurate than generative ones, since they solve an easier problem and do not "waste effort" in modeling the observations [145]. Other advantages include the ability to leverage a large number of input features $\mathbf{x}$, and the ability to relax strong independence assumptions.

PGMs have been widely used in numerous NLP tasks, because of their ability to predict multiple interrelated variables (i.e., structured output). For example, the dependencies between words in a sentence are modeled by Markov chains (i.e., language models) [100], the dependencies between POS tags of the words in a sentence are modeled by HMMs or CRFs, the hierarchical dependencies between syntactic constituents in a parse tree are modeled by CRFs, and so on. Fundamental to many discourse analysis tasks is also the ability to predict multivariate outputs. For example, the contents of a news article follow a hidden topical structure [19]; the dialog acts of a conversation are interrelated [200]; discourse entities, their attributes and their mentions are interrelated [59]; rhetorical structure and relations are interrelated [69]. Table 1.7 shows examples of probabilistic graphical models used in different discourse analysis tasks.

| Tasks | Graphical models used |
|---|---|
| Content (topic) modeling | HMMs [19], LDAs [61] |
| Dialog act modeling | HMMs [200], CRFs [102] |
| Reference resolution | CRFs [59] |
| Shallow discourse parsing in PDTB | CRFs [75] |

**Table 1.7:** Examples of graphical models used in discourse analysis. PDTB stands for Penn Discourse Treebank [164].

However, it is important to note that there is no universally best model — a model that works well in one task may work poorly in another — this is essentially the **no free lunch theorem** [231]. As a result, there is now a growing interest in devising new graphical models for different tasks in discourse analysis. In this thesis, as summarized in Table 1.4, we use an LDA with a Dirichlet Tree prior for topic segmentation in asynchronous conversation, discriminative UGMs (i.e., CRFs) as the parsing models in our discourse parser, and generative DGMs (i.e., unsupervised HMMs) for modeling dialog acts in asynchronous conversation. As

we describe in detail in the respective chapters, our choice of model for a specific task is driven by four considerations, namely the structure (or complexity) of the task, the amount of labeled data available for training the model, the number of features we want to incorporate into the model, and the (hidden) random variables we want to consider jointly in our model.

## 1.3 Our Contributions

Our contributions in this thesis aim to overcome the challenges for different discourse analysis tasks in asynchronous conversations as described in Section 1.1. Our key hypothesis is that to effectively address these technical challenges, we need to apply sophisticated graph-based methods and probabilistic graphical models described in Section 1.2. In the following, we summarize our contributions.

### 1.3.1 Topic Segmentation and Labeling

In **Chapter 2**, we propose a complete computational framework for performing topic segmentation and labeling in asynchronous conversations.[10]

Since there was no previous study on topic segmentation and labeling in asynchronous conversation, there were no standard corpora and evaluation metrics available for research purposes. We present two new corpora of email and blog conversations annotated with topics, and evaluate annotator reliability using a new set of metrics, which are also used to evaluate the computational models.

For *topic segmentation*, we extend LDA [26] and LCSeg [74], two state-of-the-art unsupervised models, to incorporate a fine-grained conversational structure (i.e., the Fragment Quotation Graph (FQG)), generating two novel unsupervised models **LDA+FQG** and **LCSeg+FQG**. We incorporate the FQG into LDA by replacing its standard Dirichlet prior with an informative Dirichlet Tree prior. On the other hand, we propose a graph-based clustering model to incorporate the FQG into LCSeg. In addition to that, we also propose a novel **supervised** segmentation model that combines lexical, conversational and topic features using a classifier in the graph-based clustering framework.

[10] Our corpora, annotation manual and source code for computational models are publicly available from www.cs.ubc.ca/labs/lci/bc3.html

For *topic labeling*, we propose to generate topic labels using an unsupervised extractive approach that identifies the most representative phrases in the text. Specifically, we propose two novel random walk models that respectively capture two forms of conversation specific information: (i) the fact that the leading sentences in a topical cluster often carry the most informative clues, and (ii) the fine-grained conversational structure of the conversation, i.e., the FQG.

We evaluated our framework in a series of experiments. Experimental results for the topic segmentation task demonstrate that both LDA and LCSeg benefit significantly when they are extended to consider the FQG, with LCSeg+FQG being the best unsupervised model. The comparison of the supervised segmentation model with the unsupervised models shows that the supervised method outperforms the unsupervised ones even using a limited number of labeled conversations, being the best segmentation model overall. Remarkably, the segmentation decisions of LCSeg+FQG and the supervised models are also highly correlated with human annotations. The experiment on the topic labeling task reveals that the random walk model performs better when it exploits conversation specific clues from the leading sentences and the conversational structure. The evaluation of the end-to-end system also shows promising results in both corpora, when compared with human annotations.

### 1.3.2 Rhetorical Analysis

In **Chapter 3**, we propose a complete probabilistic discriminative framework for rhetorical analysis comprising both a discourse segmenter and a discourse parser.[11]

For *discourse segmentation*, we propose a novel discriminative model that achieves state-of-the-art performance using fewer features than the existing models. Our main contribution is to come up with the right set of features including rules extracted from syntactic parse trees, POS tags and chunk tags.

For *discourse parsing*, our contributions aim to address the key limitations of the existing parsers (see Section 1.1.2), as we summarize them below:

Existing discourse parsers model the structure and the labels of a discourse tree (DT) separately, and do not capture the sequential dependencies between the DT

[11]The source code and an online demo of our rhetorical analysis framework is available at http://alt.qcri.org/discourse/Discourse-Parser-Demo/

constituents. To address this, we propose a novel discourse parser based on probabilistic discriminative parsing models, expressed as Conditional Random Fields (CRFs) [205], to infer the probability of all possible DT constituents. The CRF models effectively represent the structure and the label of a DT jointly, and whenever possible, capture the sequential dependencies between the constituents.

While existing discourse parsers do not discriminate between intra-sentential and multi-sentential parsings, we believe that distinguishing between these two can result in a more effective parsing method, which can exploit the strong correlation observed between the text structure and the DT structure. Furthermore, two separate parsing models — one for intra-sentential and one for multi-sentential — could leverage their own informative feature sets and the fact that coherence relations are distributed differently intra-sententially vs. multi-sententially.

In order to develop a complete and robust discourse parser, we combine our intra-sentential and multi-sentential parsers in two different ways. Since most sentences have a well-formed discourse sub-tree in the full DT (e.g., the second sentence in Figure 1.3), our first approach constructs a tree for every sentence using our intra-sentential parser, and then runs the multi-sentential parser on the resulting sentence-level trees. However, this approach would disregard those cases where rhetorical structures violate sentence boundaries (e.g., the first sentence in Figure 1.3). Our second approach, in the attempt of dealing with these cases, builds sentence-level sub-trees by applying the intra-sentential parser on a sliding window covering two adjacent sentences and by then consolidating the results produced by overlapping windows. After that, the multi-sentential parser takes all these sentence-level sub-trees and builds a full rhetorical parse for the whole document.

Finally, while existing discourse parsers apply greedy and sub-optimal parsing algorithms to build the discourse tree for a document, we apply an optimal parsing algorithm to find the most probable discourse tree.

While previous studies tested their approach only on one corpus, we evaluate our discourse segmenter and parser on two very different genres: news articles and instructional manuals. The results show that our approach to discourse parsing significantly outperforms the state-of-the-art, often by a wide margin.

### 1.3.3 Dialog Act Modeling

In **Chapter 4**, we describe our unsupervised approaches to dialog act modeling in asynchronous conversation. In particular, we investigate a graph-theoretic deterministic framework and two probabilistic conversational models for clustering sentences in an asynchronous conversation based on their dialog act types.

The graph-theoretic framework clusters sentences based on their lexical and structural similarity, but ignores the sequential dependencies between the acts. On the other hand, probabilistic conversational models frame the task as an unsupervised sequence labeling problem that we solve using variations of HMMs.

As described before, unlike synchronous conversations, asynchronous conversations often lack sequential dependencies between the act types in the temporal order of the utterances. We argue that the sequences extracted from the conversational structure allow a more effective learning of the sequential dependencies. We show that this is the case for both email and forum conversations, in particular when for email we use the fine-grained conversational structure, i.e., the FQG.

Similar to Ritter et al. [174], we also observe that simple unsupervised **HMMs**, when applied to cluster sentences based on their dialog acts, tends to also find some irrelevant topical clusters. Ritter et al. [174] address this problem by proposing an **HMM+Topic** model which tries to separate the topic words from the dialog act indicators. In our work, we propose an **HMM+Mix** model, which not only separates the topics, but also improves the emission distribution by defining it as a mixture model.

We evaluate our models on two different datasets: email and forum posts. To our knowledge, we are the first to perform a quantitative evaluation of unsupervised dialog act models for asynchronous conversation. The empirical results demonstrate that (i) the graph-theoretic framework is not the right model for this task, (ii) the probabilistic conversational models learn better sequence dependencies when they are trained on the sequences extracted from the graph structure of the conversation rather than when they are trained on the temporal sequences, and (iii) HMM+Mix is a better conversational model than the simple HMM model.

Finally, in **Chapter 5**, we conclude the thesis with a summary of our contributions

and directions for future work. We note that the exposition throughout this dissertation assumes that our readers are familiar with basic probabilistic statistical models, though not with the particular discourse processing tasks we study.

# Chapter 2

# Topic Segmentation and Labeling

In this chapter, we study the task of automatically identifying the high-level discourse structure, i.e., the topical structure of an asynchronous conversation. We present two new corpora of email and blog conversations annotated with topics, and evaluate annotator reliability for topic segmentation and labeling tasks. We propose a complete computational framework for performing topic segmentation and labeling in asynchronous conversations. Our approach extends state-of-the-art methods by considering the fine-grained structure of an asynchronous conversation, along with other important features. We do that by applying recent graph-based methods for NLP. For topic segmentation, we propose two novel unsupervised models that exploit the fine-grained conversational structure, and a novel graph-theoretic supervised model that combines lexical, conversational and topic features. For topic labeling, we propose two novel (unsupervised) random walk models that respectively capture conversation specific clues from two different sources: the leading sentences and the fine-grained conversational structure. Empirical evaluation shows that the segmentation and labeling performed by our best models outperform the state-of-the-art, and are highly correlated with human annotations.[1]

[1]This chapter is based on the journal article Joty et al. [98] (**JAIR-2013**). Portions of this work were also previously published in two peer-reviewed conference proceedings: Joty et al. [94] (**EMNLP-2010**) and Joty et al. [96] (**ICWSM-2011**).

## 2.1 Introduction

What makes a **topic** in asynchronous conversation? As mentioned before, defining the term topic is not a trivial task, and by large it depends on the target application. Since our ultimate goal is to be able to automatically generate summaries of asynchronous conversations (e.g., email, blog), our notion of topic is similar to that of Galley et al. [74] for meeting summarization. In particular, we consider a topic to be something about which the participants discuss or argue or express their opinions. In other words, a topic in asynchronous conversation emulates an item (or issue) in the meeting agenda. For example, an email conversation about an upcoming meeting may discuss *location*, *agenda* and *who should attend*. A blog conversation in Slashdot[2], that begins with a discussion on *breaches of US army servers*, also covers *Iraq and Vietnam wars*, *hacker vs. cracker* and many others.

Multiple topics seem to occur naturally in social interactions, whether synchronous (e.g., meetings, chats) or asynchronous. In the naturally occurring ICSI multi-party meetings [92], Galley et al. [74] report an average of 7.5 topical segments per conversation. In multi-party chat, Elsner and Charniak [63] report an average of 2.75 discussions active at a time. In the email and blog (asynchronous) conversational corpora that we present in this chapter, annotators found an average of 2.5 and 10.77 topics per email and blog conversation, respectively.

**Topic segmentation** refers to the task of grouping the sentences of an asynchronous conversation into a set of coherent topical clusters (or segments)[3], and **topic labeling** is the task of assigning a short description to each of the topical clusters to facilitate interpretations of the topics [165]. For example, in the sample truncated email conversation from our corpus shown in Figure 2.1, the majority of our three annotators found three different topics (or clusters). Likewise, in the truncated blog conversation shown in Figure 2.2, our annotators found six different topics. The right most column in each figure specifies a particular segmentation by assigning the same topic ID (or cluster ID) to sentences belonging to the same topic. The topics in each figure are also differentiated using different colors. The topic labels assigned by the annotators are listed below each conversation (e.g.,

[2] http://it.slashdot.org/story/09/05/28/1952214/hackers-breached-us-army-servers

[3] In this chapter, we use the terms topical cluster and topical segment interchangeably.

'Telecon cancellation', 'Tag document', 'Responding to I18N' in Figure 2.1).

Topic segmentation and labeling is often considered an essential prerequisite for higher-level conversation analysis [14] and has been shown to be useful in many language processing applications including text summarization [56, 80, 104], text generation [19], information extraction [3], and conversation visualization [117].

While extensive research has been conducted in topic segmentation for monolog (e.g., articles) and synchronous dialog (e.g., meetings), no-one has studied the problem of segmenting and labeling asynchronous conversations. Therefore, there are no reliable annotation scheme, no standard corpus, and no agreed-upon metrics available. Also, it is our key observation that, because of its asynchronous nature, and the use of quotations [51], topics in these conversations are often interleaved and do not change in a sequential way. That is, if we look at the temporal order of the sentences in a conversation, the discussion of one topic may appear to intersect with the discussion of others. As can be seen in Figure 2.1, after a discussion of topic 3 in the second and third email, topics 1 and 2 are revisited in the fourth email, then topic 3 is again brought back in the fifth email. Therefore, the sequentiality constraint of topic segmentation in monolog and synchronous dialog does not hold in asynchronous conversation. As a result, we do not expect models which have proved successful in monolog or synchronous dialog to be as effective when directly applied to asynchronous conversation.

Our contributions in this work aim to remedy these problems. First, we present two new corpora of email and blog conversations annotated with topics, and evaluate annotator reliability for the topic segmentation and labeling tasks using a new set of metrics, which are also used to evaluate the performance of the computational models. To our knowledge, these are the first such corpora that will be made publicly available. Second, we present a complete topic segmentation and labeling framework for asynchronous conversations. Our approach extends state-of-the-art methods (for monologs and synchronous dialogs) by considering the fine-grained structure of the asynchronous conversation along with other conversational features. In doing so, we apply recent graph-based methods for NLP [139] such as min-cut and random walk on paragraph, sentence or word graphs.

For topic segmentation, we propose two novel unsupervised models that exploit, in a principled way, the fine-grained conversational structure beyond the lex-

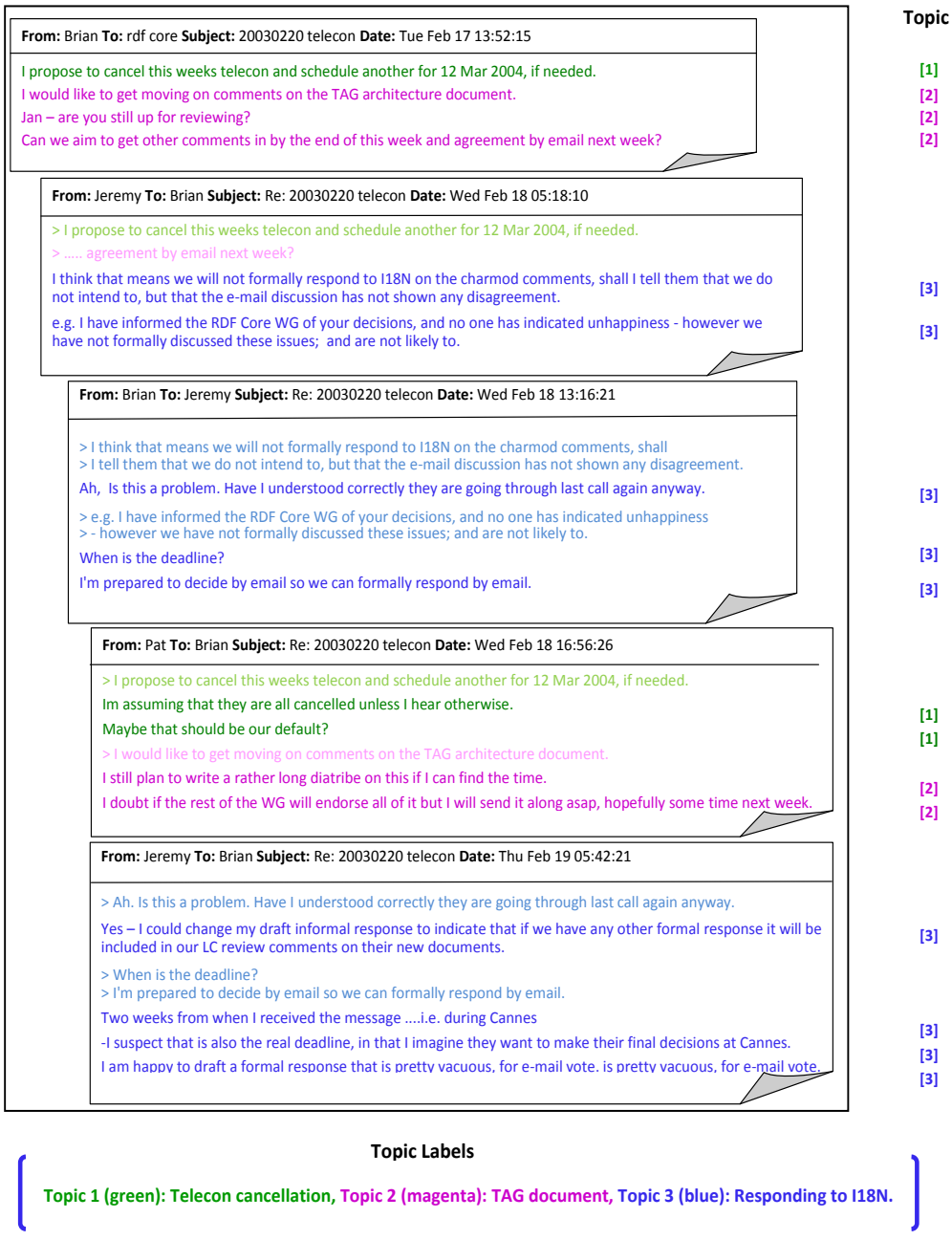



**Figure 2.1:** Sample truncated email conversation from our email corpus. Each color indicates a different topic. The right most column specifies the topic assignments for the sentences.


**Figure 2.2:** Sample truncated blog conversation from our blog corpus. Each color indicates a different topic. The right most column (Topic) specifies the topic assignments for the sentences. The Fragment column specifies the fragments in the fragment quotation graph (see Section 2.3.1).

ical information. We also propose a novel graph-theoretic supervised segmentation model that combines lexical, conversational, and topic features. For topic labeling, we propose to generate labels using an unsupervised extractive approach that identifies the most representative phrases in the text. Specifically, we propose two novel random walk models that respectively capture two forms of conversation specific information: (i) the fact that the leading (i.e., first few) sentences in a topical cluster often carry the most informative clues, and (ii) the fine-grained conversational structure. To our knowledge, this is also the first comprehensive study to address the problem of topic segmentation and labeling in asynchronous conversation.

Our framework was tested in a series of experiments. Experimental results in the topic segmentation task show that the unsupervised segmentation models benefit when they consider the finer conversational structure of asynchronous conversations. A comparison of the supervised segmentation model with the unsupervised models reveals that the supervised method, by optimizing the relative weights of the features, outperforms the unsupervised ones even using only a few labeled conversations. Remarkably, the segmentation decisions of the best unsupervised and the supervised models are also highly correlated with human annotations. As for the experiments on topic labeling, they show that the random walk model performs better when it exploits the conversation specific clues from the leading sentences and the conversational structure. The evaluation of the end-to-end system also shows promising results in both corpora, when compared with human annotations.

The remainder of this chapter is structured as follows. After discussing related work in Section 2.2, we present our topic segmentation and labeling models in Section 2.3. We then describe our corpora and evaluation metrics in Section 2.4. The experiments and analysis are presented in Section 2.5. We summarize our contributions and consider directions for future work in Section 2.6.

## 2.2 Related Work

Three research areas are directly related to our study: topic segmentation, topic labeling, and extracting the conversation structure of asynchronous conversations.

### 2.2.1 Topic Segmentation

Topic segmentation has been extensively studied both for monologs and synchronous dialogs where the task is to divide the discourse into topically coherent **sequential** segments. Several unsupervised and supervised methods have been proposed. See the survey paper by Purver [165] for an excellent overview. The unsupervised models primarily exploit the strong correlation between topic and lexical usage. These models can be categorized into two broad classes based on their underlying intuitions: similarity-based models and probabilistic generative models.

The key intuition behind **similarity-based models** is that sentences in a segment are more lexically similar to each other than to sentences in the preceding or the following segment. These approaches differ in: (1) how they measure lexical similarity, and (2) how they use the similarity measures to perform segmentation.

One such early approach is TextTiling [83], which still forms the baseline for many recent advancements. It operates in three steps: tokenization, lexical score determination, and depth score computation. In the tokenization step, it forms pseudo-sentences, which are fixed length sentences, each containing $n$ stemmed words. Then it considers blocks of $k$ pseudo-sentences, and for each gap between two consecutive pseudo-sentences it measures the cosine-based lexical similarity between the adjacent blocks by representing them as term frequency (TF) vectors. Finally, it measures the depth of the similarity valley for each gap, and assigns the topic boundaries at the appropriate sentence gaps based on a threshold.

When similarity is computed only on the basis of TF vectors, it can cause problems because of sparseness, and because it treats the terms independently. Choi et al. [44] use Latent Semantic Analysis (LSA) to measure the sentence similarity and show that LSA-based similarity outperforms TF-based similarity. Unlike TextTiling, which uses a threshold to decide on topic boundaries, Choi et al. [44] use divisive clustering to find topical segments. We use similarity measures based on both TF and LSA as features in our supervised segmentation model.

Another variation of the cohesion-based approach is LCSeg [74], which uses lexical chains [143]. LCSeg first finds the lexical chains based on term repetitions, and weights those based on term frequency and chain length. The cosine similarity between two adjacent blocks' lexical chain vectors is then used as a measure of lex-

ical cohesion in a TextTiling-like algorithm to find the segments. LCSeg achieves results comparable to the previous approaches (e.g., [44]) in both monolog (news article) and synchronous dialog (meeting). Galley et al. [74] also propose a supervised model for segmenting meeting transcripts. They use a C4.5 probabilistic classifier with lexical and conversational features and show that it outperforms the unsupervised method (LCSeg).

Hsueh et al. [90] apply the models of [74] to both the manual transcripts and the ASR (automatic speech recognizer) output of meetings. They perform segmentation at both coarse (topic) and fine (subtopic) levels. At the topic level, they get similar results as [74]– the supervised model outperforming LCSeg. However, at the subtopic level, LCSeg surprisingly outperforms the supervised model indicating that finer topic shifts are better characterized by lexical similarity alone.

In our work, we initially show how LCSeg performs poorly, when applied to the temporal ordering of asynchronous conversation. This is because, as we mentioned earlier, topics in asynchronous conversations often do not change sequentially following the temporal order of the sentences. To address this, we propose a novel extension of LCSeg that leverages the fine conversational structure of asynchronous conversations. We also propose a novel supervised topic segmentation model for asynchronous conversation that achieves even higher segmentation accuracy by combining lexical, conversational, and topic features.

Malioutov and Barzilay [121] use a minimum cut clustering model to segment spoken lectures (i.e., spoken monolog). They form a weighted undirected graph where the nodes represent sentences and the weighted edges represent the TF.IDF-based cosine similarity between the sentences. Then the segmentation can be solved as a graph partitioning problem with the assumption that the sentences in a segment should be similar, while sentences in different segments should be dissimilar. They optimize the normalized cut criterion [185] to extract the segments. In general, the minimization of the normalized cut criterion is NP-complete. However, the sequentiality constraint of topic segmentation in monolog allows them to find an exact solution in polynomial time. Their approach performs better than [44] in the corpus of spoken lectures. Since the sequentiality constraint does not hold in asynchronous conversation, we implement this model without this constraint by approximating the solution, and compare it with our models.

**Probabilistic generative models** form another class of unsupervised segmentation models, which are based on the intuition that a discourse is a *hidden sequence* of topics, each of which has its own characteristic word distribution. The distribution changes with the change of a topic. Topic segmentation in these models is the task to infer the most likely sequence of topics given the observed words. Variants of **Hidden Markov Models (HMMs)** and **Latent Dirichlet Allocations (LDAs)** [26] are proposed for topic segmentation in monolog and synchronous dialog.

Blei and Moreno [27] propose an aspect Hidden Markov Model (AHMM) to perform topic segmentation in written and spoken (i.e., transcribed) monologs, and show that the AHMM model outperforms the HMM for this task. Purver et al. [167] propose a variant of LDA for segmenting meeting transcripts, and use the top words in the topic-word distributions as topic labels. However, their approach does not outperform LCSeg. Eisenstein and Barzilay [62] propose another variant of LDA by incorporating cue words into the (sequential) segmentation model. In a follow-up work, Eisenstein [61] proposes a constrained LDA model that uses *multi-scale lexical cohesion* to perform hierarchical topic segmentation. Nguyen et al. [153] successfully incorporate speaker identity into a hierarchical nonparametric model for segmenting multi-party synchronous conversations (e.g., meetings, debates). In our work, we demonstrate how the general LDA model performs for topic segmentation in asynchronous conversation and propose a novel extension of LDA that exploits the fine conversational structure.

### 2.2.2 Topic Labeling

In the first comprehensive approach to topic labeling, Mei et al. [137] propose methods to label a multinomial topic model (e.g., the topic-word distributions returned by LDA). Crucial to their approach is how they measure the semantic similarity between a topic-word distribution and a candidate label extracted from the same corpus. They perform this task by assuming another word distribution for the label and deriving the Kullback-Leibler (KL) divergence between the two distributions. It turns out that this measure is equivalent to the weighted point-wise mutual information (PMI) of the topic-words with the candidate label, where the weights are actually the probabilities in the topic-word distribution. They use Maximum

Marginal Relevance (MMR) [33] to select the labels which are relevant, but not redundant. When labeling multiple topic-word distributions, to find discriminative labels, they adjust the semantic similarity scoring function such that a candidate label which is also similar to other topics gets a lower score. In our work, we also use MMR to promote diversity in the labels for a topic. However, to get distinguishable labels for different topical segments in a conversation, we rank the words so that a high scoring word in one topic should not have high scores in other topics.

Recently, Lau et al. [111] propose methods to learn topic labels from Wikipedia titles. They use the top-10 words in each topic-word distribution to extract the candidate labels from Wikipedia. Then they extract a number of features to represent each candidate label. The features are actually different metrics used in previous studies to measure the association between the topic words and the candidate label (e.g., PMI, t-test, $\chi^2$ test). They use Amazon Mechanical Turk to get human annotators to rank the top-10 candidate labels, and use the average scores (given by the annotators) to learn a (supervised) regression model. In a related work, Feng et al. [68] consider the task of classifying coarse discussions based on topics. They induce a topic profile (i.e., a list of candidate topics) from the coarse textbook and use a Rocchio-based classifier [123] to classify the discussion threads.

Zhao et al. [236] addresses the problem of topical keyphrase extraction from Twitter. Initially they use a modified Twitter-LDA model [237], which assumes a single topic assignment for a tweet, to discover the topics (i.e., topic-word distributions) in a corpus of Twitter conversations. Then, they use a PageRank [155] to rank the words in each topic-word distribution. Finally, they perform a bi-gram test to generate keyphrases from the top ranked words in each topic.

While most of the above studies try to mine topics from the whole corpus, our problem is to find the topical segments and label those for a given conversation, where topics are closely related and distributional variations are subtle (e.g., 'Game contents or size', 'Game design' in Figure 2.2). Therefore, statistical association metrics like PMI, the t-test or the chi-square test may not be reliable in our case because of data scarcity. Also at the conversation-level, the topics are so specific to a particular discussion (e.g., 'Telecon cancellation', 'TAG document', 'Responding to I18N' in Figure 2.1) that exploiting external knowledge bases like Wikipedia as a source of candidate labels is not a reasonable option for us. In fact, none of

the human-authored labels in our developement set appears in Wikipedia as a title. Therefore, we propose to generate topic labels using a keyphrase extraction method that finds the most representative phrase(s) in the given text.

Several supervised and unsupervised methods have been proposed for keyphrase extraction (see [134] for a comprehensive overview). The supervised models (e.g., [91, 135]) follow the same two-stage framework. First, candidate keyphrases are extracted using n-gram sequences or a shallow parser (chunker). Second, a classifier filters the candidates. This strategy has been quite successful, but it is domain specific and labor intensive. Every new domain may require new annotations, which at times becomes too expensive and unrealistic. In contrast, our approach is to adopt an unsupervised paradigm, which is more robust across new domains, but still capable of achieving comparable performance to the supervised methods.

Mihalcea and Tarau [140] use a graph-based (unsupervised) random walk model to extract keyphrases from journal abstracts and achieve state-of-the-art performance [139].[4] However, this model is generic and not designed to exploit properties of asynchronous conversations. We propose two novel random walk models to incorporate conversation specific information. Specifically, our models exploit information from two different sources: (i) from the leading sentences of the topical segments, and (ii) from the fine conversational structure of the conversation.

### 2.2.3 Conversational Structure Extraction

As mentioned earlier, there could be several simultaneous conversations (threads) going on in a multi-party interaction. Recent work on synchronous conversations has been focused on disentangling multi-party chat which has a linear structure. Shen et al. [184] and Wang and Oard [223] follow a similar approach. They apply a *single-pass clustering* method which takes a new utterance and measures its distance to the threads already detected, and based on a predefined threshold assigns the new utterance either to the closest thread or to a newly created thread. To measure the distance, they augment the traditional TF.IDF-based cosine similarity with additional features computed from labeled data. Shen et al. [184] include two additional linguistic features, namely utterance type (e.g., declarative, interrogative)

[4]This research was published before in [140].

and usage of personal pronouns, and Wang and Oard [223] include temporal (i.e., time) and social (i.e., mentioning names) contexts in their measures of distance.

Elsner and Charniak [63] propose a two-step method for disentangling multi-party chats. In the first step, a binary classifier determines how likely is that a pair of utterances belong to the same conversation. In the second step, a (graph-based) correlation clustering method [15] finds the conversations by optimizing a criterion that tries to make sure that pairs of utterances likely to belong to the same conversation, end up in the same conversation, while pairs of utterances that are likely to be in different conversations, end up in different conversations. They use three types of features in their classifier: *chat-specific, discourse* and *content*. In their follow-up work [64], they experiment with various local coherence models (e.g., entity grid [18]) for the disentanglement task and improve on their prior work.

Mayfield et al. [131] propose methods to extract a hierarchical structure from multi-party chat. The structure consists of information at three different levels: utterance, sequence and thread. They use a classifier to tag utterances as either giving or receiving information. For detecting the sequences and threads, they use a number of classifiers with constraints in an Integer Linear Programming framework. Eshghi and Healey [66] show the existence of fine-grained dialog contexts in a conversation. They present evidence that shows how these fine-grained dialogue contexts are distinguished not in terms of topics but in terms of active participants.

While disentanglement is necessary for many multi-party synchronous conversations, asynchronous media like email and social media services (e.g., Gmail, Slashdot, Twitter) generally organize comments into tree-structured threads using reply-to relations. In absence of the reply-to relations, automatic methods to uncover the thread structure have also been proposed. Wang et al. [224] propose unsupervised methods based on lexical similarity and proximity relations between messages to uncover the hidden thread structure of newsgroup discussions. More recent work proposes supervised methods using classifiers and sequence labelers (e.g., Decision Trees, CRFs) with a number of useful features [11, 220].

While the above approaches attempt to uncover the reply-to relations between messages, the use of quotations in asynchronous conversations can express a conversational structure that is finer grained and can be more informative than the one revealed by reply-to relations [36]. For example, consider the relation between the

new text fragments and the quoted text fragments (i.e., marked with the quotation mark '>') in figures 2.1 and 2.2. The proximity between quoted and new text fragments can represent a conversational link between the two (i.e., they talk about the same topic) that would not appear by only looking at the reply-to relations.

Carenini et al. [34] previously presented a novel method to capture an email conversation at this finer level by analyzing the embedded quotations in emails. A Fragment Quotation Graph (FQG) was formed, which was shown to be beneficial for email summarization [35] and dialog act modeling [95]. In this work, we generalize the FQG to any asynchronous conversation and demonstrate that topic segmentation and labeling models can also benefit significantly from this fine conversational structure of asynchronous conversation.

## 2.3 Topic Models for Asynchronous Conversations

Developing topic segmentation and labeling models for asynchronous conversations is challenging partly because of the specific characteristics of these media. As mentioned earlier, unlike monolog (e.g., articles) and synchronous dialog (e.g., meetings), topics in asynchronous conversations may not change in a sequential way, with topics being interleaved. Furthermore, as can be noticed in figures 2.1 and 2.2, writing style varies among participants, and many people tend to use informal, short and ungrammatical sentences, thus making the discourse much less structured. One aspect of asynchronous conversation that at first glance may appear to help topic modeling is that each message comes with a header. However, often headers do not convey much topical information and sometimes they can even be misleading. For example, in the blog conversation (Figure 2.2), participants keep talking about different topics using the same title (i.e., 'nice nice nice'), which does not convey any topic information. Arguably, all these unique properties of asynchronous conversations limit the application of state-of-the-art techniques that have been successful in monolog and synchronous dialog. Below, we first describe these techniques and then we present how we have extended them to effectively deal with asynchronous conversations.

### 2.3.1 Topic Segmentation Models

We are the first to study the problem of topic segmentation in asynchronous conversation. Therefore, we first show how existing models, which are originally developed for monolog and synchronous dialog, can be naively applied to asynchronous conversations. Then, by pointing out their limitations, we propose our novel topic segmentation models for asynchronous conversations.

#### Existing Models

**LCSeg** [74] and **LDA** [26] are two state-of-the-art (unsupervised) models for topic segmentation in monolog and synchronous dialog [165]. In the following, we briefly describe these models and how they can be directly applied to asynchronous conversations.

*Lexical Cohesion-based Segmenter (LCSeg)*

LCSeg is a sequential segmentation model originally developed for segmenting meeting transcripts. It exploits the linguistic property called **lexical cohesion**, and assumes that topic changes are likely to occur where strong word repetitions start and end. It first computes **lexical chains** [143] for each non-stop word based on word repetitions.[5] Then the chains are weighted according to their term frequency and the chain length. The more populated and compact chains get higher scores. The algorithm then works with two adjacent analysis windows, each of a fixed size $k$, which is empirically determined. At each sentence boundary, it computes the cosine similarity (or lexical cohesion function) between the two windows by representing each window as a vector of chain-scores of its words. Specifically, the lexical cohesion between windows ($X$ and $Y$) is computed with:

$$LexCoh(X,Y) = cos\_sim(X,Y) = \frac{\sum_{i=1}^{N} w_{i,X}.w_{i,Y}}{\sqrt{\sum_{i=1}^{N} w_{i,X}^2 . \sum_{i=1}^{N} w_{i,Y}^2}} \tag{2.1}$$

where $N$ is the number of chains and

[5] One can also consider other lexical semantic relations (e.g., synonym, hypernym) in lexical chaining but the best results account for only repetition.

$$w_{i,\Omega} = \begin{cases} rank(C_i) & \text{if chain } C_i \text{ overlaps } \Omega \in \{X, Y\} \\ 0 & \text{otherwise} \end{cases}$$

A sharp change at local minima in the resulting similarity (or lexical cohesion) curve signals a high probability of a topic boundary. The curve is smoothed, and for each local minimum a segmentation probability is computed based on its relative depth below its nearest peaks on either side. Points with the highest segmentation probability are then selected as hypothesized topic boundaries. This method is very similar to TextTiling [83] except that the similarity is computed based on the scores of the chains instead of term frequencies.

LCSeg can be directly applied to an asynchronous conversation by arranging components (e.g., emails, posts, tweets) based on their arrival time (i.e., their temporal order) and running the algorithm to get the topic boundaries.

*Latent Dirichlet Allocation (LDA)*

LDA is a generative model that relies on the fundamental idea that documents are admixtures of topics, and a topic is a multinomial distribution over words. It specifies the following distribution over words within a document:

$$P(x_{ij}) = \sum_{k=1}^{K} P(x_{ij}|z_{ij} = k, \boldsymbol{b_k}) P(z_{ij} = k|\boldsymbol{\pi_i}) \tag{2.2}$$

where $K$ is the number of topics, $P(x_{ij}|z_{ij} = k, \boldsymbol{b_k})$ is the probability of word $x_{ij}$ in document $i$ for topic $k$, and $P(z_{ij} = k|\boldsymbol{\pi_i})$ is the probability that the $k^{th}$ topic was sampled for the word token $x_{ij}$. We refer to the multinomial distributions $\boldsymbol{b_k}$ and $\boldsymbol{\pi_i}$ as topic-word and document-topic distributions, respectively. Figure 2.3 shows the resultant graphical model in plate notation for $N$ documents, $K$ topics and $M_i$ tokens in each document $i$. Note that, $\alpha$ and $\beta$ are the standard Dirichlet priors on $\boldsymbol{\pi_i}$ and $\boldsymbol{b_k}$, respectively. Variational EM can be used to estimate $\boldsymbol{\pi}$ and $\boldsymbol{b}$ [26]. One can also use Gibbs sampling to directly estimate the posterior distribution over $z$, i.e., $P(z_{ij} = k|x_{ij})$; namely, the topic assignments for word tokens [199].

This model can be directly applied to an asynchronous conversation by considering each comment as a document. By assuming the words in a sentence occur

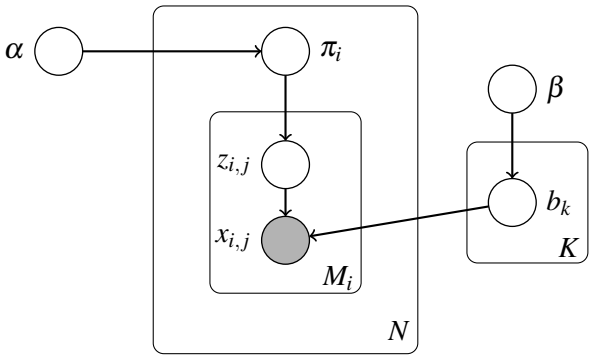



**Figure 2.3:** Graphical model for LDA in plate notation.

independently we can estimate the topic assignments for each sentence $s$ as:

$$P(z_m = k|s) = \prod_{x_m \in s} P(z_m = k|x_m) \tag{2.3}$$

Finally, the topic for sentence $s$ can be assigned by:

$$k^* = argmax_k \, P(z_m = k|s) \tag{2.4}$$

### Limitations of Existing Models

The main limitation of the two models discussed above is that they make the bag-of-words (BOW) assumption, ignoring facts that are specific to a multi-party, asynchronous conversation. LCSeg considers only term frequency and how closely these terms occur in the temporal order of the sentences. If topics are interleaved and do not change sequentially in the temporal order, as is often the case in asynchronous conversations, then LCSeg would fail to find the topic segments correctly.

On the other hand, the only information relevant to LDA is term frequency. Several extensions of LDA over the BOW approach have been proposed. For example, Wallach [219] extends the model beyond BOW by considering n-gram sequences. Griffiths et al. [76] present an extension that is sensitive to word-order and automatically learns the syntactic as well as semantic factors that guide word choice. Boyd-Graber and Blei [29] describe another extension to consider the syntax of the sentences.

We argue that these models are still inadequate for finding topical segments correctly in asynchronous conversations especially when topics are closely related and their distributional variations are subtle (e.g., 'Game contents or size' and 'Game design'). To better identify the topics one needs to consider the features specific to asynchronous conversations (e.g., conversation structure, speaker, recipient). In the following, we propose our novel unsupervised and supervised topic segmentation models that incorporate these features.

### Proposed Unsupervised Models

One of the most important indicators for topic segmentation in asynchronous conversation is its conversation structure. As can be seen in the examples (figures 2.1 and 2.2), participants often reply to a post and/or use quotations to talk about the same topic. Notice also that the use of quotations can express a conversational structure that is at a finer level of granularity than the one revealed by reply-to relations. In our corpora, we found an average quotation usage of 9.85 per blog conversation and 6.44 per email conversation. Therefore, we need to leverage this key information to get the best out of our models. Specifically, we need to capture the conversation structure at the quotation (i.e., text fragment) level, and to incorporate this structure into our segmentation models in a principled way.

In the following, we first describe how we can capture the conversation structure at the fragment level. Then we show how the unsupervised segmentation models LCSeg and LDA can be extended to take this conversation structure into account, generating two novel unsupervised models for topic segmentation.

#### *Extracting Finer-level Conversation Structure*

Since consecutive turns in asynchronous conversations can be far apart in time, when participants reply to a post or comment, a quoted version of the original message is often included (specially in email) by default in the draft reply in order to preserve context. Furthermore, people tend to break down the quoted message so that different questions, requests or claims can be dealt with separately. As a result, each message, unless it is at the beginning, will contain a mix of quoted and novel paragraphs (or fragments) that may well reflect a reply-to relationship between

paragraphs that is at a finer level of granularity than the one explicitly recorded between messages. Carenini et al. [34] propose a method to capture this finer-level conversation structure in the form of a graph called **Fragment Quotation Graph (FQG)**. Below, we demonstrate how to build a FQG for the sample blog conversation shown in Figure 2.2 following [34].

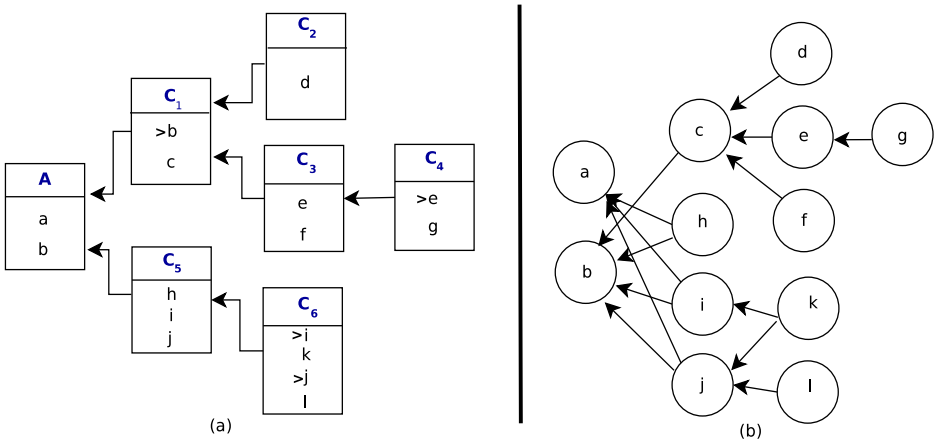



**Figure 2.4:** (a) The main *A*rticle and the *C*omments with the fragments for the example in Figure 2.2. Arrows indicate 'reply-to' relations. (b) The corresponding Fragment Quotation Graph (FQG).

Figure 2.4 (a) shows the same blog conversation, but for the sake of illustration, instead of showing the real content, we abbreviate it as a sequence of labels (e.g., $a, b$), each label corresponding to a text fragment (see the Fragment column in Figure 2.2). Building a FQG is a two-step process.

- *Node creation:* Initially, by processing the whole conversation, we identify the new and the quoted fragments of different depth levels. The depth level of a quoted fragment is determined by the number of quotation marks (e.g., $>$, $>>$). For instance, comment $C_1$ contains a new fragment $c$ and a quoted fragment $b$ of depth level 1. $C_6$ contains two new fragments $k$ and $l$, and two quoted fragments $i$ and $j$ of depth level 1, and so on. Then in the second step, we compare the fragments with each other and based on their lexical overlap we find the distinct fragments. If necessary, we split the fragments in this step. For example, $ef$ in $C_3$ is divided into distinct fragments $e$ and

$f$ when compared with the fragments of $C_4$. This process gives 12 distinct fragments which constitute the nodes of the FQG shown in Figure 2.4(b).

- *Edge creation:* We create edges to represent likely replying relationships between fragments assuming that any new fragment is a potential reply to its neighboring quotations of depth level 1. For example, for the fragments of $C_6$ in Figure 2.4(a), we create two edges from $k$ (i.e., (k,i),(k,j)) and one edge from $l$ (i.e., (l,j)) in Figure 2.4(b). If a comment does not contain any quotation, then its fragments are linked to the new fragments of the comment to which it replies, capturing the original reply-to relation between comments.

Note that the FQG is only an approximation of the reply relations between fragments. In some cases, proximity may not indicate any connection and in other cases a connection can exist between fragments that are never adjacent in any comment. Furthermore, this process could lead to less accurate conversational structure when quotation marks (or cues) are not present. Nonetheless, Carenini et al. [35] showed that considering the FQG can be beneficial for email summarization, and recently, we showed its benefits in unsupervised dialog act modeling [95] (see Chapter 4). In this chapter, we show that topic segmentation (this section) and labeling (Section 2.3.2) models can also benefit significantly from this fine conversational structure. Minimizing the noise in FQGs is left as future work.

*LCSeg with FQG (LCSeg+FQG)*

If we examine the FQG carefully, the paths (considering the fragments of the first comment as root nodes) can be interpreted as subconversations, and topic shifts are likely to occur along the pathway as we walk down a path. We incorporate FQG into LCSeg in three steps.

- *Path extraction:* First, we extract all the paths of a FQG. For example, for the FQG in Figure 2.4(b), we extract the paths $< a, j, l >$, $< b, c, e, g >$, $< b, c, d >$, and so on.

- *LCSeg application:* We then run the LCSeg algorithm on each of the extracted paths separately and collect the segmentations. For example, when

we apply LCSeg to $< b, c, e, g >$ and $< b, c, d >$ paths in Figure 2.4(b) separately, we may get the following segmentations— $< b, c \mid e, g >$ and $< b, c \mid d >$, where '|' denotes a segment boundary.[6] Notice that a fragment can be in multiple paths (e.g., *b*, *c*) which will eventually cause its sentences to be in multiple segments. So, in the final step, we need a consolidation method.

- *Consolidation:* Our intuition is that sentences in a consolidated segment should appear together in a segment more often when LCSeg is applied in step 2, and if they do not appear together in any segment, they should at least be similar. To achieve this, we construct a weighted undirected graph $G(V,E)$, where the nodes $V$ represent the sentences and the edge weights $w(x,y)$ represent the number of segments in which sentences $x$ and $y$ appear together; if $x$ and $y$ do not appear together in any segment, then their cosine similarity is used as edge weight. More formally,

$$w(x,y) = \begin{cases} n, & \text{if } x \text{ and } y \text{ appear together in } n \text{ segments and } n>0 \\ cos\_sim(x,y), & \text{if } n = 0 \end{cases}$$

We measure the cosine similarity between sentences $x$ and $y$ as follows:

$$cos\_sim(x,y) = \frac{\sum_{w \in x,y} tf_{w,x}.tf_{w,y}}{\sqrt{\sum_{x_i \in x} tf^2_{x_i,x}} \cdot \sqrt{\sum_{y_i \in y} tf^2_{y_i,y}}} \tag{2.5}$$

where $tf_{a,s}$ denotes the term frequency of term $a$ in sentence $s$. The cosine similarity ($0 \leq cos\_sim(x,y) \leq 1$) provides informative edge weights for the sentence pairs that are not directly connected by LCSeg segmentation decisions.[7] The consolidation problem can be formulated as a **k-way-mincut** graph partitioning problem with the **normalized cut (Ncut)** criterion [185]:

[6] For convenience, we are showing the segmentations at the fragment level, but the segmentations are actually at the sentence level.

[7] Our work presented in [94] did not consider the cosine similarity when two sentences do not appear together in any of the segments. However, later we found out that including the cosine similarity offers more than 2% absolute gain in segmentation performance.

$$Ncut_k(V) = \frac{cut(A_1, V - A_1)}{assoc(A_1, V)} + \frac{cut(A_2, V - A_2)}{assoc(A_2, V)} + \cdots + \frac{cut(A_k, V - A_k)}{assoc(A_k, V)} \tag{2.6}$$

where $A_1, A_2 \cdots A_k$ form a partition (i.e., disjoint sets of nodes) of the graph, and $V - A_k$ is the set difference between $V$ (i.e., set of all nodes) and $A_k$. The $cut(A,B)$ measures the total edge weight from the nodes in set $A$ to the nodes in set $B$, and $assoc(A,V)$ measures the total edge weight from the nodes in set $A$ to all nodes in the graph. More formally:

$$cut(A,B) = \sum_{u \in A, v \in B} w(u,v) \tag{2.7}$$

$$assoc(A,V) = \sum_{u \in A, t \in V} w(u,t) \tag{2.8}$$

Note that the partitioning problem can be solved using any correlation clustering method (e.g., [15]). Previous work on graph-based topic segmentation [121] has shown that the Ncut criterion is more appropriate than just the cut criterion, which accounts only for total edge weight connecting $A$ and $B$, and therefore, favors cutting small sets of isolated nodes in the graph. However, solving Ncut is NP-complete. Hence, we approximate the solution following the method proposed in [185], which is time efficient and has been successfully applied to image segmentation in computer vision.

Notice that this approach makes a difference only if the FQG of the conversation contains more than one path. In fact, in our corpora we found an average number of paths of 7.12 and 16.43 per email and blog conversations, respectively.

*LDA with FQG (LDA+FQG)*

A key advantage of probabilistic Bayesian models, such as LDA, is that they allow us to incorporate multiple knowledge sources in a coherent way in the form of priors (or regularizer). To incorporate FQG into LDA, we propose to regularize LDA so that two sentences in the same or adjacent fragments are likely to appear

in the same topical cluster. The first step towards this aim is to regularize the topic-word distributions (i.e., $\boldsymbol{b}$ in Figure 2.3) with a **word network** such that two connected words get similar topic distributions.

For now, assume that we are given a word network as an undirected graph $G(V,E)$, with nodes $V$ representing the words and the edges $(u,v) \in E$ representing the links between words $u$ and $v$. We want to regularize the topic-word distributions of LDA such that two connected words $u$ and $v$ in the word network have similar topic distributions (i.e., $\boldsymbol{b}_{\boldsymbol{k}}^{(u)} \approx \boldsymbol{b}_{\boldsymbol{k}}^{(v)}$ for $k = 1 \ldots K$). The standard conjugate Dirichlet prior $\text{Dir}(\boldsymbol{b}_{\boldsymbol{k}}|\beta)$, however does not allow us to do that, because here all words share a common variance parameter, and are mutually independent except for the normalization constraint [141]. Recently, Andrzejewski et al. [6] describe a method to encode **must-links** and **cannot-links** between words using a Dirichlet Forest prior. Must-link between two words enforces the words to have similar distributions over topics. On the other hand, cannot-link between two words enforces the words not to both have large probability within any topic, although it is allowed for one to have a large probability and the other small, or both small [6].

Our goal is just to encode the must-links. Therefore, we reimplemented their model with its capability of encoding just the (must-)links. Must-links between words such as $(a,b),(b,c)$, or $(x,y)$ in Figure 2.5(a) can be encoded into LDA using a **Dirichlet Tree (DT)** prior. Like the traditional Dirichlet, the DT prior is also a conjugate to the multinomial, but under a different parameterization. Instead of representing a multinomial sample as the outcome of a K-sided die, in the tree representation (e.g., Figure 2.5(b)), a sample (i.e., a leaf in the tree) is represented as the outcome of a finite stochastic process. The probability of a leaf (i.e., a word in our case) is the product of branch probabilities leading to that leaf. A DT prior is the distribution over leaf probabilities.

Let $\omega^n$ be the edge weight leading into node $n$, $C(n)$ be the children of node $n$, $L$ be the leaves of the tree, $I$ be the internal nodes, and $L(n)$ be the leaves in the subtree under node $n$. We generate a sample $\boldsymbol{b}_{\boldsymbol{k}}$ from $\text{DT}(\omega)$ by drawing a multinomial at each internal node $i \in I$ from $\text{Dir}(\omega^{C(i)})$ (i.e., the edge weights from node $i$ to its children). The probability density function of $\text{DT}(\boldsymbol{b}_{\boldsymbol{k}}|\omega)$ is given by:

$$DT(\boldsymbol{b_k}|\boldsymbol{\omega}) \propto \left(\prod_{l\in L} b_l^{k^{\omega^l}-1}\right)\left(\prod_{i\in I}\left(\sum_{j\in L(i)} b_j^k\right)^{\Delta(i)}\right) \tag{2.9}$$

where $\Delta(i) = \omega^i - \sum_{j\in C(i)} \omega^j$, the difference between the in-degree and out-degree of an internal node $i$. Notice when $\Delta(i) = 0$ for all $i \in I$, the Dirichlet tree distribution reduces to the standard Dirichlet distribution.

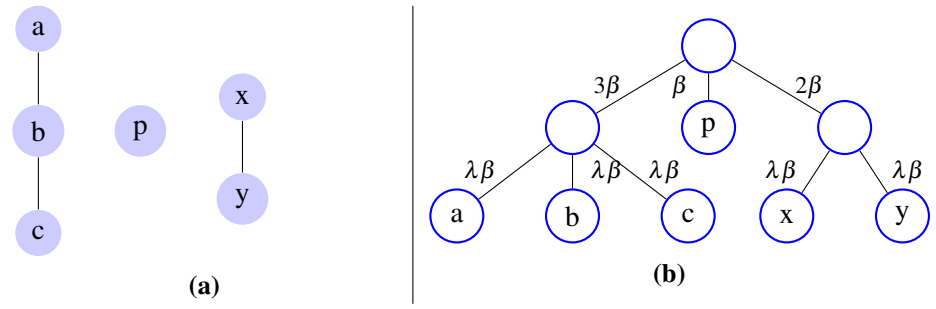



**Figure 2.5:** (a) Sample word network, (b) A Dirichlet Tree (DT) built from such word network.

Suppose we are given the word network as shown in Figure 2.5(a). The network can be decomposed into a collection of chains (e.g., $(a,b,c)$, $(p)$, and $(x,y)$). For each chain containing multiple elements (e.g., $(a,b,c)$, $(x,y)$), there is a subtree in the DT (Figure 2.5(b)), with one internal node (blank in Figure) and the words of the chain as its leaves. The weight from the internal node to each of its leaves is $\lambda\beta$, where $\lambda$ is the regularization strength and $\beta$ is the parameter of the standard symmetric Dirichlet prior on $\boldsymbol{b_k}$. The root node of the DT then connects to the internal nodes with $|L(i)|\beta$ weight. The leaves (words) for the single element chains (e.g, (p)) are then connected to the root of the DT directly with weight $\beta$. Notice that when $\lambda = 1$, $\Delta(i) = 0$, and it reduces to the standard LDA (i.e., no regularization). By tuning $\lambda$ we control the strength of the regularization.

At this point what is left to be explained is how we construct the word network. To regularize LDA with a FQG, we construct a word network where a word is linked to the words in the same or adjacent fragments in the FQG. Specifically, if word $w_i \in frag_x$ and word $w_j \in frag_y$ and $w_i \neq w_j$, we create a link $(w_i, w_j)$ if $x = y$ or $(x,y) \in E_{fqg}$, where $E_{fqg}$ is the set of edges in the FQG. This implicitly compels

two sentences in the same or adjacent fragments to have similar topic distributions, and to appear in the same topical segment.

### Proposed Supervised Model

Although the unsupervised models discussed in the previous section have the key advantage of not requiring any labeled data, they can be limited in their ability to learn domain-specific knowledge from a possibly large and diverse set of features [62]. Beside discourse cohesion, which captures changes in content, there are other important domain-specific distinctive features which signal topic change. For example, discourse markers (or cue phrases) (e.g., *okay, anyway, now, so*) and prosodic cues (e.g., longer pauses) directly provide clues about topic change, and have been shown to be useful features for topic segmentation in monolog and synchronous dialog [74, 159]. We hypothesize that asynchronous conversations can also feature their own distinctive characteristics for topic shifts. For example, features like *sender* and *recipient* are arguably useful for segmenting asynchronous conversations, as different participants can be more or less active during the discussion of different topics. Therefore, as a next step to build an even more accurate topic segmentation model for asynchronous conversations, we propose to combine different sources of possibly useful information in a principled way.

The supervised framework serves as a viable option to combine a large number of features and optimize their relative weights for decision making, but relies on labeled data for training. The amount of labeled data required to achieve an acceptable performance is always an important factor to consider for choosing supervised vs. unsupervised. In this work, we propose a supervised topic segmentation model that outperforms all the unsupervised models, even when it is trained on a small number of labelled conversations.

Our supervised model is built on the **graph-theoretic** framework which has been used in many NLP tasks, including coreference resolution [190] and chat disentanglement [63]. This method works in two steps.

- *Classification:* A binary classifier which is trained on a labeled dataset marks each pair of sentences of a conversation as **same** or **different** topics.
- *Graph partitioning:* A weighted undirected graph $G = (V, E)$ is formed,

where the nodes $V$ represent the sentences in the conversation and the edge-weights $w(x,y)$ denote the probability (given by the classifier) of the two sentences $x$ and $y$ to appear in the *same* topic. Then an optimal partition is extracted from the graph.

*Sentence pair classification*

The classifier's accuracy in deciding whether a pair of sentences $x$ and $y$ is in the **same** or **different** topics is crucial for the model's performance. Note that since each sentence pair of a conversation defines a data point, a conversation containing $n$ sentences produces $1+2+\ldots+(n-1) = \frac{n(n-1)}{2} = O(n^2)$ training examples. Therefore, a training dataset containing $m$ conversations produces $\sum_{i=1}^{m} \frac{n_i(n_i-1)}{2}$ training examples, where $n_i$ is the number of sentences in the $i^{th}$ conversation. This quadratic expansion of training examples enables the classifier to achieve its best classification accuracy with only a few labeled conversations.

By pairing up the sentences of each email conversation in our email corpus, we got a total of $14,528$ data points of which $58.8\%$ are in the *same* class (i.e., *same* is the most likely in email), and by pairing up the sentences of each blog conversation in our blog corpus, we got a total of $572,772$ data points of which $86.3\%$ are in the *different* class (i.e., *different* is the most likely in blog).[8] To select the best classifier, we experimented with a variety of classifiers with the full feature set (Table 2.2). Table 2.1 shows the performance of the classifiers averaged over a **leave-one-out** procedure, i.e., for a corpus containing $m$ conversations, train on $m-1$ conversations and test on the rest.

K-Nearest Neighbor (KNN) performs very poorly. Logistic Regression (LR) with $l_2$ regularization delivers the highest accuracy on both datasets. Support Vector Machines (SVMs) [49] with linear and rbf kernels perform reasonably well, but not as well as LR. The Ridged Multinomial Logistic Regression (RMLR) [109], a kernelized LR, extremely overfits the data. We opted for the LR with $l_2$ regularization because it not only delivers the best performance in term of accuracy, but it is also very efficient. The limited memory BFGS (L-BFGS) fitting algorithm used in LR is efficient in terms of both time (quadratic convergence rate; fastest among the

[8] See Section 2.4 for a detailed description of our corpora. The class labels are produced by taking the maximum vote of the three annotators.

| Classifier | Type | Regularizer | Accuracy **(Blog)** | | Accuracy **(Email)** | |
|---|---|---|---|---|---|---|
| | | | Train | Test | Train | Test |
| KNN | non-parametric | - | 62.7% | 61.4 % | 54.6% | 55.2% |
| **LR** | **parametric** | $\boldsymbol{l_2}$ | **90.8%** | **91.9%** | **71.7%** | **72.5%** |
| LR | parametric | $l_1$ | 86.8% | 87.6% | 69.9% | 67.7% |
| RMLR (rbf) | non-parametric | $l_2$ | 91.7% | 82.0% | 91.1% | 62.1% |
| SVM (lin) | parametric | - | 76.6% | 78.7 % | 68.3% | 69.6% |
| SVM (rbf) | non-parametric | - | 80.5% | 77.9% | 75.9% | 67.7% |
| Majority class | - | - | 86.3% (different) | | 58.8% (same) | |

**Table 2.1:** Performance of the classifiers using the full feature set (Table 2.2). For each training set, regularizer strength $\lambda$ (or C in SVMs) was learned by 10-fold cross validation.

listed models) and space ($O(mD)$, where $m$ is the memory parameter of L-BFGS and $D$ is the number of features).

Table 2.2 summarizes the full feature set and the mean test set accuracy (using leave-one-out) achieved with different types of features in our LR classifier. [9]

**Lexical** features encode similarity between two sentences $x$ and $y$ based on their raw contents. Term frequency-based similarity is a widely used feature in previous work (e.g., TextTiling [83]). We compute this feature by considering two analysis windows, each of fixed size $k$. Let $X$ be the window including sentence $x$ and the preceding $k-1$ sentences, and $Y$ be the window including sentence $y$ and the following $k-1$ sentences. We measure the cosine similarity between the two windows by representing them as vectors of **TF.IDF** [178] values of the words. Another important domain specific feature that proved to be useful in previous research (e.g., [74]) is **cue words** (or discourse markers) that signal the presence of a topic boundary (e.g., 'coming up', 'joining us' in news). Since our work concerns conversations (not monologs), we adopt the cue word list derived automatically from a meeting corpus by Galley et al. [74]. If $y$ answers or greets $x$ then it is likely that they are in the same topic. Therefore, we use the Question Answer (**QA**) pairs and **greeting** words as two other lexical features.

[9] We believe that discourse feature like co-reference resolution would be helpful for topic segmentation. However, to our knowledge, there is no publicly available co-reference resolution system for asynchronous conversation. The existing co-reference resolution system that are developed for monolog are also limited in terms of accuracy.

| | |
|---|---|
| **Lexical** | **Accuracy: 86.8 Precision: 62.4 Recall: 4.6 (Blog)**<br>**Accuracy: 59.6 Precision: 59.7 Recall: 99.8 (Email)** |
| $TFIDF_1$ | TF.IDF-based similarity between $x$ and $y$ with window size k=1. |
| $TFIDF_2$ | TF.IDF-based similarity between $x$ and $y$ with window size k=2. |
| Cue Words | Either $x$ or $y$ contains a cue word. |
| QA | $x$ asks a question explicitly using ? and $y$ answers it using any of (yes, yeah, okay, ok, no, nope). |
| Greet | Either $x$ or $y$ has a greeting word (hi, hello, thanks, thx, tnx, thank.) |
| **Conversation** | **Accuracy: 88.2 Precision: 81.6 Recall: 20.5 (Blog)**<br>**Accuracy: 65.3 Precision: 66.7 Recall: 85.1 (Email)** |
| Gap | The gap between $y$ and $x$ in number of sentence(s). |
| Speaker | $x$ and $y$ have the same sender (yes or no). |
| $FQG_1$ | Distance between $x$ and $y$ in FQG in terms of fragment id. (i.e., $\|frag\_id(y) - frag\_id(x)\|$). |
| $FQG_2$ | Distance between $x$ and $y$ in FQG in terms of number of edges. |
| $FQG_3$ | Distance between $x$ and $y$ in FQG in number of edges but this time considering it as an undirected graph. |
| Same/Reply | whether $x$ and $y$ are in the same comment or one is a reply to the other. |
| Name | $x$ mentions $y$'s speaker or vice versa. |
| **Topic** | **Accuracy: 89.3 Precision: 86.4 Recall: 17.3 (Blog)**<br>**Accuracy: 67.5 Precision: 68.9 Recall: 76.8 (Email)** |
| $LSA_1$ | LSA-based similarity between $x$ and $y$ with window size k=1. |
| $LSA_2$ | LSA-based similarity between $x$ and $y$ with window size k=2. |
| LDA | LDA segmentation decision on $x$ and $y$ (same or different). |
| LDA+FQG | LDA+FQG segmentation decision on $x$ and $y$ (same or different). |
| LCSeg | LCSeg segmentation decision on $x$ and $y$ (same or different). |
| LCSeg+FQG | LCSeg+FQG segmentation decision on $x$ and $y$ (same or different). |
| LexCoh | Lexical cohesion between $x$ and $y$. |
| **Combined** | **Accuracy: 91.9 Precision: 78.8 Recall: 25.8 (Blog)**<br>**Accuracy: 72.5 Precision: 70.4 Recall: 81.5 (Email)** |

**Table 2.2:** Features with performance on test sets (using leave-one-out).

**Conversational** features capture conversational properties of an asynchronous conversation. Time gap and speaker are commonly used features for segmenting synchronous conversation [63, 74]. We encode similar information in asynchronous media by counting the number of sentences between $x$ and $y$ (in their temporal order) as the **gap**, and their senders as the **speakers**. The strongest baseline Speaker (see Section 2.5.1) also proves its effectiveness in asynchronous domains. The results in Section 2.5.1 also suggest that fine conversational structure in the form of FQG can be beneficial when it is incorporated into the unsupervised segmentation models. We encode this valuable information into our supervised segmentation model by computing three distance features on the FQG: $\boldsymbol{FQG_1}, \boldsymbol{FQG_2}$ and $\boldsymbol{FQG_3}$. State-of-the-art email and blog systems use the reply-to relation to group comments into threads. If $y$'s comment is the **same as or reply to** $x$'s comment, then it is likely that the two sentences talk about the same topic. Participants sometimes mention each other's **name** in multi-party conversations to make disentanglement easier [63]. We also use this as a feature in our supervised segmentation model.

**Topic** features are complex and encode topic information from existing segmentation models. Choi et al. [44] used Latent Semantic Analysis (**LSA**) to measure the similarity between two sentences and showed that the LSA-based similarity yields better results than the direct TF.IDF-based similarity since it surmounts the problems of synonymy (e.g., car, auto) and polysemy (e.g., money bank, river bank). To compute LSA, we first construct a word-document matrix $W$ for a conversation, where $W_{i,j}$ = the frequency of word $i$ in comment $j$ $\times$ the IDF score of word $i$. We perform truncated Singular Value Decomposition (SVD) of $W$: $W \approx U_k \Sigma_k V_k^T$, and represent each word $i$ as a $k$ dimensional[10] vector $\mathbf{\Lambda}_i^k$. Each sentence is then represented by the weighted sum of its word vectors. Formally, the LSA representation for sentence $s$ is $\mathbf{\Lambda}_s = \sum_{i \in s} tf_i^s . \mathbf{\Lambda}_i^k$, where $tf_i^s$ = the term frequency of word $i$ in sentence $s$. Then just like the TF.IDF-based similarity, we compute the LSA-based similarity between sentences $x$ and $y$, but this time by representing the corresponding windows (i.e., $X$ and $Y$) as LSA vectors.

The segmentation decisions of the **LDA**, **LDA+FQG**, **LCSeg** and **LCSeg+FQG**

[10]The value of $k$ was empirically set to $\frac{1}{4}\times$number of comments based on our development set.

models described in the previous section are also encoded as topic features.[11] As described in Section 2.3.1, LCSeg computes a lexical cohesion (**LexCoh**) function between two consecutive windows based on the scores of the lexical chains. Galley et al. [74] show a significant improvement when this function is used as a feature in the supervised (sequential) topic segmentation model for meetings. However, since our problem of topic segmentation is not sequential, we want to compute this function for any two given windows $X$ and $Y$ (not necessary consecutive). To do that, we first extract the lexical chains with their scores and spans (i.e., beginning and end sentence numbers) for the conversation. The lexical cohesion function is then computed with the method described in Equation 2.1.

We describe our classifier's performance in terms of raw accuracy (correct decisions/total), precision and recall of the *same* class for different types of features averaged over a leave-one-out procedure (Table 2.2). Among the feature types, topic features yield the highest accuracy and same-class precision in both corpora ($p < 0.01$). Conversational features also have proved to be important and achieve higher accuracy than lexical features ($p < 0.01$). Lexical features have poor accuracy, only slightly higher than the majority baseline that always picks the most likely class. However, when we combine all the features, we get the best performance ($p < 0.005$). These results demonstrate the importance of topical and conversational features beyond the lexical features used by the existing segmentation models. When we compare the performance on the two corpora, we notice that while in blog the accuracy and the same-class precision are higher than in email, the same-class recall is much lower. Although this is reasonable given the class distributions in the two corpora (i.e., 13.7% and 58.8% examples are in the same-class in blog and email, respectively), surprisingly, when we tried to deal with this problem by applying the bagging technique [30], the performance does not improve significantly. Note that some of the classification errors occurred in the sentence-pair classification phase are recovered in the graph partitioning step (see below). The reason is that the incorrect decisions will be outvoted by the nearby sentences that are clustered correctly.

[11] The work presented in [96] did not include the segmentation decisions of the LDA+FQG and LCSeg+FQG models as features. However, including these features improves both classification accuracy and segmentation accuracy.

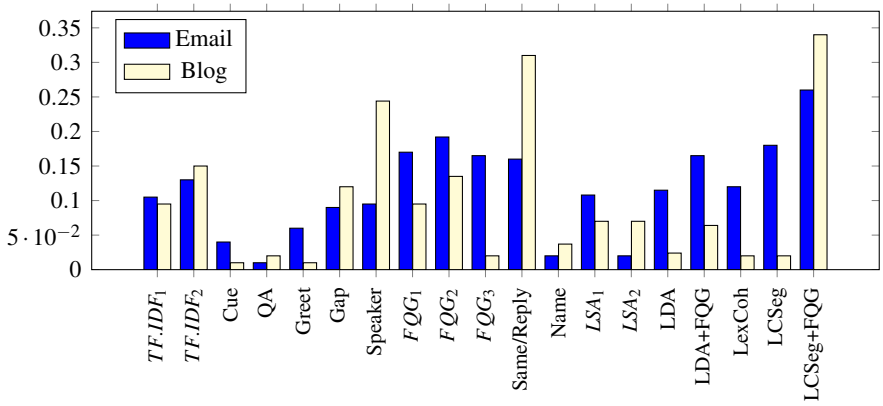



**Figure 2.6:** Relative importance of the features averaged over leave-one-out.

We further analyze the contribution of individual features. Figure 2.6 shows the relative importance of the features based on the absolute values of their coefficients in our LR classifier. The segmentation decision of LCSeg+FQG is the most important feature in both domains. The Same/Reply feature is also an effective feature, especially in blog. In blog, the Speaker feature also plays an important role. The $FQG_2$ feature (i.e., distance in number of edges in the directed FQG) is also effective in both domains, especially in email. The other two features on FQG (i.e., $FQG_1$, $FQG_3$) are also very relevant in email.

Finally, in order to determine how many annotated conversations we need to achieve the best segmentation performance, Figure 2.7 shows the classification error rate (incorrect decisions/total), tested on 5 randomly selected conversations and trained on an increasing number of randomly added conversations. Our classifier appears to achieve its best performance with a small number of labeled conversations. For blog, the error rate flattens with only 8 conversations, while for email, this happens with about 15. This is not surprising since blog conversations are much longer (average of 220.55 sentences) than email conversations (average of 26.3 sentences), generating a similar number of training examples with only a few conversations (a conversation with $n$ sentences produces $O(n^2)$ training examples).

*Graph partitioning*

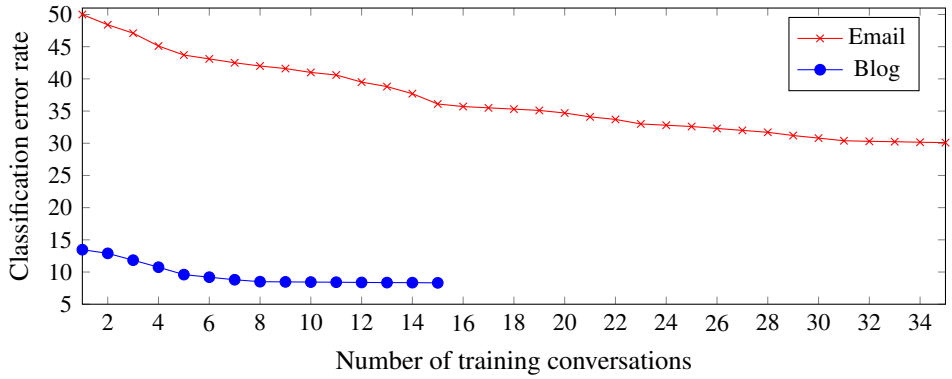



**Figure 2.7:** Error rate vs. number of training conversations.

Given a weighted undirected graph $G = (V, E)$, where the nodes $V$ represent the sentences and the edge-weights $w(x, y)$ denote the probability (given by our classifier) of the two sentences $x$ and $y$ to appear in the same topic, we again formulate the segmentation task as a **k-way-mincut** graph partitioning problem with the intuition that sentences in a segment should discuss the same topic, while sentences in different segments should discuss different topics. We optimize the **normalized cut** criterion (i.e., Equation 2.6) to extract an optimal partition as was done before for consolidating various segments in LCSeg+FQG.

### 2.3.2 Topic Labeling Models

Now that we have methods to automatically identify the topical segments in an asynchronous conversation, the next step in the pipeline is to generate one or more informative descriptions or labels for each segment to facilitate interpretations of the topics. We are the first to address this problem in asynchronous conversation.

Ideally, a topic label should be meaningful, semantically similar to the underlying topic, general and discriminative (when there are multiple topics) [137]. Traditionally, the top $k$ words in a multinomial topic model like LDA are used to describe a topic. However, as pointed out by Mei et al. [137], at the word-level, topic labels may become too generic and impose cognitive difficulties on a user to interpret the meaning of the topic by associating the words together. For example, in Figure 2.2, without reading the text, from the words {*release, free, reaction,*

*Daggerfall*}, it may be very difficult for a user to understand that the topic is about Daggerfall's free release and people's reaction to it. On the other hand, if the labels are expressed at the sentence-level, they may become too specific to cover the whole theme of the topic [137]. Based on these observations, recent studies [111, 137] advocate for **phrase-level** topic labels, which are also consistent with the monolog corpora built as a part of the Topic Detection and Tracking (TDT) project[12]. Note that we also observe a preference for phrase-level labels within our own asynchronous conversational corpora in which annotators without specific instructions spontaneously generated topic labels at the phrase-level. Considering all this, we treat phrase-level as our target level of granularity for a topic label.

Our problem is no different from the problem of **keyphrase indexing** [134] where the task is to find a set of keyphrases either from a given text or from a controlled vocabulary (i.e., domain-specific terminologies) to describe the topics covered in the text. In our setting, we do not have such a controlled vocabulary. Furthermore, exploiting generic knowledge bases like Wikipedia as a source of devising such a controlled vocabulary [134] is not a viable option in our case since the topics are very specific to a particular discussion (e.g., 'Free release of Daggerfall and reaction', 'Game contents or size' in Figure 2.2). In fact, none of the human-authored labels in our development set appears verbatim in Wikipedia. We propose to generate topic labels using a **keyphrase extraction** approach that identifies the most representative phrase(s) in the given text. We adapt a graph-based **unsupervised** ranking framework, which is domain independent, and without relying on any labeled data achieves state-of-the-art performance on keyphrase extraction [139]. Figure 2.8 shows our topic labeling framework. Given a (topically) segmented conversation, our system generates $k$ keyphrases to describe each topic in the conversation. Below we discuss the different components of the system.

### Preprocessing

In the preprocessing step, we tokenize the text and apply a **syntactic filter** to select the words of a certain part-of-speech (POS). We use the state-of-the-art Illinois tagger[13] to tokenize the text and annotate the tokens with their POS tags. We ex-

[12] http://projects.ldc.upenn.edu/TDT/

[13] Available at http://cogcomp.cs.illinois.edu/page/software

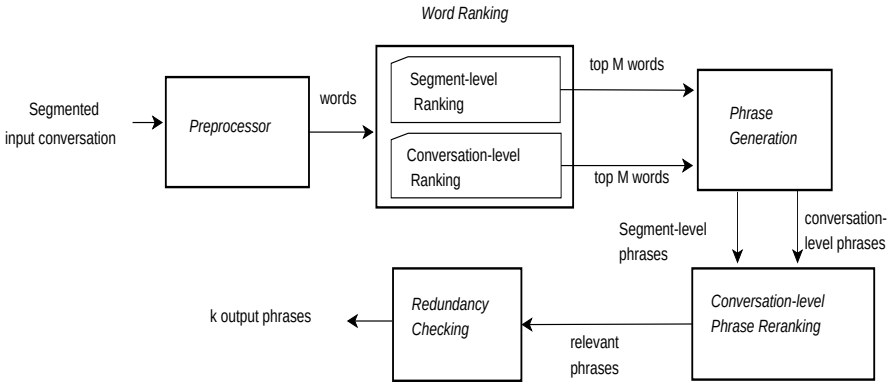



**Figure 2.8:** Topic labeling framework for asynchronous conversation.

perimented with five different syntactic filters. They select (i) nouns, (ii) nouns and adjectives, (iii) nouns, adjectives and verbs, (iv) nouns, adjectives, verbs and adverbs, and (v) all words, respectively. The filters also exclude stopwords. The second filter, that selects only nouns and adjectives, achieves the best performance on our development set, which is also consistent with the finding of [140]. Therefore, this syntactic filter is used in our system.

### Word Ranking

The words selected in the preprocessing step correspond to the nodes in our word graph. A direct application of the ranking method described in [140] would define the edges based on the **co-occurrence** relation between the respective words, and then apply the PageRank [155] algorithm to rank the nodes. We argue that co-occurrence relations may be insufficient for finding topic labels in asynchronous conversations. To better identify the labels one needs to consider aspects that are specific to asynchronous conversations. In particular, we propose to incorporate two different forms of conversation specific information into our graph-based ranking model: (1) informative clues from the **leading sentences** of a topical segment, and (2) the fine-grained conversational structure (i.e., the **Fragment Quotation Graph (FQG)**) of the conversation. In the following, we describe these two novel extensions in turn.

*Incorporating Information from the Leading Sentences*

In general, the leading sentences of a topic segment carry informative clues for the topic labels, since this is where the speakers will most likely try to signal a topic shift and introduce the new topic. Our key observation is that this is especially true for asynchronous conversations, in which topics are interleaved and less structured. For example, in Figure 2.2, notice that in almost every case, the leading sentences of the topical segments cover the information conveyed by the respective labels. This property is further confirmed in Figure 2.9, which shows the percentage of non-stopwords in the human-authored labels that appear in leading sentences of the segments in our development set. The first sentence covers about 29% and 38% of the words in the gold labels in the blog and email corpora, respectively. The first two sentences cover around 35% and 45% of the words in the gold labels in blog and email, respectively. When we consider the first three sentences, the coverage increases to 39% and 49% for blog and email, respectively. The increment is less as we add more sentences.

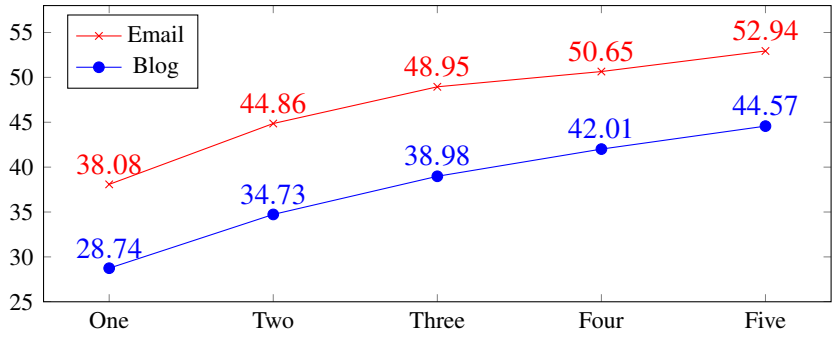



**Figure 2.9:** Percentage of words in the human-authored labels appearing in leading sentences of the topical segments.

To leverage this useful information in our ranking model, we propose the following **biased random walk** model, where $P(w|U_k)$, the score of a word $w$ given a set of leading sentences $U_k$ in topic segment $k$, is expressed as a convex combination of its relevance to the leading sentences $U_k$ (i.e., $\rho(w|U_k)$) and its relatedness with other words in the segment:

$$P(w|U_k) = \lambda \frac{\rho(w|U_k)}{\sum_{z \in C_k} \rho(z|U_k)} + (1-\lambda) \sum_{y \in C_k} \frac{e(y,w)}{\sum_{z \in C_k} e(y,z)} P(y|U_k) \tag{2.10}$$

where the value of $\lambda$ ($0 \leq \lambda \leq 1$), which we call the **bias**, is a trade-off between the two components and should be set empirically. For higher values of $\lambda$, we give more weight to the word's relevance to the leading sentences compared to its relatedness with other words in the segment. Here, $C_k$ is the set of words in segment $k$, which represents the nodes in the graph. The denominators in both components are for normalization. We define $\rho(w|U_k)$ as:

$$\rho(w|U_k) = log(tf_w^{U_k} + 1).log(tf_w^k + 1) \tag{2.11}$$

where $tf_w^{U_k}$ and $tf_w^k$ are the number of times word $w$ appears in $U_k$ and segment $k$, respectively. A similar model has proven to be successful in measuring the relevance of a sentence to a query in query-based sentence retrieval [4].

Recall that when there are multiple topics in a conversation, a requirement for the topic labels is that labels of different topics should be discriminative (or distinguishable) [137]. This implicitly indicates that a high scoring word in one segment should not have high scores in other segments of the conversation. Keeping this in mind, we define the (undirected) edge weights $e(y,w)$ in Equation 2.10 as follows:

$$e(y,w) = tf_{w,y}^k \times log \frac{K}{0.5 + tf_{w,y}^{k'}} \tag{2.12}$$

where $K$ denotes the number of topics (or topic segments) in the conversation, and $tf_{w,y}^k$ and $tf_{w,y}^{k'}$ are the number of times words $w$ and $y$ co-occur in a window of size $s$ in segment $k$ and in segments except $k$ in the conversation, respectively. Notice that this measure is similar in spirit to the TF.IDF metric [178], but it is at the co-occurrence level. The co-occurrence relationship between words captures syntactic dependencies and lexical cohesion in a text, and is also used in [140].[14]

Equation 2.10 above can be written in matrix notation as:

[14]Mihalcea and Tarau [140] use an unweighted graph for key phrase extraction. However, in our experiments, we get better results with a weighted graph.

$$\pi = [\lambda Q + (1-\lambda)R]^T \pi = A^T \pi, \tag{2.13}$$

where $Q$ and $R$ are square matrices such that $Q_{i,j} = \frac{\rho(j|U_k)}{\sum_{z \in C_k} \rho(z|U_k)}$ for all $i$, and $R_{i,j} = \frac{e(i,j)}{\sum_{j \in C_k} e(i,j)}$, respectively. Notice that $A$ is a stochastic matrix (i.e., all rows add up to 1), therefore, it can be treated as the transition matrix of a Markov chain. If we assume each word is a state in a Markov chain, then $A_{i,j}$ specifies the transition probability from state $i$ to state $j$ in the corresponding Markov chain. Another interpretation of $A$ can be given by a biased random walk on the graph. Imagine performing a random walk on the graph, where at every time step, with probability $\lambda$, a transition is made to the words that are relevant to the leading sentences and with probability $1-\lambda$, a transition is made to the related words in the segment. Every transition is weighted according to the corresponding elements of $Q$ and $R$. The vector $\pi$ we are looking for is the stationary distribution of this Markov chain and is also the (normalized) eigenvector of $A$ for the eigenvalue 1. A Markov chain will have a unique stationary distribution if it is ergodic [182]. We can ensure the Markov chain to have this property by reserving a small probability for jumping to any other state from the current state[15] [155]. For larger matrices, $\pi$ can be efficiently computed by an iterative method called the **power method**.

*Incorporating Conversational Structure*

In Section 2.3.1, we described how the fine conversation structure in the form of a Fragment Quotation Graph (FQG) can be effectively exploited in our topic segmentation models. We hypothesize that our topic labeling model can also benefit from the FQG. In previous work on email summarization, Carenini et al. [35] applied PageRank to the FQG to measure the importance of a sentence and demonstrated the benefits of using a FQG. This finding implies that an important node in the FQG is likely to cover an important aspect of the topics discussed in the conversation. Our intuition is that, to be in the topic label, a keyword should not only co-occur with other keywords, but it should also come from an important fragment

[15]For simplicity, we do not make this random jump component explicit in our equations. But, readers should keep in mind that all the transition matrices described in this chapter contain this component.

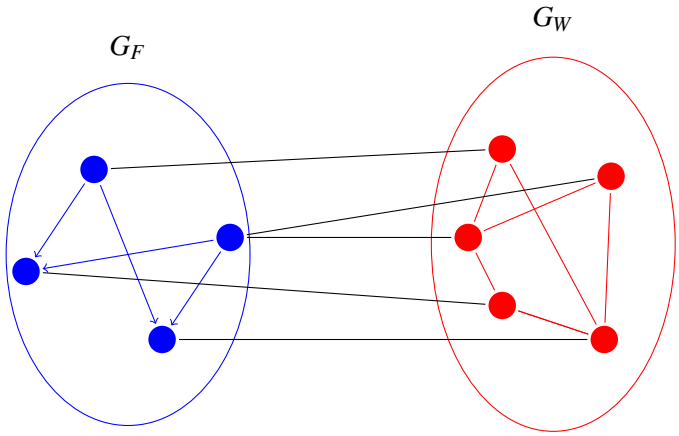



**Figure 2.10:** Three sub-graphs used for co-ranking: the fragment quotation graph $G_F$, the word co-occurrence graph $G_W$, and the bipartite graph $G_{FW}$ that ties the two together. Blue nodes represent fragments, red nodes represent words.

in the FQG. We believe there is a mutually reinforcing relationship between the FQG and the Word Co-occurrence Graph (WCG) that should be reflected in the rankings. Our proposal is to implement this idea as a process of **co-ranking** [238] in a heterogeneous graph, where three random walks are combined together.

Let $G = (V,E) = (V_F \cup V_W, E_F \cup E_W \cup E_{FW})$ be the heterogeneous graph of fragments and words. As shown in Figure 2.10, it contains three sub-graphs. First, $G_F = (V_F, E_F)$ is the unweighted directed FQG, with $V_F$ denoting the set of fragments and $E_F$ denoting the set of directed links between fragments. Second, $G_W = (V_W, E_W)$ is the weighted undirected WCG, where $V_W$ is the set of words in the segment and $E_W$ is the set of edge-weights as defined in Equation 2.12. Third, $G_{FW} = (V_{FW}, E_{FW})$ is the weighted bipartite graph that ties $G_F$ and $G_W$ together representing the occurrence relations between the words and the fragments. Here, $V_{FW} = V_F \cup V_W$, and weighted undirected edges in $E_{FW}$ connect each fragment $v_f \in V_F$ to each word $v_w \in V_W$, with the weight representing the number of times word $v_w$ occurs in fragment $v_f$.

The co-ranking framework combines three random walks, one on $G_F$, one on $G_W$ and one on $G_{FW}$. Let $F$ and $W$ denote the transition matrices for the (intra-

class) random walks in $G_F$ and $G_W$, respectively, and $\mathbf{f}$ and $\mathbf{w}$ denote their respective stationary distributions. Since, $G_{FW}$ is a bipartite graph, the (inter-class) random walk on $G_{FW}$ can be described by two transition matrices, $FW_{|V_F|\times|V_W|}$ and $WF_{|V_W|\times|V_F|}$. One intra-class step changes the probability distribution from (**f, 0**) to ($F^T$**f, 0**) or from (**0,** $\mathbf{w}$) to (**0,** $W^T\mathbf{w}$), while one inter-class step changes the distribution from (**f,** $\mathbf{w}$) to ($WF^T\mathbf{w}$, $FW^T\mathbf{f}$) (see [238] for details). The coupling is regulated by a parameter $\delta$ ($0 \leq \delta \leq 1$) that determines the extent to which the ranking of words and the ranking of fragments depend on each other. Specifically, the two update steps in the power method are:

$$f^{t+1} = (1-\delta)\,(F^T f^t) + \delta\, WF^T(FW^T WF^T)w^t \tag{2.14}$$

$$w^{t+1} = (1-\delta)\,(W^T w^t) + \delta\, FW^T(WF^T FW^T)f^t \tag{2.15}$$

We described the co-ranking framework above assuming that we have a WCG and its corresponding FQG. However, recall that while the WCG is built for a topic segment, the FQG described so far (Figure 2.4) is based on the whole conversation. In order to construct a FQG for a topic segment in the conversation, we take only those fragments (and the edges) from the conversation-level FQG that include only the sentences of that segment. This operation has two consequences. One, some conversation-level fragments may be pruned. Two, some sentences in a conversation-level fragment may be discarded. For example, the FQG for topic (segment) ID 1 in Figure 2.2 includes only the fragments $a, h, i, j$, and $l$, and the edges between them. Fragment $j$, which contains three sentences in the conversation-level FQG, contains only one sentence in the FQG for topic ID 1.

### Phrase Generation

Once we have a ranked list of words for describing a topical segment, we select the top $M$ keywords for constructing the keyphrases (labels) from these keywords. We take a similar approach to [140]. Specifically, we mark the $M$ selected keywords in the text, and collapse the sequences of adjacent keywords into keyphrases. For example, consider the first sentence, ".. 15th anniversary of the Elder Scrolls series .." in Figure 2.2. If 'Elder', 'Scrolls' and 'series' are selected as keywords, since

they appear adjacent in the text, they are collapsed into one single keyphrase 'Elder Scrolls series'. The score of a keyphrase is then determined by taking the maximum score of its constituents (i.e., keywords).

Rather than constructing the keyphrases in the post-processing phase, as we do, an alternative approach is to first extract the candidate phrases using either n-gram sequences or a chunker in the preprocessing, and then rank those candidates [91, 134]. However, determining the optimal value of $n$ in the n-gram sequence is an issue, and including all possible n-gram sequences for ranking excessively increases the problem size. Mei et al. [137] also show that using a chunker leads to poor results due to the inaccuracies in the chunker, especially when it is applied to a new domain like ours.

**Conversation-level Phrase Re-ranking**

So far, we have extracted phrases only from the topic segment ignoring the rest of the conversation. This method fails to find a label if some of its constituents appear outside the segment. For example, in our Blog corpus, the phrase *server security* in the human-authored label *server security and firewall* does not appear in its topical segment, but appears in the whole conversation. In fact, in our development set, about 14% and 8% words in the blog and email labels, respectively, come from parts of the conversation that are outside the topic segment. Thus, we propose to extract informative phrases from the whole conversation, re-rank those with respect to the individual topics (or segments) and combine only the relevant conversation-level phrases with the segment-level ones.

We rank the words of the whole conversation by applying the ranking models described in Section 2.3.2 and extract phrases using the same method described in Section 2.3.2. Note that when we apply our biased random walk model to the whole conversation, there is no concept of leading sentences and no distinction between the topics. Therefore, to apply to the whole conversation, we adjust our biased random walk model (Equation 2.10) as follows:

$$P(w) = \sum_{y \in C_k} \frac{e(y,w)}{\sum_{z \in C_k} e(y,z)} P(y) \tag{2.16}$$

where $e(y,w) = tf_{w,y}$, is the number of times words $w$ and $y$ co-occur in a window

of size $s$ in the conversation. On the other hand, the co-ranking framework, when applied to the whole conversation, combines two conversation-level graphs: the FQG of the conversation, and the WCG built for all words in the conversation.

To re-rank the phrases extracted from the whole conversation with respect to a particular topic in the conversation, we reuse the score of the words in that topic segment (given by the ranking models in Section 2.3.2). As before, the score of a (conversation-level) phrase is determined by taking the maximum (segment-level) score of its constituents (words). If a word does not occur in the topic segment, its score is assumed to be 0.

#### Redundancy Checking

Once we have the ranked list of labels (keyphrases), the last step is to produce the final $k$ labels as output. When selecting multiple labels for a topic, we expect the new labels to be diverse without redundant information to achieve broad coverage of the topic. We use the Maximum Marginal Relevance (MMR) [33] criterion to select the labels that are relevant, but not redundant. Specifically, we select the labels one by one, by maximizing the following MMR criterion each time:

$$\hat{l} = argmax_{l \in W-S}[\rho \; Score(l) - (1-\rho) \; max_{\acute{l} \in S} \; Sim(\acute{l}, l)] \tag{2.17}$$

where $W$ is the set of all labels and $S$ is the set of labels already selected as output. We define the similarity between two labels $\acute{l}$ and $l$ as: $Sim(\acute{l}, l) = n_o/n_l$, where $n_o$ is the number of overlapping (modulo stemming) words between $\acute{l}$ and $l$, and $n_l$ is the number of words in $l$. The parameter $\rho$ ($0 \leq \rho \leq 1$) quantifies the amount of redundancy allowed.

## 2.4 Corpora and Metrics

Due to the lack of publicly available corpora of asynchronous conversations annotated with topics, we developed the first corpora annotated with topic information.

### 2.4.1 Data Collection

For email, we selected our publicly available BC3 email corpus [213] which contains 40 email conversations from the World Wide Web Consortium (W3C) mailing list.[16] The BC3 corpus, previously annotated with sentence-level speech acts, subjectivity, extractive and abstractive summaries, is one of a growing number of corpora being used for email research [36]. This corpus has an average of 5 emails per conversation and a total of 1024 sentences after excluding the quoted sentences. Each conversation also provides the thread structure based on reply-to relations.

For blog, we manually selected 20 conversations of various lengths, all short enough to still be feasible for humans to annotate, from the popular technology-related news website Slashdot[17]. Slashdot was selected because it provides reply-to links between comments, allowing accurate thread reconstruction, and since the comments are moderated by the users of the site, they are expected to have a decent standard. A conversation in Slashdot begins with an article (i.e., a short synopsis paragraph possibly with a link to the original story), and is followed by a lengthy discussion section containing multiple threads of comments and single comments. This is unlike an email conversation which contains a single thread of emails. The main article is assumed to be the root in the conversation tree (based on reply-to), while the threads and the single comments form the sub-trees in the tree. In our blog corpus, we have a total of $4,411$ sentences. The total number of comments per blog conversation varies from 30 to 101 with an average of 60.3, the number of threads per conversation varies from 3 to 16 with an average of 8.35 and the number of single comments varies from 5 to 50 with an average of 20.25.

### 2.4.2 Topic Annotation

As noted by Purver [165], topic segmentation and labeling in general is a non-trivial and subjective task even for humans, particularly when the text is unedited and less organized. The conversation phenomenon called **'Schism'** makes it even more challenging for conversations. During a schism, a new conversation takes birth from an existing one, not necessarily because of a topic shift but because

[16] http://research.microsoft.com/en-us/um/people/nickcr/w3c-summary.html
[17] http://slashdot.org/

some participants refocus their attention onto each other, and away from whoever held the floor in the parent conversation [177]. In the example email conversation shown in Figure 2.1, a schism takes place when the participants discuss the topic 'responding to I18N'. Not all our annotators agree on the fact that the topic 'responding to I18N' swerves from the topic 'TAG document'.

To properly design an effective annotation manual and procedure, we performed a two-phase pilot study before carrying out the actual annotation. Our initial annotation manual was inspired by the AMI annotation manual used for topic segmentation of ICSI meeting transcripts.[18] For the pilot study, we selected two blog conversations from Slashdot and five email conversations from the W3C corpus. Note that these conversations were not picked from our corpora. Later in our experiments we use these conversations as our **development set** for tuning different parameters of the computational models. In the first phase of the pilot study five computer science graduate students volunteered to do the annotation, generating five different annotations for each conversation. We then revised our annotation manual based on their feedback and a detailed analysis of possible sources of disagreement. In the second phase, we tested our procedure with a university postdoc doing the annotation. See Appendix A.2 for details on our annotation manual.

We prepared two different annotation manuals – one for email and one for blog. We chose to do so for two reasons. (i) As discussed earlier, our email and blog conversations are structurally different and have their own specific characteristics. (ii) The email corpus already had some annotations (e.g., abstract summaries) that we could reuse for topic annotation, whereas our blog corpus is brand new without any existing annotation.

For the actual annotation we recruited and paid three cognitive science fourth year under-graduates, who are native speakers of English and also Slashdot bloggers. On average, they took about 7 and 28.5 hours to annotate the 40 email and 20 blog conversations, respectively. In all, we have three different annotations for each conversation in our corpora. For blog conversations, the task of finding topics was carried out in four steps:

1. The annotators read the whole conversation (i.e., article, threads of com-

[18] http://mmm.idiap.ch/private/ami/annotation/TopicSegmentationGuidelinesNonScenario.pdf

ments and single comments) and wrote a short summary ($\leq$ 3 sentences) only for the threads.

2. They provided short high-level descriptions for the topics discussed in the conversation (e.g., 'Game contents or size', 'Bugs or faults'). These descriptions serve as **reference topic labels** in our work. The target number of topics and their labels were not given in advance and the annotators were instructed to find as many or as few topics as needed to convey the overall content of the conversation.

3. They assigned the most appropriate topic to each sentence. However, if a sentence covered more than one topic, they labeled it with all the relevant topics according to their order of relevance. They used the predefined topic 'OFF-TOPIC' if the sentence did not fit into any topic. Wherever appropriate they also used two other predefined topics: 'INTRO' (e.g., 'hi X') and 'END' (e.g., 'Best, X').

4. The annotators authored a single high-level 250 word summary of the whole conversation. This step was intended to help them remember anything they may have forgotten and to revise the annotations in the previous three steps.

For each email conversation in BC3, we already had three human-authored summaries. So, along with the actual conversations, we provided the annotators with those summaries to give them a brief overview of the discussion. After reading a conversation and the associated summaries, they performed tasks 2 and 3 as in the same procedure they followed for annotating blogs. The annotators carried out the tasks on paper. We created the hierarchical thread view of the conversation based on the reply-to relations between the comments (or emails) using indentations and printed each participant's information in a different color as in Gmail.

In the email corpus, the three annotators found 100, 77 and 92 topics respectively (269 in total), and in the blog corpus, they found 251, 119 and 192 topics respectively (562 in total). Table 2.3 shows some basic statistics computed on the three annotations of the conversations.[19] On average, we have 26.3 sentences and

[19]We got 100% agreement on the two predefined topics 'INTRO' and 'END'. Therefore, in all our computations we excluded the sentences marked as either 'INTRO' or 'END'.

2.5 topics per email conversation, and 220.55 sentences and 10.77 topics per blog conversation. On average, a topic in email conversations contains 12.6 sentences, and a topic in blog conversations contains 27.16 sentences. The average number of topics active at a time are 1.4 and 5.81 for email and blog conversations, respectively. The average **entropy** which corresponds to the granularity of an annotation (as described in the next section) is 0.94 for email conversations and 2.62 for blog conversations. These statistics (i.e., the number of topics and the topic density) indicate that there is a substantial amount of segmentation (and labeling) to do.

| | Mean | | Max | | Min | |
|---|---|---|---|---|---|---|
| | **Email** | **Blog** | **Email** | **Blog** | **Email** | **Blog** |
| Number of sentences | 26.3 | 220.55 | 55 | 430 | 13 | 105 |
| Number of topics | 2.5 | 10.77 | 7 | 23 | 1 | 5 |
| Average topic length | 12.6 | 27.16 | 35 | 61.17 | 3 | 11.67 |
| Average topic density | 1.4 | 5.81 | 3.1 | 10.12 | 1 | 2.75 |
| Entropy | 0.94 | 2.62 | 2.7 | 3.42 | 0 | 1.58 |

**Table 2.3:** Statistics on three human annotations per conversation.

### 2.4.3 Evaluation (and Agreement) Metrics

In this section we describe the metrics used to compare different annotations. These metrics measure both how much our annotators agree with each other, and how well our models and various baselines perform. For a given conversation, different annotations can have different numbers of topics, different topic assignments of the sentences (i.e., the clustering) and different topic labels. Below we describe the metrics used to measure the segmentation performance followed by the metrics used to measure the labeling performance.

#### Metrics for Topic Segmentation

As different annotations can group sentences into different clusters, agreement metrics widely used in supervised classification, such as the $\kappa$ statistic and $F_1$ score, are not applicable. Again, our problem of topic segmentation in asynchronous conversation is not sequential in nature. Therefore, the standard metrics widely used in

sequential topic segmentation in monolog and synchronous dialog, such as $P_k$ [21] and $WindowDiff(WD)$ [162], are also not applicable. Rather, the **one-to-one** and **local agreement** metrics described in [63] are more appropriate for our segmentation task.

The one-to-one metric measures global agreement between two annotations by pairing up topical segments from the two annotations in a way (i.e., by computing the optimal max-weight bipartite matching that maximizes the total overlap), and then reports the percentage of overlap. The local agreement metric $loc_k$ measures agreement within a context of $k$ sentences. To compute the $loc_3$ score for the m-th sentence in the two annotations, we consider the previous 3 sentences: m-1, m-2 and m-3, and mark them as either 'same' or 'different' depending on their topic assignment. The $loc_3$ score between two annotations is the mean agreement on these 'same' or 'different' judgments, averaged over all sentences. See Appendix A.1 for a detailed description of these metrics with concrete examples.

We report the annotators' agreement found in one-to-one and $loc_3$ metrics in Table 2.4. For each human annotation, we measure its agreement with the two other human annotations separately, and report the mean agreements. For email, we get high agreement in both metrics, though the local agreement (average of 83%) is a little higher than the global one (average of 80%). For blog, the annotators have high agreement in $loc_3$ (average of 80%), but they disagree more in one-to-one (average of 54%). A low one-to-one agreement in blog is quite acceptable since blog conversations are much longer and less focused than email conversations (see Table 2.3). By analyzing the two corpora we also noticed that in blogs, people are more informal and often make implicit jokes (see Figure 2.2), which makes the discourse even more unstructured. As a result, the segmentation task in blogs is more challenging for humans as well as for our models. Note that in a similar annotation task for chat disentanglement, Elsner and Charniak [63] report an average one-to-one score of 53%. Since the one-to-one score for naive baselines (see Section 2.5.1) is much lower than the human agreement, this metric differentiates human-like performance from naive baselines. Therefore, computing one-to-one correlation with the human annotations is a legitimate evaluation for our models.

When we analyze the source of disagreement in the annotation, we find that by far the most frequent reason is the same as the one observed by Elsner and Charniak

| | Mean | | Max | | Min | |
|---|---|---|---|---|---|---|
| | **Email** | **Blog** | **Email** | **Blog** | **Email** | **Blog** |
| one-to-one | 80.4 | 54.2 | 100.0 | 84.1 | 31.3 | 25.3 |
| $loc_3$ | 83.2 | 80.1 | 100.0 | 94.0 | 43.7 | 63.3 |

**Table 2.4:** Annotator agreement in one-to-one and $loc_3$ on the two corpora.

[63] for the chat disentanglement task; namely, some annotators are more specific (i.e., fine) than others (i.e., coarse). To determine the level of specificity in an annotation, similarly to [63], we use the information-theoretic concept of **entropy**. If we consider the topic of a randomly picked sentence in a conversation as a random variable $X$, its entropy $H(X)$ measures the level of detail in an annotation. For topics $k$ each having length $n_k$ in a conversation of length $N$, we compute $H(X)$ as:

$$H(X) = -\sum_{k=1}^{K} \frac{n_k}{N} log_2 \frac{n_k}{N} \tag{2.18}$$

where $K$ is the total number of topics in the conversation. The entropy gets higher as the number of topics increases and the topics are evenly distributed in a conversation. In our corpora, it varies from 0 to 2.7 in email conversations and from 1.58 to 3.42 in blog conversations (Table 2.3). These variations demonstrate the differences in specificity for different annotators, but do not determine their agreement on the general structure. To quantify this, we use the **many-to-one** metric proposed by Elsner and Charniak [63]. It maps each of the source clusters to the single target cluster with which it gets the highest overlap, then computes the total percentage of overlap. This metric is asymmetrical, and not to be used for performance evaluation.[20] However, it provides some insights about the annotation specificity. For example, if one splits a cluster of another annotator into multiple sub-clusters then, the many-to-one score from fine to coarse annotation is 100%. In our corpora, by mapping from fine (high-entropy) to coarse (low-entropy) annotation we get high many-to-one scores, with an average of 95% in email conversations and an average of 72% in blog conversations (Table 2.5). This suggests that the finer annotations have mostly the same scopic boundaries as the coarser ones.

[20] One can easily optimize it by assigning a different topic to each of the source sentences.

| | Mean | | Max | | Min | |
|---|---|---|---|---|---|---|
| | **Email** | **Blog** | **Email** | **Blog** | **Email** | **Blog** |
| many-to-one | 94.9 | 72.3 | 100 | 98.2 | 61.1 | 51.4 |

**Table 2.5:** Annotator agreement in many-to-one on the two corpora.

### Metrics for Topic Labeling

Recall that we extract keyphrases from the conversation as topic labels. Traditionally keyphrase extraction is evaluated using precision, recall and F-measure based on exact matches between the extracted keyphrases and the human-assigned keyphrases [135, 140]. However, it has been noted that this approach based on exact matches underestimates the performance [212]. For example, when compared with the reference keyphrase 'Game contents or size', a credible candidate keyphrase 'Game contents' gets evaluated as wrong in this metric. Therefore, recent studies [101, 234] suggest to use the *n*-gram-based metrics that account for near-misses, similar to the ones used in text summarization (e.g., ROUGE [114]) and machine translation (e.g., BLEU [158]).

Kim et al. [101] evaluated the utility of different *n*-gram-based evaluation metrics for keyphrase extraction and showed that the metric which we call **mutual-overlap (m-o)**, correlates most with human judgments.[21] Therefore, one of the metrics we use for evaluating our topic labeling models is m-o. Given a reference keyphrase $p_r$ of length (in words) $n_r$, a candidate keyphrase $p_c$ of length $n_c$, and $n_o$ being the number of overlapping (modulo stemming) words between $p_r$ and $p_c$, mutual-overlap is formally defined as:

$$mutual{-}overlap(p_r, p_c) = \frac{n_o}{max(n_r, n_c)} \tag{2.19}$$

This metric gives full credit to exact matches and morphological variants, and partial credit to two cases of overlapping phrases: (i) when the candidate keyphrase includes the reference keyphrase, and (ii) when the candidate keyphrase is a part

[21] Kim et al. [101] call this metric **R-precision (R-p)**, which is different from the actual definition of R-p for keyphrase evaluation given by [234]. Originally, R-p is the precision measured when the number of candidate keyphrases equals the number of gold keyphrases.

of the reference keyphrase. Notice that m-o as defined above evaluates a single candidate keyphrase against a reference keyphrase. In our setting, we have a single reference keyphrase (i.e., topic label) for each topical cluster, but as mentioned before, we may want our models to extract the top $k$ keyphrases. Therefore, we modify m-o to evaluate a set of $k$ candidate keyphrases $P_c$ against a reference keyphrase $p_r$ as follows, calling it **weighted-mutual-overlap (w-m-o)**:

$$weighted-mutual-overlap(p_r, P_c) = \sum_{i=1}^{k} \frac{n_o}{max(n_r, n_c^i)} S(p_c^i) \tag{2.20}$$

where $S(p_c^i)$ is the normalized score (i.e., it satisfies $0 \leq S(p_c^i) \leq 1$ and $\sum_{i=1}^{k} S(p_c^i) = 1$) of the i-th candidate phrase $p_c^i \in P_c$. For $k = 1$, this metric is equivalent to mutual-overlap, and for higher values of $k$, it takes the sum of $k$ mutual-overlap scores, each weighted by its normalized score.

The w-m-o metric described above only considers word overlap and ignores other semantic relations (e.g., synonymy, hypernymy) between words. However, annotators when writing the topic descriptions, may use words that are not directly from the conversation, but are semantically related. For example, given a reference keyphrase 'meeting agenda', its lexical semantic variants like 'meeting schedule' or 'meeting plan' should be treated as correct. Therefore, we also consider a generalization of w-m-o that incorporates lexical semantics. We define **weighted-semantic-mutual-overlap (w-s-m-o)** as follows:

$$weighted-semantic-mutual-overlap(p_r, P_c) = \sum_{i=1}^{k} \frac{\sum_{t_r \in p_r} \sum_{t_c \in p_c^i} \sigma(t_r, t_c)}{max(n_r, n_c^i)} S(p_c^i) \tag{2.21}$$

where $\sigma(t_r, t_c)$ is the semantic similarity between the nouns $t_r$ and $t_c$. The value of $\sigma(t_r, t_c)$ is between 0 and 1, where 1 denotes notably high similarity and 0 denotes little-to-none. Notice that, since this metric considers semantic similarity between all possible pairs of nouns, the value of this measure can be greater than 100% (when presented in percentage). We use the metrics (e.g., lin_similarity, wup_similarity) provided in the WordNet::Similarity package [161] for computing

WordNet-based similarity, and always choose the most frequent sense for a noun. The results we get are similar across the similarity metrics. For brevity, we just discuss lin_similarity here.

### Metrics for End-to-End Evaluation

Just like the human annotators, our end-to-end system takes an asynchronous conversation as input, finds the topical segments in the conversation, and then assigns short descriptions (topic labels) to each of the topical segments. It would be fairly easy to compute agreement on topic labels based on mutual overlap, if the number of topics and topical segments were fixed across the annotations of a given conversation. However, since different annotators (system or human) can identify a different number of topics and different clusterings of sentences, measuring annotator (model or human) agreement on the topic labels is not a trivial task. To solve this, we first map the clusters of one annotation (say $A_1$) to the clusters of another (say $A_2$) by the optimal one-to-one mapping described in the previous section. After that, we compute the w-m-o and w-s-m-o scores on the labels of the mapped (or paired) clusters. Formally, if $l_i^1$ is the label of cluster $c_i^1$ in $A_1$ that is mapped to the cluster $c_j^2$ with label $l_j^2$ in $A_2$, we compute w-m-o$(l_i^1, l_j^2)$ and w-s-m-o$(l_i^1, l_j^2)$.

Table 2.6 reports the human agreement for w-m-o and w-s-m-o on the two corpora. Similar to segmentation, we get higher agreement on labeling for both metrics on email. Plausibly, the reasons remain the same; the length and the characteristics (e.g., informal, less focused) of blog conversations make the annotators disagree more. However, note that these measures are computed based on one-to-one mappings of the clusters and may not reflect the agreement one would get if the annotators were asked to label the same segments.

| | Mean | | Max | | Min | |
|---|---|---|---|---|---|---|
| | **Email** | **Blog** | **Email** | **Blog** | **Email** | **Blog** |
| w-m-o | 36.8 | 19.9 | 100.0 | 54.2 | 0.0 | 0.0 |
| w-s-m-o | 42.5 | 28.2 | 107.3 | 60.8 | 0.0 | 5.2 |

**Table 2.6:** Annotator agreement in w-m-o and w-s-m-o on the two corpora.

## 2.5 Experiments

In this section we present our experimental results. First, we show the performance of the topic segmentation models. Then we show the performance of the topic labeling models based on manual segmentation. Finally, we present the performance of the end-to-end system, i.e., when the topic labeler uses automatic segmentation.

### 2.5.1 Topic Segmentation Evaluation

This section presents the experiments on the topic segmentation task.

#### Experimental Setup for Topic Segmentation

We ran six different topic segmentation models on our corpora presented in Section 2.4. Our first model is the graph-based unsupervised segmentation model presented by Malioutov and Barzilay [121]. Since the sequentiality constraint of topic segmentation in monolog and synchronous dialog does not hold in asynchronous conversation, we implement this model without this constraint. Specifically, this model (call it **M&B**) constructs a weighted undirected graph $G(V,E)$, where the nodes $V$ represent the sentences and the edge weights $w(x,y)$ represent the cosine similarity (Equation 2.5) between sentences $x$ and $y$. It then finds the topical segments by optimizing the normalized cut criterion (Equation 2.6). Thus, M&B considers the conversation globally, but models only lexical similarity.

The other five models are LDA, LDA+FQG, LCSeg, LCSeg+FQG and the Supervised model (SUP) as described in Section 2.3. The tunable parameters of the different models were set based on their performance on our developement set. The hyperparameters $\alpha$ and $\beta$ in LDA were set to their default values ($\alpha$=$50/K$, $\beta$=0.01) as suggested in [199].[22] The regularization strength $\lambda$ in LDA+FQG was set to 20. The parameters of LCSeg were set to their default values since this setting delivers the best performance on the development set. For a fair comparison, we set the same number of topics per conversation in all of the models. If at least two of the three annotators agree on the topic number, we set that number, otherwise we set the floor value of the average topic number. The mean statistics of the six model annotations are shown in Table 2.7. Comparing with the statistics of the

[22]The performance of LDA does not seem to be sensitive to the values of $\alpha$ and $\beta$.

human annotations in Table 2.3, we can notice that these numbers are within the bounds of the human annotations.[23]

| | | M&B | LDA | LDA+FQG | LCSeg | LCSeg+FQG | SUP |
|---|---|---|---|---|---|---|---|
| **Email** | Topic number | 2.41 | 2.10 | 1.90 | 2.41 | 2.41 | 2.41 |
| | Topic length | 12.41 | 13.3 | 15.50 | 12.41 | 12.41 | 12.41 |
| | Topic density | 1.90 | 1.83 | 1.60 | 1.01 | 1.39 | 1.42 |
| | Entropy | 0.99 | 0.98 | 0.75 | 0.81 | 0.93 | 0.98 |
| **Blog** | Topic number | 10.6 | 10.65 | 10.65 | 10.65 | 10.65 | 10.65 |
| | Topic length | 20.42 | 20.32 | 20.32 | 20.32 | 20.32 | 20.32 |
| | Topic density | 7.38 | 9.39 | 8.32 | 1.00 | 5.21 | 5.30 |
| | Entropy | 2.54 | 3.33 | 2.37 | 2.85 | 2.81 | 2.85 |

**Table 2.7:** Mean statistics of different model's annotation

We also evaluate the following baselines, which any useful model should outperform.

- **All different** Each sentence in the conversation constitutes a separate topic.
- **All same** The whole conversation constitutes a single topic.
- **Speaker** The sentences from each participant constitute a separate topic.
- **Blocks of $k$ (= 5, 10, 15, 20, 25, 30):** Each consecutive group of $k$ sentences in the temporal order of the conversation constitutes a separate topic.

### Results for Topic Segmentation

Table 2.8 presents the human agreement and the agreement of the models with the human annotators on our corpora. For each model annotation, we measure its agreement with the three human annotations separately using the metrics described in Section 2.4.3, and report the mean agreements. In the table, we also show the performance of the two best baselines– *the Speaker* and *Blocks of k*.

Most of the baselines perform rather poorly. *All different* is the worst baseline of all with mean one-to-one scores of only 0.05 and 0.10, and mean $loc_3$ scores

[23] Although the topic numbers per conversation are fixed for different models, LDA and LDA+FQG may find fewer topics (see Equation 2.3 and 2.4).

| | | Baselines | | Models | | | | | | Human |
|---|---|---|---|---|---|---|---|---|---|---|
| | | Speaker | Blocks of $k$ | M&B | LDA | LDA+ FQG | LCSeg | LCSeg +FQG | SUP | |
| **Email** | Mean 1-to-1 | 51.8 | 38.3 | 62.8 | 57.3 | 61.5 | 62.2 | **69.3** | **72.3** | 80.4 |
| | Max 1-to-1 | 94.3 | 77.1 | 100.0 | 100.0 | 100.0 | 100.0 | 100.0 | 100.0 | 100.0 |
| | Min 1-to-1 | 23.4 | 14.6 | 36.3 | 24.3 | 24.0 | 33.1 | 38.0 | 42.4 | 31.3 |
| | Mean $loc_3$ | 64.1 | 57.4 | 62.4 | 54.1 | 60.6 | 72.0 | **72.7** | **75.8** | 83.2 |
| | Max $loc_3$ | 97.0 | 73.1 | 100.0 | 100.0 | 100.0 | 100.0 | 100.0 | 100.0 | 100.0 |
| | Min $loc_3$ | 27.4 | 42.6 | 36.3 | 38.1 | 38.4 | 40.7 | 40.6 | 40.4 | 43.7 |
| **Blog** | Mean 1-to-1 | 33.5 | 32.0 | 30.0 | 25.2 | 28.0 | 36.6 | **46.7** | **48.5** | 54.2 |
| | Max 1-to-1 | 61.1 | 46.0 | 45.3 | 42.1 | 56.3 | 53.6 | 67.4 | 66.1 | 84.1 |
| | Min 1-to-1 | 13.0 | 15.6 | 18.2 | 15.3 | 16.1 | 23.7 | 26.6 | 28.4 | 25.3 |
| | Mean $loc_3$ | 67.0 | 52.8 | 54.1 | 53.0 | 55.4 | 56.5 | **75.1** | **77.2** | 80.1 |
| | Max $loc_3$ | 87.1 | 68.4 | 64.3 | 65.6 | 67.1 | 76.0 | 89.0 | 96.4 | 94.0 |
| | Min $loc_3$ | 53.4 | 42.3 | 45.1 | 38.6 | 46.3 | 43.1 | 56.7 | 63.2 | 63.3 |

**Table 2.8:** Segmentation performance of the two best Baselines, Human and Models. In the Blocks of $k$ column, $k = 5$ for email and $k = 20$ for blog.

of only 0.47 and 0.25 in the blog and email corpus, respectively. *Blocks of 5* is one of the best baselines in email, but it performs poorly in blog with mean one-to-one of 0.19 and mean $loc_3$ of 0.54. On the contrary, *Blocks of 20* is one of the best baselines in blog, but performs poorly in email. This is intuitive since the average number of topics and topic length in blog conversations (10.77 and 27.16) are much higher than those of email (2.5 and 12.6). *All same* is optimal for conversations containing only one topic, but its performance rapidly degrades as the number of topics increases. It has mean one-to-one scores of 0.29 and 0.28 and mean $loc_3$ scores of 0.53 and 0.54 in the blog and email corpora, respectively. *Speaker* is the strongest baseline in both domains.[24] In several cases it beats some of the under-performing models.

In the email corpus, in the one-to-one metric, generally the models agree with the annotators more than the baselines do, but less than the annotators agree with each other. We observe a similar trend in the local metric $loc_3$, however on this metric, some models fail to beat the best baselines. Notice that human agreement for some of the annotations is quite low (see the Min scores), even lower than the mean agreement of the baselines. As explained before, this is due to the fact that some human annotations are much more fine-grained than others.

[24]There are many anonymous authors in our blog corpus. We treat them as separate authors.

In the blog corpus, the agreements on the global metric (one-to-one) are much lower than that on the email corpus. The reasons were already explained in Section 2.4.3. We notice a similar trend in both metrics– some under-performing models fail to beat the baselines, while others perform better than the baselines, but worse than the human annotators.

The comparison among the models reveals a general pattern. The probabilistic generative models LDA and LDA+FQG perform disappointingly on both corpora. A reason could be the limited amount of data available for training. In our corpora, the average number of sentences per blog conversation is 220.55 and per email conversation is 26.3, which might not be sufficient for the LDA models [145]. An error analysis further reveals that nearby sentences in the segmentations performed by LDA models gets excessively distributed over topics. A likely explanation is that the independence assumption made by these models when computing the distribution over topics for a sentence from the distributions of its words causes the local context to be excessively distributed over topics. If we compare the performance of LDA+FQG with the performance of LDA, we get a significant improvement with LDA+FQG in both metrics on both corpora ($p<0.01$).[25] The regularization with the FQG prevents the local context from being excessively distributed over topics.

The unsupervised graph-based model M&B performs better than the LDA models in most cases (i.e., except $loc_3$ in blog) ($p < 0.001$). However, its performance is still far below the performance of the top performing models like LCSeg+FQG and the supervised model. The reason is that even though, by constructing a complete graph, this method considers the conversation globally, it only models lexical similarity and disregards other important features of asynchronous conversation like the fine conversation structure and the speaker.

Comparison of LCSeg with LDAs and M&B reveals that LCSeg in general is a better model. LCSeg outperforms LDA by a wide margin in one-to-one on two datasets and in $loc_3$ on email ($p < 0.001$). The difference between LCSeg and LDA in $loc_3$ on blog is also significant with $p < 0.01$. LCSeg also outperforms M&B in most cases ($p < 0.01$) except in one-to-one on email. Since LCSeg is a sequential model it extracts the topics keeping the context intact. This helps it to achieve high

[25] All tests of statistical significance were performed using paired t-tests.

$loc_3$ agreement for shorter conversations like email conversations. But, for longer conversations like blog conversations, it overdoes this (i.e., extracts larger chunks of sentences as a topic segment) and gets low $loc_3$ agreement. This is unsurprising if we look at its topic density in Table 2.7 on the two datasets— the density is very low in the blog corpus compared to the annotators and other well performing models. Another reason of its superior performance over LDAs and M&B could be its term weighting scheme. Unlike LDAs and M&B, which consider only repetition, LCSeg also considers how tightly the repetition happens. However, there is still a large gap in performance between LCSeg and other top performing models (LCSeg+FQG, SUP). As explained earlier, topics in an asynchronous conversation may not change sequentially in the temporal order of the sentences. If topics are interleaved then LCSeg fails to identify them correctly. Furthermore, LCSeg does not consider other important features beyond lexical cohesion.

When we incorporate FQG into LCSeg, we get a significant improvement in one-to-one on both corpora and in $loc_3$ on blog ($p<0.0001$). Even though the improvement in $loc_3$ on email is not significant, the agreement is quite high compared to other unsupervised models. Overall, LCSeg+FQG is the best unsupervised model. This supports our claim that sentences connected by reply-to relations in the FQG usually refer to the same topic.

Finally, when we combine all the features into our graph-based supervised segmentation model (SUP in Table 2.8), we get a significant improvement over LCSeg+FQG in both metrics across both domains ($p<0.01$). Beside the features, this improvement might also be due to the fact that, by constructing a complete graph, this model considers relations between all possible sentence pairs in a conversation, which we believe is a key requirement for topic segmentation in asynchronous conversations. The agreements achieved by the supervised model are also much closer to those of the human annotators.

An error analysis of the segmentations performed by LCSeg+FQG and the supervised model shows that although these models are accurate in clustering sentences that are long, they fail to do so for some short sentences, especially when the sentence does not contain any informative word (e.g., *excuse me?*). These errors are more frequent in the blog corpus because participants in blogs often make implicit jokes and use more informal language. Some of these errors are also due

to inaccuracies in the sentence segmenter which mistakenly finds short sentences.

We understand that while most of the lexical and topic features are inactive (in the supervised model) for these short and uninformative sentences, conversational features should at least give some hints through the edge weights in the graph for these instances. However, note that the most informative conversational feature — the FQG — could be too noisy in some cases to be informative. Therefore, we believe reducing the noise in the FQG could help us tackle some of the errors made by LCSeg+FQG and the supervised model.

### 2.5.2 Topic Labeling Evaluation

In this section we present the experimental evaluation of the topic labeling models when the models are provided with manual (or gold) segmentation. This allows us to judge their performance independently of the topic segmentation task.

#### Experimental Setup for Topic Labeling

As mentioned in Section 2.4, in the email corpus, the three annotators found 100, 77 and 92 topics (or topical segments) respectively (269 in total), and in the blog corpus, they found 251, 119 and 192 topics respectively (562 in total). The annotators wrote a short high-level description for each topic. These descriptions serve as reference topic labels in our evaluation.[26] The goal of the topic labeling models is to automatically generate such informative descriptions for each topical segment. We compare our approach with two baselines. The first baseline **FreqBL** ranks the words in a topical segment according to their frequencies. The second baseline **LeadBL**, expressed by Equation 2.11, ranks the words based on their relevance only to the leading sentences in a topical segment.

We also compare our model with two state-of-the-art keyphrase extraction methods. The first one is the unsupervised general TextRank model proposed by Mihalcea and Tarau [140] (call it **M&T**) that does not incorporate any conversation specific information. The second one is the supervised model **Maui** proposed in [135]. Briefly, Maui first extracts all n-grams up to a maximum length of 3 as

[26]Notice that in our setting, for each topic segment we have only one reference label to compare with. Therefore, we do not show the human agreement on the labeling task in tables 2.9 and 2.10.

candidate keyphrases. Then a bagged decision tree classifier filters the candidates using nine different features. Due to the lack of labeled training data in asynchronous conversations, we train Maui on the human-annotated dataset released as part of the SemEval-2010 task 5 on automatic keyphrase extraction from scientific articles [103]. This dataset contains 244 scientific papers from the ACM digital library, each comes with a set of author-assigned and reader-assigned keyphrases. The total number of keyphrases assigned to the 244 articles by both the authors and the readers is 3705.

We experimented with two different versions of our biased random walk model that incorporates informative clues from the leading sentences. One, **BiasRW**, does not include any conversation-level phrase (Section 2.3.2), and the other one, **BiasRW+** does. The parameter $U_k$, the set of leading sentences, was empirically set to the first two sentences and the bias parameter $\lambda$ was set to 0.85 based on our development set.

We experimented with four different versions of the co-ranking framework depending on what type of random walk is performed on the word co-occurrence graph (WCG) and whether the model includes any conversation-level phrases. Let **CorGen** denote the co-ranking model with a general random walk on WCG, and **CorBias** denote the co-ranking model with a biased random walk on WCG. These two models do not include any conversation-level phrases while **CorGen+** and **CorBias+** do. The coupling strength $\delta$ and the co-occurrence window size $s$ were empirically set to 0.4 and 2, respectively, based on the development set. The dumping factor was set to its default value 0.85.

Note that all the models (except Maui) and the baselines follow the same preprocessing and post-processing (i.e., phrase generation and redundancy checking) steps. The value of $M$ in phrase generation was set to 25% of the total number of words in the cluster, and $\rho$ in redundancy checking was set to 0.35 based on the development set.

### Results for Topic Labeling

We evaluate the performance of different models using the metrics described in Section 2.4.3. Tables 2.9 and 2.10, respectively, show the mean weighted-mutual-

overlap (w-m-o) and weighted-semantic-mutual-overlap (w-s-m-o) scores of different models (in percentage) for different values of $k$ (i.e., number of output labels) on the two corpora.

Both the baselines have proved to be strong, beating the existing models in almost every case. This tells us that the frequency of the words in the topic segment and their occurrence in the leading sentences carry important information for topic labeling. Generally speaking, LeadBL is a better baseline for email, while for blog FreqBL is better than LeadBL.

The supervised model Maui is the worst performer in both metrics on the two corpora. Its performance is also consistently low across the corpora for any particular value of $k$. A possible explanation is that Maui was trained on a domain (scientific articles), which is rather different from asynchronous conversations. Another reason may be that Maui does not consider any conversational features.

The general random walk model M&T also delivers poor performance on our corpora, failing to beat the baselines in both measures. This indicates that the random walk model based on only co-occurrence relations between the words is not sufficient for finding topic labels in asynchronous conversations. It needs to consider conversation specific information.

By incorporating clues from the leading sentences, our biased random walk model BiasRW gives improved performance over the baselines in both metrics for all the values of $k$ on the two corpora ($p<0.05$). This demonstrates the usefulness of considering the leading sentences as an information source for topic labeling in asynchronous conversation.

The general co-ranking model CorGen, by incorporating the conversation structure, outperforms the baselines in both metrics for all $k$ on blog ($p<0.05$), but fails to do so in many cases on email. On blog, there is also no significant difference between BiasRW and CorGen in w-m-o for all $k$ (Table 2.9), but CorGen outperforms BiasRW in w-s-m-o (Table 2.10) for higher values of $k$ (2,3,4,5) ($p<0.05$). On the other hand, on email, BiasRW always outperforms CorGen in both metrics for all $k$ ($p<0.05$). So we can conclude that, on blog, exploiting the conversation structure seems to be more beneficial than the leading sentences, whereas on email, we observe the opposite. The reason could be that the topic segments in blog are much longer than those of email (average length 27.16 vs. 12.6). Therefore, the

| | | k=1 | | k=2 | | k=3 | | k=4 | | k=5 | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | | Email | Blog | Email | Blog | Email | Blog | Email | Blog | Email | Blog |
| **Baselines** | FreqBL | 22.86 | 19.05 | 17.47 | 16.17 | 14.96 | 13.83 | 13.17 | 13.45 | 12.06 | 12.59 |
| | LeadBL | 22.41 | 18.17 | 18.94 | 15.95 | 15.92 | 13.75 | 14.36 | 12.61 | 13.76 | 11.93 |
| **Models** | M&T | 15.87 | 18.23 | 12.68 | 14.31 | 10.33 | 12.15 | 9.63 | 11.38 | 9.07 | 11.03 |
| | Maui | 10.48 | 10.03 | 9.86 | 9.56 | 9.03 | 9.23 | 8.71 | 8.90 | 8.50 | 8.53 |
| | BiasRW | 24.77 | 20.83 | 19.78 | 17.28 | 17.38 | 15.06 | 16.24 | 14.53 | 15.80 | 14.26 |
| | BiasRW+ | 24.91 | 23.65 | **20.36** | 19.69 | 18.09 | 17.76 | 16.20 | 16.78 | 15.78 | 15.86 |
| | CorGen | 17.60 | 20.76 | 15.32 | 17.64 | 15.14 | 15.78 | 14.23 | 15.03 | 14.08 | 14.75 |
| | CorGen+ | 18.32 | 22.44 | 15.86 | 19.65 | 15.46 | 18.01 | 14.89 | 16.90 | 14.45 | 16.13 |
| | CorBias | 24.84 | 20.96 | 19.88 | 17.73 | 17.61 | 16.22 | 16.99 | 15.64 | 16.81 | 15.38 |
| | CorBias+ | **25.13** | **23.83** | 20.20 | **19.97** | **18.21** | **18.33** | **17.15** | **17.28** | **16.90** | **16.55** |

**Table 2.9:** Mean weighted-mutual-overlap (w-m-o) scores for different values of $k$ on two corpora.

FQGs of blog segments are generally larger and capture more information than the FQGs of email segments. Besides, email discussions are more focused than blog discussions. The leading sentences in email segments carry more informative clues than those of blog segments. This is also confirmed in Figure 2.9, where the leading sentences in email cover more of the human-authored words than they do in blog.

| | | k=1 | | k=2 | | k=3 | | k=4 | | k=5 | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | | Email | Blog | Email | Blog | Email | Blog | Email | Blog | Email | Blog |
| **Baselines** | FreqBL | 23.36 | 23.52 | 20.50 | 21.03 | 19.82 | 20.18 | 18.47 | 19.58 | 17.81 | 19.27 |
| | Lead-BL | 24.99 | 21.19 | 21.69 | 20.61 | 20.40 | 19.49 | 19.57 | 18.98 | 19.17 | 18.71 |
| **Models** | M&T | 18.71 | 22.08 | 16.25 | 19.59 | 14.62 | 17.91 | 14.29 | 17.27 | 14.06 | 16.92 |
| | Maui | 14.79 | 14.14 | 13.76 | 13.67 | 13.03 | 12.87 | 12.69 | 12.10 | 11.73 | 11.52 |
| | BiasRW | **28.87** | 24.63 | 24.76 | 22.51 | 22.48 | 21.36 | 21.67 | 20.95 | 21.28 | 20.78 |
| | BiasRW+ | 27.96 | 24.51 | 24.71 | 23.05 | 22.56 | 22.88 | 21.19 | 22.08 | 20.82 | 21.73 |
| | CorGen | 23.66 | 24.69 | 21.97 | 23.83 | 21.51 | 22.86 | 20.98 | 22.37 | 20.44 | 22.22 |
| | CorGen+ | 23.50 | 24.30 | 22.09 | **24.35** | 21.96 | **23.89** | 21.36 | 23.42 | 20.90 | 23.00 |
| | CorBias | 28.44 | **25.66** | **26.39** | 24.15 | 24.47 | 23.18 | **23.70** | 22.76 | **23.56** | 22.67 |
| | CorBias+ | 27.97 | 25.26 | 26.34 | 24.19 | **24.69** | 23.60 | 23.65 | **23.44** | 23.23 | **23.20** |

**Table 2.10:** Mean weighted-semantic-mutual-overlap scores for different values of $k$ on two corpora.

By combining the two forms of conversation specific information into a single model, CorBias delivers improved performance over CorGen and BiasRW in both metrics. On email, CorBias is significantly better than CorGen for all $k$ with both metrics ($p<0.01$). On blog, CorBias gets significant improvement over Bi-

asRW for higher values of $k$ $(3,4,5)$ with both metrics ($p<0.05$). The two sources of information are complementary and help each other to overcome the domain-specific limitations of the respective models. Therefore, one should exploit both information sources to build a generic domain-independent system.

When we include the conversation-level phrases (+ versions), we get a significant improvement in w-m-o on blog ($p<0.01$), but not on email. This may be because blog conversations have many more topical segments than email conversations (average topic number 10.77 vs. 2.5). Thus, there is little information for the label of a topical segment outside that segment in email conversations. However, note that including conversation-level phrases does not hurt the performance significantly in any case.

| | | k=2 | | k=3 | | k=4 | | k=5 | |
|---|---|---|---|---|---|---|---|---|---|
| | | Email | Blog | Email | Blog | Email | Blog | Email | Blog |
| **Baselines** | FreqBL | 27.02 | 23.69 | 29.79 | 24.29 | 31.12 | 24.88 | 31.25 | 25.58 |
| | LeadBL | 28.72 | 21.69 | 30.86 | 23.14 | 31.99 | 24.19 | 31.99 | 25.33 |
| **Models** | M&T | 21.45 | 21.70 | 23.12 | 23.18 | 25.23 | 23.82 | 25.45 | 24.07 |
| | Maui | 14.00 | 14.85 | 15.57 | 17.33 | 17.15 | 19.23 | 18.40 | 20.03 |
| | BiasRW | 29.34 | 24.92 | 31.42 | 25.18 | 32.58 | 25.89 | 32.97 | 26.64 |
| | BiasRW+ | 29.47 | 25.88 | 31.43 | 27.38 | 32.96 | 28.47 | 33.87 | 29.17 |
| | CorGen | 23.45 | 25.05 | 28.44 | 25.72 | 30.10 | 26.40 | 30.33 | 27.10 |
| | CorGen+ | 24.56 | 25.87 | 28.46 | 26.61 | 31.14 | 27.63 | 32.91 | 28.50 |
| | CorBias | 28.98 | 25.27 | 30.90 | 26.41 | 32.24 | 27.14 | 33.25 | 27.65 |
| | CorBias+ | **29.76** | **25.96** | **31.04** | **27.65** | **33.61** | **28.63** | **35.35** | **29.58** |

**Table 2.11:** Mean weighted-mutual-overlap (w-m-o) scores when the best of $k$ labels is considered.

To further analyze the performance of the models, Table 2.11 shows the mean w-m-o scores when only the best of $k$ output labels is considered. This allows us to judge the models' ability to generate the best label in the top $k$ list. The results are much clearer here. Generally speaking, among the models that do not include conversation-level phrases, CorBias is the best model, while including conversation-level phrases improves the performance further.

Table 2.12 shows some of the examples from our test set where the system-generated (i.e., CorBias+) labels are very similar to the human-authored ones. There are also many cases like the ones in Table 2.13, where the system-generated

| | Human-authored | System-generated (top 5) |
|---|---|---|
| **Email** | *Details of Bristol meeting* | *Bristol, face2face meeting, England, October* |
| | *Nashville conference* | *Nashville conference, Courseware developers, mid October, event* |
| | *Meeting agenda* | *detailed agenda, main point, meetings, revision, wcag meetings* |
| | *Design guidelines* | *general rule, design guidelines, accessible design, absolutes, forbid* |
| | *Contact with Steven* | *Steven Pemberton, contact, charter, status, schedule w3c* |
| **Blog** | *faster than light (FTL) travel* | *FTL travel, need FTL, limited FTL, FTL drives, long FTL* |
| | *Dr. Paul Laviolette* | *Dr. Paul Laviolette, bang theory, systems theory, extraterrestial beacons, laugh* |
| | *Vietnam and Iraq warfare* | *Vietnam war, incapable guerrilla war, war information, war ii, vietnamese war* |
| | *Pulsars* | *mean pulsars, pulsars slow time, long pulsars, relative pulsars, set pulsars* |
| | *Linux distributions* | *linux distro, linux support, major linux, viable linux* |

**Table 2.12:** Examples of Human-authored and System-generated labels.

labels are reasonable, although they get low w-m-o and w-s-m-o scores when compared with the human-authored labels.

| Human-authored | System-generated |
|---|---|
| *Meeting time and place* | *October, mid October, timing, w3c timing issues, Ottawa* |
| *Archaeology* | *religious site, burial site, ritual site, barrows tomb* |
| *Bio of Al* | *Al Gilman, standards reformer, repair interest group, ER IG, ER teams* |
| *Budget Constraints* | *budget, notice, costs, smaller companies, travel* |
| *Food choice* | *roast turkey breast, default choices, small number, vegetable rataouille, lunch* |

**Table 2.13:** Examples of System-generated labels that are reasonable but get low scores.

This is because most of the human-authored labels in our corpora are abstractive in nature. Annotators often write their own labels rather than simply copying keyphrases from the text. In doing so, they rely on their expertise and general world knowledge that may go beyond the contents of the conversation. In fact, although annotators reuse many words from the conversation, only 9.81% of the human-authored labels in blog and 12.74% of the human-authored labels in email appear verbatim in their respective conversations. Generating human-like labels will require a deeper understanding of the text and robust textual inference, for which our extractive approach can provide some useful input. For an example, see our recent paper [136], which describes such an abstractive approach.

### 2.5.3 Full System Evaluation

In this section we present the performance of our end-to-end system. We first segment a given asynchronous conversation using our best topic segmenter (the supervised model), and then feed its output to our best topic labeler (the CorBias+ model). Table 2.14 presents the human agreement and the agreement of our system with the human annotators based on the best of $k$ outputs. For each system annotation we measure its agreement in w-m-o and w-s-m-o with the three human annotations using the method described in Section 2.4.3.

| | | System | | | | | Human |
|---|---|---|---|---|---|---|---|
| | | k=1 | k=2 | k=3 | k=4 | k=5 | |
| **Email** | Mean w-m-o | 19.19 | 23.62 | 26.19 | 27.06 | 28.06 | 36.84 |
| | Max w-m-o | 100.0 | 100.0 | 100.0 | 100.0 | 100.0 | 100.0 |
| | Mean w-s-m-o | 24.98 | 32.08 | 34.63 | 36.92 | 38.95 | 42.54 |
| | Max w-s-m-o | 108.43 | 108.43 | 108.43 | 108.43 | 108.43 | 107.31 |
| **Blog** | Mean w-m-o | 9.71 | 11.71 | 14.55 | 15.83 | 16.72 | 19.97 |
| | Max w-m-o | 26.67 | 26.67 | 35.00 | 35.00 | 35.00 | 54.17 |
| | Mean w-s-m-o | 15.46 | 19.77 | 23.35 | 25.57 | 26.23 | 28.22 |
| | Max w-s-m-o | 47.10 | 47.28 | 47.28 | 48.54 | 48.54 | 60.76 |

**Table 2.14:** Performance of the end-to-end system and human agreement.

Notice that in email, our system gets 100% agreement in w-m-o metric for some conversations. However, there is a substantial gap between the mean and the max w-m-o scores. Similarly, in w-s-m-o, our system achieves a maximum of 108% agreement, but the mean varies from 25% to 39% depending on different values of $k$. In blog, the w-m-o and w-s-m-o scores are much lower. The maximum scores achieved in w-m-o and w-s-m-o metrics in blog are only 35% and 49% (for $k = 5$), respectively. The mean w-m-o score varies from 10% to 17%, and the mean w-s-m-o score varies from 15% to 28% for different values of $k$. This demonstrates the difficulties of topic segmentation and labeling tasks in blog conversations.

Comparing with Table 2.11, we can notice that inaccuracies in the topic segmenter affects the overall performance. However, our results are encouraging. Even though for lower values of $k$ there is a substantial gap between our results and the human agreement, as the value of $k$ increases, our results get closer to the

human agreement, especially in w-s-m-o.

## 2.6 Conclusion and Future Directions

In this work we presents two new corpora of email and blog conversations annotated with topics, which, along with the proposed metrics, will allow researchers to evaluate their work quantitatively. We also present a complete computational framework for topic segmentation and labeling in asynchronous conversation. Our approach extends state-of-the-art methods by considering a fine-grained structure of the asynchronous conversation, along with other conversational features. We do this by applying recent graph-based methods for NLP such as min-cut and random walk on paragraph, sentence or word graphs.[27]

For topic segmentation, we extend LDA and LCSeg, two state-of-the-art unsupervised models, to incorporate a fine-grained conversational structure (i.e., the FQG), generating two novel unsupervised models LDA+FQG and LCSeg+FQG for asynchronous conversation. We incorporate the FQG into LDA by replacing its standard Dirichlet prior with an informative Dirichlet Tree prior. On the other hand, we propose a graph-based clustering model to incorporate the FQG into LCSeg. In addition to that, we also propose a novel supervised segmentation model that combines lexical, conversational and topic features using a classifier in the graph-based clustering framework. For topic labeling, we propose two novel random walk models that extract the most representative keyphrases from the text, by respectively capturing conversation specific clues from two different sources: the leading sentences and the fine conversational structure i.e., the FQG.

Experimental results in the topic segmentation task demonstrate that both LDA and LCSeg benefit significantly when they are extended to consider the FQG, with LCSeg+FQG being the best unsupervised model. The comparison of the supervised segmentation model with the unsupervised models shows that the supervised method outperforms the unsupervised ones even using only a few labeled conversations, being the best segmentation model overall. The outputs of LCSeg+FQG and the supervised model are also highly correlated with human annotations in both

[27] Our conversational corpora annotated with topics, annotation manual and source code for all the systems are publicly available from www.cs.ubc.ca/labs/lci/bc3.html

local and global metrics. The experiment on the topic labeling task reveals that the random walk models perform better when they exploit the conversation specific clues and the best results are achieved when all the sources of clues are exploited. The evaluation of the complete end-to-end system also shows promising results when compared with human performance.

Error analysis in the topic segmentation task reveals that as the conversation becomes more informal, less focused and less structured, it imposes more difficulties to humans as well as to automatic systems. The application of probabilistic generative models like LDAs tends to excessively split local contexts across multiple topics. On the other hand, the similarity-based sequential segmentation model LCSeg tends to extract larger chunks of sentences as segments for long conversations. The incorporation of FQG helps both unsupervised models to tackle many of these errors. Combining all the important features in a graph-based supervised framework further reduces the errors. However, we believe even further error reduction is possible by reducing noise in the FQG.

A comparison between system-generated and human-authored topic labels tells us that extractive approaches may not be enough to generate human-like labels. Generating human-like labels will require a deeper understanding of the text and robust textual inference, for which our approach can provide some useful input.

This work can be extended in many ways. Given that most of the human-authored labels are abstractive in nature, we plan to extend our labeling framework to generate more abstract human-like labels that could better synthesize the information expressed in a topic segment. A promising approach would be to rely on more sophisticated methods for information extraction, combined with more semantics (e.g., phrase entailment) and data-to-text generation techniques. Another interesting avenue for future work is to perform a more extrinsic evaluation of our methods. Instead of testing them with respect to a human gold standard, it would be extremely interesting to see how effective they are when used to support other NLP tasks, such as summarization and visualization. We are also interested in the future to transfer our approach to other similar domains by domain adaptation methods. We plan to work on both synchronous and asynchronous domains.

# Chapter 3

# Rhetorical Parsing

Chapter 2 presented our complete computational framework for finding the high-level discourse structure i.e., the global topic structure of an asynchronous conversation. In addition to the global topic structure of a conversation, each comment (or message) locally exhibits a finer discourse structure called **rhetorical structure**, which logically binds the discourse units (i.e., clauses, sentences) together. In this chapter, we study rhetorical analysis, which seeks to capture this finer structure (a tree) of discourse. We propose a complete probabilistic discriminative framework for performing rhetorical analysis. Our framework comprises a discourse segmenter and a discourse (rhetorical) parser.[1] First, the discourse segmenter, which is based on a binary classifier, identifies the elementary discourse units in a given text. Then, the 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. We present two approaches to combine these two stages of discourse parsing effectively. A series of empirical evaluations over two different datasets demonstrates that our discourse parser significantly outperforms the state-of-the-art, often by a wide margin.[2]

[1] In this chapter, we use the terms discourse parsing and rhetorical parsing interchangeably.

[2] Portions of this work were previously published in two conference proceedings: Joty et al. [97] (**EMNLP-2012**) and Joty et al. [99] (**ACL-2013**).

## 3.1 Introduction

A discourse of any kind is not formed of isolated and unrelated textual units, but of collocated, related and structured units. In Chapter 2, we have seen that an asynchronous conversation addresses a common topic, often covering a number of subtopics. In addition, each message in a conversation locally forms a coherent monolog by binding its sentences logically — each sentence follows smoothly from the ones before and leads into the ones which come afterwards. In other words, the author ensures a consistent **coherence structure** to make the text interpretable as a whole. Without this, a text is just a sequence of non-sequiturs. For example, consider the following two texts.

- It rained heavily during the night. The game is postponed.
- It rained heavily during the night. I like maths.

In the first text, the first sentence provides an *Explanation* for the second sentence, which makes the text interpretable as a whole. However, it is hard to establish such a relation between the two sentences in the second text, and most readers will have difficulties in understanding it. The reader will either reject it, simply calling it *"incoherent"* or spend some time to construct an explanation of what liking maths has to do with raining heavily. By asking this, the reader is actually questioning the coherence of the text.

In a coherent text, discourse units (i.e., clauses, sentences) must be linked by meaningful connections, connections like *Explanation* that are called **coherence relations**. In **rhetorical analysis**, we seek to uncover this coherence structure underneath the text, which has been shown to be beneficial for many NLP applications including text summarization [54, 118, 126], sentence compression [194], text generation [163], sentiment analysis [112, 189] and question answering [215]. Furthermore, rhetorical structure can be useful for other discourse analysis tasks like co-reference resolution using Veins theory of discourse [50].

A number of formal theories of discourse have been proposed to describe the coherence structure of a text [10, 52, 122, 129, 225]. **Rhetorical Structure Theory (RST)** [122], one of the most influential of them, represents texts by labeled hierarchical structures, called Discourse Trees (DTs). For instance, let us consider

the same example we saw in Chapter 1 as shown in Figure 3.1 for the following text from the RST-DT corpus [38]:

- But he added: "Some people use the purchasers' index as a leading indicator, some use it as a coincident indicator. But the thing it's supposed to measure — manufacturing strength — it missed altogether last month."

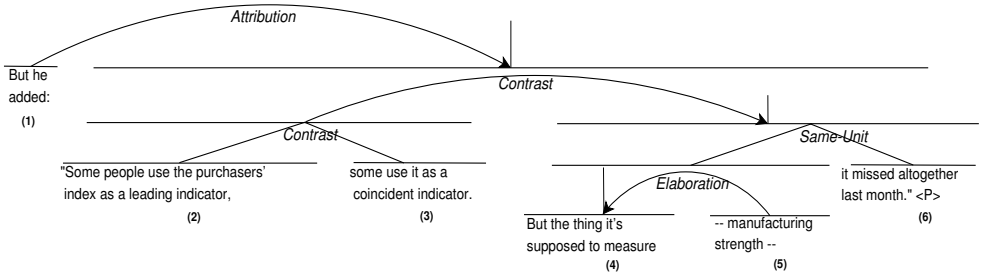



**Figure 3.1:** Discourse tree for two sentences in RST-DT. Each of the sentences contains three EDUs. The second sentence has a well-formed discourse tree, but the first sentence does not have one.

The leaves of a DT correspond to contiguous atomic text spans, called **Elementary Discourse Units (EDUs)** (six in the example). Adjacent EDUs are connected by rhetorical relations (e.g., *Elaboration*, *Contrast*), forming larger discourse units (represented by internal nodes), which in turn are also subject to this relation linking. Discourse units linked by a rhetorical relation are further distinguished based on their relative importance in the text: **nuclei** are the core parts of the relation while **satellites** are peripheral or supportive ones. For example, in Figure 3.1, *Elaboration* is a relation between a nucleus (EDU 4) and a satellite (EDU 5), and *Contrast* is a relation between two nuclei (EDUs 2 and 3). Rhetorical analysis in RST involves two subtasks: **discourse segmentation** is the task of breaking the text into a sequence of EDUs,[3] and **discourse parsing** is the task of linking the discourse units (EDUs and larger units) into a labeled hierarchical tree.

While recent advances in automatic discourse segmentation have attained considerably higher accuracies [72], discourse parsing still poses significant challenges

[3]Note that the discourse segmentation task considered in this chapter is different from the topic segmentation task studied in the previous chapter.

[69] and the performance of the existing parsers [84, 191, 201] is still considerably inferior compared to human gold-standard. Our work in this chapter aims to reduce this performance gap and take discourse parsing one step further. To this end, we address three key limitations of existing discourse parsers as follows.

First, existing discourse parsers typically model the structure and the labels of a DT separately, and also do not consider the sequential dependencies between the discourse tree constituents, which has been recently shown to be critical [69]. To address this limitation, as the first contribution, we propose a novel discourse parser based on probabilistic discriminative parsing models, expressed as Conditional Random Fields (CRFs) [205], to infer the probability of all possible discourse tree constituents. The CRF models effectively represent the structure and the label of a discourse tree constituent jointly, and whenever possible, capture the sequential dependencies between the constituents.

Second, existing parsers apply greedy and sub-optimal parsing algorithms to build the DT for a document. To cope with this limitation, our CRF models support a probabilistic bottom-up parsing algorithm which is non-greedy and optimal.

Third, existing discourse parsers do not discriminate between **intra-sentential parsing** (i.e., building the DTs for the individual sentences) and **multi-sentential parsing** (i.e., building the DT for the whole document). However, we argue that distinguishing between these two conditions can result in more effective parsing. Two separate parsing models could exploit the fact that rhetorical relations are distributed differently intra-sententially vs. multi-sententially. Also, they could independently choose their own informative feature sets. As another key contribution of our work, we devise two different parsing components: one for intra-sentential parsing, the other for multi-sentential parsing. This provides for scalable, modular and flexible solutions, that can exploit the strong correlation observed between the text structure (sentence boundaries) and the structure of the rhetorical tree.

In order to develop a complete and robust discourse parser, we combine our intra-sentential and multi-sentential parsers in two different ways. Since most sentences have a well-formed discourse sub-tree in the full DT (for example, the second sentence in Figure 3.1), our first approach constructs a DT for every sentence using our intra-sentential parser, and then runs the multi-sentential parser on the resulting sentence-level DTs. However, this approach would disregard those cases

where rhetorical structures violate sentence boundaries (also called 'leaky' boundaries [218]). For example, consider the first sentence in Figure 3.1. It does not have a well-formed discourse sub-tree because the unit containing EDUs 2 and 3 merges with the next sentence and only then is the resulting unit merged with EDU 1. Our second approach, in order to deal with the leaky cases, builds sentence-level sub-trees by applying the intra-sentential parser on a sliding window covering two adjacent sentences and by then consolidating the results produced by overlapping windows. After that, the multi-sentential parser takes all these sentence-level sub-trees and builds a full rhetorical parse for the whole document.

Our discourse parser assumes that the input text has been already segmented into elementary discourse units. As an additional contribution, we propose a novel discriminative approach to discourse segmentation that not only achieves state-of-the-art performance, but also reduces the time and space complexities by using fewer features. Notice that the combination of our segmenter with our parser forms a complete probabilistic discriminative framework for rhetorical analysis.

While previous parsers have been tested on only one corpus, we evaluate our framework on texts from two very different genres: news articles and instructional how-to-do manuals. The results demonstrate that our approach to discourse parsing provides consistent and statistically significant improvements over previous methods both at the sentence level and at the document level. The performance of our final system compares very favorably to the performance of state-of-the-art discourse parsers.

In the rest of the chapter, after discussing related work in Section 3.2, we present our rhetorical analysis framework in Section 3.3. In Section 3.4, we describe our discourse parser. Then, in Section 3.5 we present our discourse segmenter. The experiments and analysis, followed by future directions are discussed in Section 3.6. Finally, we summarize our contributions in Section 3.7.

## 3.2 Related Work

Rhetorical analysis has a long history. In this section, we provide a brief overview of the approaches that follow RST as the theory of discourse, and that are related to our work; see the survey by Stede [198] for a detailed overview.

### 3.2.1 Unsupervised Approaches

A general issue in rhetorical analysis is having sufficient data to train a model on a particular genre. It may be preferable either to develop unsupervised models that do not require any labeled data or to collect the required data automatically.

In his early work [125], Marcu presents a shallow approach relying on discourse cues (e.g., *because*, *but*) and surface patterns. He uses hand-coded rules, derived from an extensive corpus study, to break the text into EDUs and to build DTs for sentences first, then for paragraphs, and so on. Despite the fact that this work pioneered the field of rhetorical analysis, it has many limitations. First, identifying discourse cues is a difficult task on its own, because depending on the usage, the same phrase may or may not signal a coherence relation. Second, discourse segmentation using only discourse cues fails to attain high accuracy [191]. Third, rhetorical tree structures do not always correspond to paragraph structures; for example, Sporleder and Lapata [193] report that more than 20% of the paragraphs in the RST-DT corpus of news articles [38] do not correspond to a discourse unit. Fourth, discourse cues are sometimes ambiguous; for example, *but* can signal *Contrast, Antithesis* and *Concession*, and so on. Finally, a more serious problem with the rule-based approach is that often rhetorical relations are not explicitly signaled by discourse cues. For example, in RST-DT, Marcu and Echihabi [127] found that only 61 out of 238 *Contrast* relations and 79 out of 307 *Cause-Explanation* relations were explicitly signaled by cue phrases. In the British National Corpus, Sporleder and Lascarides [196] report that half of the sentences lack a discourse cue. Other studies [179, 197, 201, 206] report even higher figures: about 60% of discourse relations are not explicitly signaled. Rather than relying on hand-coded rules based on discourse cues and surface patterns, recent approaches employ machine learning techniques with a large set of informative features.

While some rhetorical relations need to be explicitly signaled by discourse cues (e.g., *Concession*) and some do not (e.g., *Background*), there is a large middle ground of relations that may be signaled or not. For these "middle ground" relations, can we exploit features present in the signaled cases to automatically identify relations when they are not signaled? The idea is to use unambiguous discourse cues (e.g., *although* for *Contrast*, *for example* for *Elaboration*) to automatically

label a large corpus with rhetorical relations that could then be used to train a supervised model.[4] A series of previous work has explored this idea. Marcu and Echihabi [127] first attempted to identify four broad classes of relations: *Contrast, Elaboration, Condition,* and *Cause-Explanation-Evidence (CEV)*. They use a naive Bayes classifier based on word-pairs $(w_1, w_2)$, where $w_1$ occurs in the left segment, and $w_2$ occurs in the right segment. Sporleder and Lascarides [195] include other features (e.g., words and their stems, POS tags, segment lengths, positions) in a boosting-based classifier (BoosTexter) to further improve classification accuracy. However, these studies evaluate classification performance on the instances where rhetorical relations are originally signaled (i.e., the cues were artificially removed). It is not clear how well this approach performs on the instances which are not originally signaled. Subsequent studies [24, 192, 196] confirm that classifiers trained on instances by stripping off the original cue phrases do not generalize well to implicit cases because they are linguistically quite different.

Note that the above approach to identifying relations in absence of manually labeled data does not perform a complete rhetorical analysis. It only attempts to identify a very small subset of coarser relations between two non-hierarchical discourse segments. Arguably, in order to perform an effective and complete rhetorical analysis, one needs to employ supervised machine learning techniques.

### 3.2.2 Supervised Approaches

In [124], Marcu applies supervised machine learning techniques to build a discourse segmenter and a shift-reduce discourse parser. Both the segmenter and the parser rely on *C*4.5 decision tree classifiers to learn the rules automatically from the data. The segmenter mainly uses shallow-syntactic (POS tags) and contextual features. To learn the shift-reduce actions, the discourse parser encodes five types of features: lexical (e.g, discourse cues), shallow-syntactic, similarity, operational (previous *n* shift-reduce operations) and rhetorical sub-structural features. Despite the fact that this work has pioneered many of today's machine learning approaches to discourse parsing, it has all the limitations mentioned in Section 3.1.

[4] We categorize this approach as unsupervised because it does not rely on human-annotated data.

Soricut and Marcu [191] present the publicly available **SPADE** system[5] that comes with probabilistic models for discourse segmentation and *sentence-level* discourse parsing. Their segmentation and parsing models are based on lexico-syntactic patterns (features) extracted from the lexicalized syntactic tree of a sentence. The discourse parser uses an optimal parsing algorithm to find the most probable rhetorical tree structure for a sentence. SPADE was trained and tested on the RST-DT corpus. This work, by showing empirically the connection between syntax and discourse at the sentence level, has greatly influenced all major contributions in this area ever since. However, it is limited in several ways. First, SPADE does not produce a full-text (document-level) parse. Second, its parsing model makes an independence assumption between the label and the structure of a DT constituent, and it ignores the sequential and the hierarchical dependencies between the constituents. Third, it relies only on lexico-syntactic features, and it follows a *generative* approach to estimate the model parameters.

Subsequent research addresses the question of how much syntax one really needs in rhetorical analysis. Sporleder and Lapata [194] focus on the *discourse chunking* problem, comprising two subtasks: discourse segmentation and non-hierarchical nuclearity assignment. More specifically, they examine whether features derived via a POS tagger and chunker would be sufficient for these purposes. They formulate discourse chunking in two alternative ways. First, *one-step classification*, where the discourse chunker, a multi-class classifier, assigns to each token one of the four labels: (i) B-NUC (beginning nucleus), (ii) I-NUC (inside nucleus), (iii) B-SAT (beginning satellite), and (iv) I-SAT (inside satellite). Therefore, this approach performs segmentation and nuclearity assignment simultaneously. Second, *two-step classification*, where in the first step, the discourse segmenter, a binary classifier, labels each token as either B (beginning) or I (inside). Then, in the second step, a nuclearity labeler, another binary classifier, assigns nuclearity statuses to the segments. The two-step approach avoids illegal chunk sequences like a B-NUC followed by an I-SAT or a B-SAT followed by an I-NUC, and in this approach, it is easier to incorporate sentence-level properties like the requirement that a sentence must contain at least one nucleus. The evaluation on RST-DT shows that

[5] http://www.isi.edu/licensed-sw/spade/

the two-step approach outperforms the one-step approach, and the performance is comparable to the performance of SPADE.

In follow-up work, Fisher and Roark [72] demonstrate over 4% absolute performance gain in discourse segmentation, by combining the features extracted from the syntactic tree with the ones derived via POS tagger and chunker. Using quite a large number of features in a binary log-linear model they achieve state-of-the-art performance in segmentation on the RST-DT test set.

Recently, Hernault et al. [84] present the publicly available **HILDA** system[6] that comes with a segmenter and a parser based on Support Vector Machines (SVMs). The segmenter is a binary SVM classifier which uses the same lexico-syntactic features used in SPADE, but with more context. The discourse parser iteratively employs two SVM classifiers in pipeline to build a DT. In each iteration, a binary classifier first decides which of the adjacent units to merge, then a multi-class classifier connects the selected units with an appropriate relation label. They report state-of-the-art performance in discourse parsing on the RST-DT corpus.

On a different genre of instructional texts, Subba and Di-Eugenio [201] propose a shift-reduce parser that relies on a classifier for relation labeling. Their classifier uses **Inductive Logic Programming (ILP)** to learn first-order logic rules from a large set of features including the linguistically rich *compositional semantics* coming from a semantic parser. They demonstrate that including compositional semantics with other features improves relation classification performance.

Both HILDA and the ILP-based approach of Subba and Di-Eugenio [201] are limited in several ways. First, they do not differentiate between intra-sentential parsing and multi-sentential parsing, and use a single uniform model in both scenarios. Second, they take a greedy (sub-optimal) approach to construct a DT. Third, they disregard sequential dependencies between DT constituents, which has been recently shown to be critical by Feng and Hirst [69]. Furthermore, HILDA considers the structure and the labels of a DT separately. Our novel discourse parser, proposed in this chapter, addresses all these limitations of the existing parsers.

[6] http://nlp.prendingerlab.net/hilda/

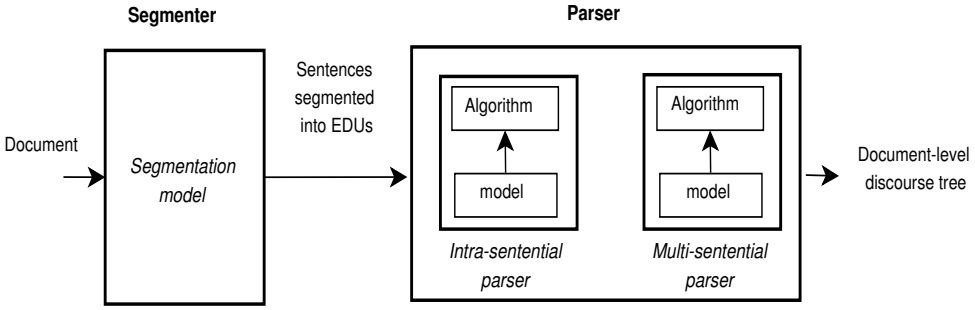



**Figure 3.2:** Rhetorical analysis framework.

## 3.3 Our Rhetorical Analysis Framework

Given a text, the first task in our rhetorical analysis pipeline (Figure 3.2) is to break the text into a sequence of Elementary Discourse Units (EDUs), i.e., discourse segmentation. Since it is taken for granted that sentence boundaries are also EDU boundaries (i.e., EDUs do not span across multiple sentences), the segmentation task boils down to finding EDU boundaries inside sentences.

Once we have identified the EDUs, the discourse parsing problem is determining which discourse units (EDUs or larger units) to relate (i.e., the structure), and how to relate them (i.e., the labels or the rhetorical relations) in the process of building the hierarchical DT. Specifically, it requires: *a parsing model* to explore the search space of possible structures and labels for their nodes; and *a parsing algorithm* for deciding among the candidates. Unlike previous studies [84, 124, 201], which follow a greedy approach, our approach to discourse parsing applies an optimal parsing algorithm to the probabilities of all possible DT constituents to find the most probable DT. A simple and straightforward strategy would be to use a single unified parsing model for both sentence-level and document-level parsing without distinguishing the two cases, as was previously done in [84, 124, 201]. However, this approach would be problematic in our case because of scalability and modeling issues. Note that the number of valid trees grows exponentially with the number of EDUs in a document.[7] Therefore, an exhaustive search over all the valid discourse trees is often unfeasible, even for relatively small documents.

For modeling, the problem is two-fold. On the one hand, it appears that rhetori-

[7] For $n+1$ EDUs, the number of valid discourse trees is actually the *Catalan number* $C_n$.

cal relations are distributed differently intra-sententially vs. multi-sententially. For example, Figure 3.3 shows a comparison between the two distributions of the six most frequent relations on a development set containing 20 randomly selected documents from the RST-DT corpus. Notice that relations *Attribution* and *Same-Unit* are more frequent than *Joint* in the intra-sentential case, whereas *Joint* is more frequent than the other two in the multi-sentential case. On the other hand, different kinds of features are applicable and informative for intra-sentential vs. multi-sentential parsing. For example, syntactic features like *dominance sets* [191] are extremely useful for sentence-level discourse parsing, but are not even applicable in the multi-sentential case. Likewise, *lexical chain features* [193], that are useful for multi-sentential parsing, are not applicable at the sentence level.

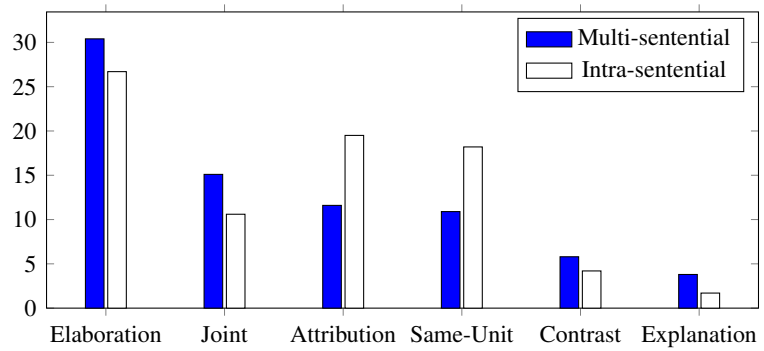



**Figure 3.3:** Distributions of six most frequent relations in intra-sentential and multi-sentential parsing scenarios.

Based on the above observations, our discourse parsing framework comprises two separate modules: an *intra-sentential parser* and a *multi-sentential parser* (Figure 3.2). 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. Both of our parsers have the same two components: a *parsing model* assigns a probability to every possible DT, and a *parsing algorithm* identifies the most probable DT among the candidate DTs in that scenario. While the two parsing models are rather different, the same parsing algorithm is shared by the two modules. Staging multi-sentential parsing on top of intra-sentential parsing in this way allows us to exploit the strong correlation observed between the text

structure and the DT structure as explained in detail in Section 3.4.3.

## 3.4 The Discourse Parser

Before describing our parsing models and the parsing algorithm in detail we introduce some terminology that we will use throughout the chapter.

A DT can be formally represented as a set of constituents of the form $R[i,m,j]$, where $i \leq m < j$. This refers to a rhetorical relation $R$ between the discourse unit containing EDUs $i$ through $m$ and the unit containing EDUs $m+1$ through $j$. For example, the DT for the second sentence in Figure 3.1 can be represented as $\{$*Elaboration-NS[4,4,5], Same-Unit-NN[4,5,6]*$\}$. Notice that a relation $R$ also specifies the nuclearity statuses of the discourse units involved, which can be one of *Nucleus-Satellite (NS), Satellite-Nucleus (SN) or Nucleus-Nucleus (NN)*.

A common assumption made for building discourse trees effectively is that they are *binary* trees [60, 191]. That is, multi-nuclear relations (e.g., *Joint, Same-Unit*) involving more than two discourse units are mapped to a hierarchical right-branching binary tree. For example, a flat $Joint(e_1,e_2,e_3,e_4)$ is mapped to a right-branching binary tree $Joint(e_1,Joint(e_2,Joint(e_3,e_4)))$.

### 3.4.1 Parsing Models

As mentioned before, the job of our intra-sentential and multi-sentential parsing models is to assign a probability to each of the constituents of all possible DTs at the sentence level and at the document level, respectively. Formally, given the model parameters $\Theta$, for each possible constituent $R[i,m,j]$ in a candidate DT at the sentence or document level, the parsing model estimates $P(R[i,m,j]|\Theta)$, which specifies a joint distribution over the label $R$ and the structure $[i,m,j]$ of the constituent. For example, when applied to the sentences in Figure 3.1 separately, our intra-sentential parsing model with (learned) parameters $\Theta$ estimates $P(R[1,1,2]|\Theta)$, $P(R[2,2,3]|\Theta)$, $P(R[1,2,3]|\Theta)$ and $P(R[1,1,3]|\Theta)$ for the first sentence, and $P(R[4,4,5]|\Theta)$, $P(R[5,\ 5,6]|\Theta)$, $P(R[4,5,6]|\Theta)$ and $P(R[4,4,6]|\Theta)$ for the second sentence, respectively, for all $R$ ranging on the set of relations.

### Intra-Sentential Parsing Model

Our novel probabilistic model for sentence-level discourse parsing is shown in Figure 3.4. The observed nodes $U_j$ 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 connections between adjacent nodes in a hidden layer encode sequential dependencies between the respective hidden nodes, and can enforce constraints such as the fact that a $S_j$= 1 must not follow a $S_{j-1}$= 1. The connections between the two hidden layers model the structure and the relation of DT constituents jointly.

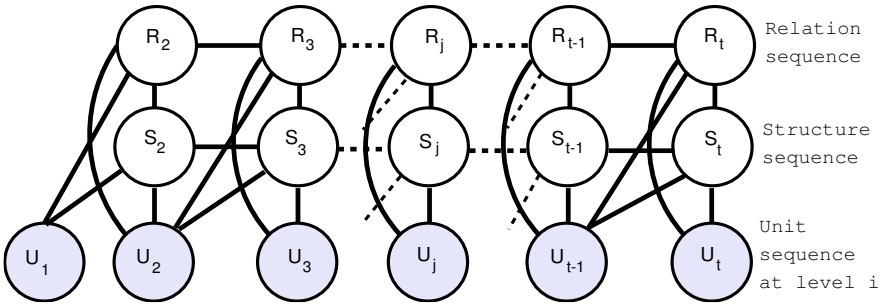



**Figure 3.4:** A chain-structured DCRF as our intra-sentential parsing model.

Notice that the graphical model in Figure 3.4 is a chain-structured undirected graphical model (also known as Markov Random Field (MRF)) with two hidden layers (chains). It becomes a **Dynamic Conditional Random Field (DCRF)** [205] when we directly model the hidden (output) variables by conditioning the clique potentials (factors) on the observed (input) variables:

$$P(R_{2:t}, S_{2:t}|\mathbf{x}, \Theta) = \frac{1}{Z(\mathbf{x}, \Theta)} \prod_{i=2}^{t-1} \phi(R_i, R_{i+1}|\mathbf{x}, \Theta) \psi(S_i, S_{i+1}|\mathbf{x}, \Theta) \omega(R_i, S_i|\mathbf{x}, \Theta) \tag{3.1}$$

where $\{\phi\}$ and $\{\psi\}$ are the factors over the edges of the relation and structure chains, respectively, and $\{\omega\}$ are the factors over the edges connecting the relation

and structure nodes (i.e., between-chain edges). Here, $\mathbf{x}$ represents input features extracted from the observed variables, and $Z(\mathbf{x},\Theta)$ is the partition function. We use the standard log-linear representation of the factors:

$$\phi(R_i,R_{i+1}|\mathbf{x},\Theta) = \exp(\Theta_r^T\ f(R_i,R_{i+1},\mathbf{x})) \quad (3.2)$$

$$\psi(S_i,S_{i+1}|\mathbf{x},\Theta) = \exp(\Theta_s^T\ f(S_i,S_{i+1},\mathbf{x})) \quad (3.3)$$

$$\omega(R_i,S_i|\mathbf{x},\Theta) = \exp(\Theta_c^T\ f(R_i,S_i,\mathbf{x})) \quad (3.4)$$

where $f(Y,Z,\mathbf{x})$ is a feature vector derived from the input features $\mathbf{x}$ and the local labels $Y$ and $Z$, and $\Theta_y$ is the corresponding weight vector.

A DCRF is a generalization of linear-chain CRFs [110] to represent complex interactions between labels, such as when performing multiple labeling tasks on the same sequence. Recently, there has been an explosion of interest in CRFs for solving structured output classification problems, with many successful applications in NLP including syntactic parsing [71], syntactic chunking [183] and discourse chunking [75] in the Penn Discourse Treebank [164]. CRFs, being a discriminative approach to sequence modeling, have several advantages over their generative counterparts such as Hidden Markov Models (HMMs) and MRFs, which first model the joint distribution $p(\mathbf{y},\mathbf{x}|\Theta)$, then infer the conditional distribution $p(\mathbf{y}|\mathbf{x},\Theta)$). It has been advocated that discriminative models are generally more accurate than generative ones since they do not "waste resources" modeling complex distributions that are observed (i.e., $p(\mathbf{x})$), instead they focus on modeling what we care about, i.e., the distribution of labels given the data [145]. Other key advantages include the ability to incorporate arbitrary overlapping local and global features, and the ability to relax strong independence assumptions. Furthermore, CRFs surmount the *label bias* problem [110] of the Maximum Entropy Markov Model (MEMM), which is considered to be a discriminative version of the HMM.

To obtain the probability of the constituents of all candidate DTs for a sentence, we apply our parsing model recursively at different levels of the DT and compute the posterior marginals over the relation-structure pairs. To illustrate the process, let us assume that the sentence contains four EDUs. At the first (bottom) level, when all the discourse units are the EDUs, there is only one possible unit sequence

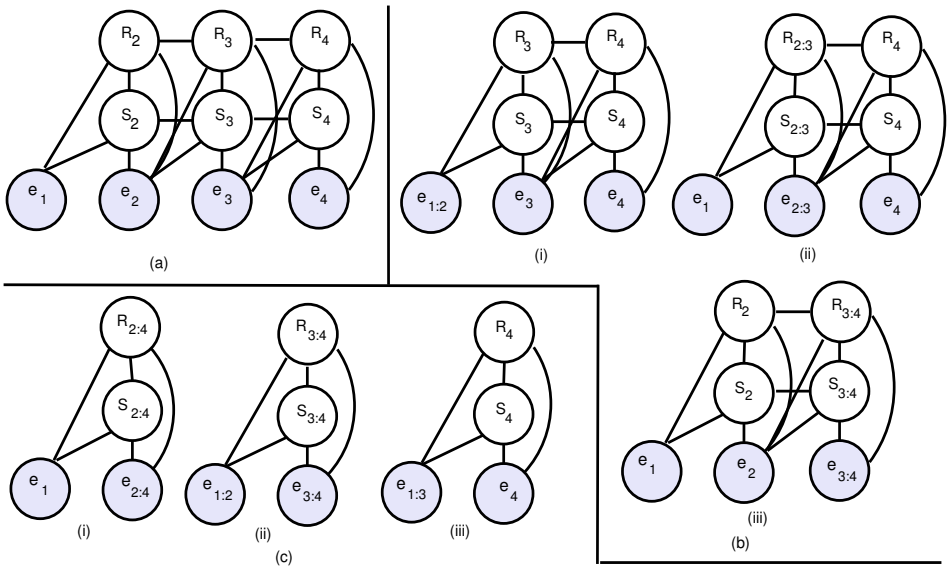



**Figure 3.5:** Our parsing model applied to the sequences at different levels of a sentence-level discourse tree. (a) Only possible sequence at the first level, (b) Three possible sequences at the second level, (c) Three possible sequences at the third level.

to which we apply our DCRF model (Figure 3.5(a)). We compute the posterior marginals $P(R_2, S_2{=}1|e_1, e_2, e_3, e_4, \Theta)$, $P(R_3, S_3{=}1|e_1, e_2, e_3, e_4, \Theta)$ and $P(R_4, S_4{=}1|e_1, e_2, e_3, e_4, \Theta)$ to obtain the probability of the constituents $R[1,1,2]$, $R[2,2,3]$ and $R[3,3,4]$, respectively. At the second level, there are three possible unit sequences $(e_{1:2}, e_3, e_4)$, $(e_1, e_{2:3}, e_4)$ and $(e_1, e_2, e_{3:4})$. Figure 3.5(b) shows their corresponding DCRF models. The posterior marginals $P(R_3, S_3{=}1|e_{1:2}, e_3, e_4, \Theta)$, $P(R_{2:3} S_{2:3}{=}1|e_1, e_{2:3}, e_4, \Theta)$, $P(R_4, S_4{=}1|e_1, e_{2:3}, e_4, \Theta)$ and $P(R_{3:4}, S_{3:4}{=}1|e_1, e_2, e_{3:4}, \Theta)$ computed from the three sequences correspond to the probability of the constituents $R[1,2,3]$, $R[1,1,3]$, $R[2,3,4]$ and $R[2,2,4]$, respectively. Similarly, we attain the probability of the constituents $R[1,1,4]$, $R[1,2,4]$ and $R[1,3,4]$ by computing their respective posterior marginals from the three possible sequences at the third (top) level.

At this point what is left to be explained is how we generate all possible sequences for a given number of EDUs in a sentence. Algorithm 1 demonstrates how we do that. More specifically, to compute the probabilities of each DT constituent $R[i,k,j]$, we need to generate sequences like $(e_1, \cdots, e_{i-1}, e_{i:k}, e_{k+1:j}, e_{j+1}, \cdots, e_n)$

for $1 \leq i \leq k < j \leq n$. In doing so, we may generate some duplicate sequences. Clearly, the sequence $(e_1, \cdots, e_{i-1}, e_{i:i}, e_{i+1:j}, e_{j+1}, \cdots, e_n)$ for $1 \leq i \leq k < j < n$ is already considered for computing the probability of $R[i+1, j, j+1]$. Therefore, it is a duplicate sequence that we exclude from our list of all possible sequences.

```
Input: Sequence of EDUs: (e_1, e_2, ..., e_n)
Output: List of sequences: L
for i = 1 → n − 1 do
    for j = i + 1 → n do
        if j == n then                    // sequences at top and bottom levels
            for k = i → j − 1 do
                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      // excludes 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
```

**Algorithm 1:** Generating all possible sequences for a sentence with $n$ EDUs.

Once we acquire the probability of all possible DT constituents, the discourse sub-trees for the sentences are built by applying an optimal probabilistic parsing algorithm (Section 3.4.2) using one of the methods described in Section 3.4.3.

### Multi-Sentential Parsing Model

Given the discourse units (sub-trees) for all the sentences in a document, a simple approach to build the rhetorical parse tree of the document would be to apply a new DCRF model, similar to the one in Figure 3.4 (with different parameters), to all the possible sequences generated from these units to infer the probability of all possible higher-order constituents. However, the number of possible sequences and their length increase with the number of sentences in a document. For example, assuming that each sentence has a well-formed DT, for a document with $n$ sentences, algorithm 1 generates $O(n^3)$ sequences, where the sequence at the bottom

level has $n$ units, each of the sequences at the second level has $n$-1 units, and so on. Since the model in Figure 3.4 has a "fat" chain structure, we could use the forwards-backwards algorithm for exact inference in this model [204]. However, forwards-backwards on a sequence containing $T$ units costs $O(TM^2)$ time, where $M$ is the number of relations in our relation set. This makes the chain-structured DCRF model impractical for multi-sentential parsing of long documents, since learning requires to run inference on every training sequence with an overall time complexity of $O(TM^2n^3)$ per document.

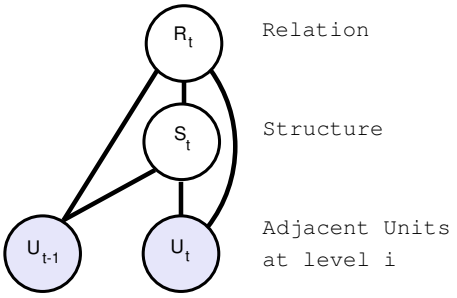



**Figure 3.6:** A CRF as a multi-sentential parsing model.

Our model for multi-sentential parsing is shown in Figure 3.6. The two observed nodes $U_{t-1}$ and $U_t$ are two adjacent discourse units. The (hidden) structure node $S \in \{0,1\}$ denotes whether the two discourse units should be linked or not. The other hidden node $R \in \{1 \ldots M\}$ represents the relation between the two units. Notice that similar to the model in Figure 3.4, this is also an undirected graphical model and becomes a **CRF** model if we directly model the labels by conditioning the clique potential $\phi$ on the input features $\mathbf{x}$, derived from the observed variables:

$$P(R_t, S_t|\mathbf{x}, \Theta) = \frac{1}{Z(\mathbf{x}, \Theta)} \phi(R_t, S_t|\mathbf{x}, \Theta) \quad (3.5)$$

$$\phi(R_t, S_t|\mathbf{x}, \Theta) = \exp(\Theta^T f(R_t, S_t, \mathbf{x})) \quad (3.6)$$

where $f(R_t, S_t, \mathbf{x})$ is a feature vector derived from the input features $\mathbf{x}$ and the labels $R_t$ and $S_t$, and $\Theta$ is the corresponding weight vector. Although this model is similar in spirit to the model in Figure 3.4, we now break the chain structure, which makes

the inference much faster (i.e., complexity of $O(M^2)$). Breaking the chain structure also allows us to balance the data for training (an equal number of instances with $S$=1 and $S$=0), which dramatically reduces the learning time of the model.

We apply our model to all possible adjacent units at all levels in the multi-sentential case, and compute the posterior marginals of the relation-structure pairs $P(R_t, S_t{=}1|U_{t-1}, U_t, \Theta)$ to obtain the probability of all possible DT constituents. Given the sentence-level discourse units, the following algorithm, which is a simplified variation of algorithm 1, extracts all possible adjacent units for a document.

**Input**: Sequence of units: $(U_1, U_2, \cdots U_n)$, where $U_x[0]{:=}$ start EDU ID of unit $x$, and $U_x[1]{:=}$ end EDU ID of unit $x$.
**Output**: List of adjacent units: $L$
**for** $i = 1 \rightarrow n-1$ **do**
  **for** $j = i+1 \rightarrow n$ **do**
    **for** $k = i \rightarrow j-1$ **do**
      $Left = U_i[0] : U_k[1]$
      $Right = U_{k+1}[0] : U_j[1]$
      $L$.append $((Left, Right))$
    **end**
  **end**
**end**

**Algorithm 2:** Generating all possible adjacent units at all levels of a document-level discourse tree.

Both of our CRF models (intra-sentential and multi-sentential) are designed using MALLET's graphical model toolkit GRMM [132]. In order to avoid over-fitting, we regularize the CRF models with $l_2$ regularization and learn the model parameters using the limited-memory BFGS (L-BFGS) fitting algorithm.

### Features Used in our Parsing Models

Crucial to parsing performance is the set of features used in the parsing models, as summarized in table 3.1. Table 3.1 also specifies in what parsing model each feature is used. Notice that some of the features are used in both models. The features are extracted from two adjacent discourse units $U_{t-1}$ and $U_t$. Most of the features have been explored in previous studies (e.g., [84, 191, 194]). However, we

improve some of these as explained below.

| **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. | |

**Table 3.1:** Features used in our intra- and multi-sentential parsing models.

**Organizational** features encode useful information about text organization as shown by duVerle and Prendinger [60]. We measure the length of the units as the number of *EDUs* and *tokens* in it. However, in order to better adjust to the length variations, rather than computing their absolute numbers in a unit, we choose to measure their *relative numbers* with respect to their total numbers in the two units. For example, if the two units under consideration contain three EDUs in total, a

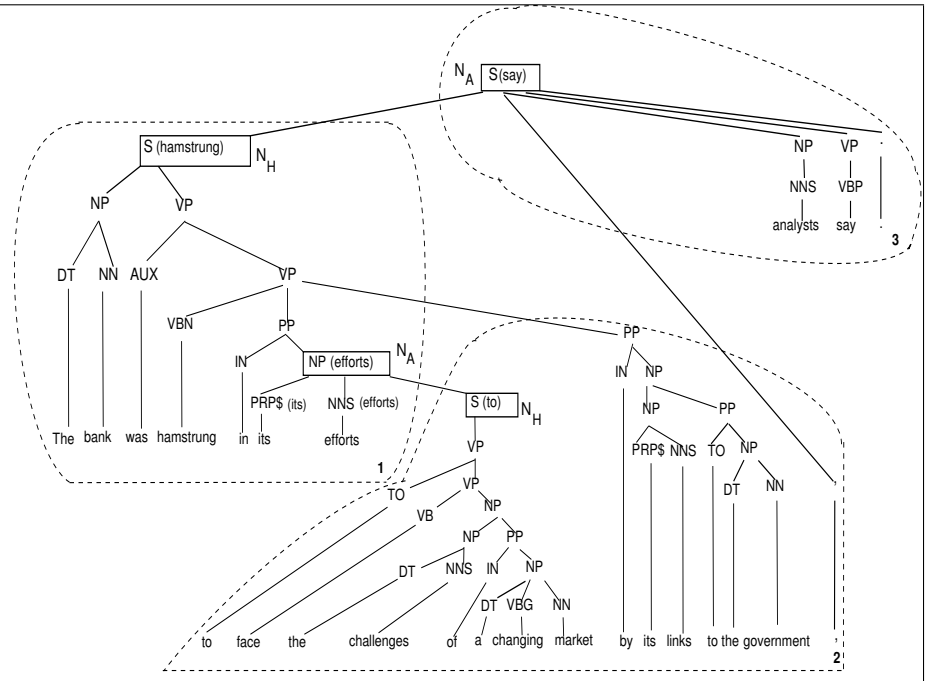
unit containing two of the EDUs will have a relative EDU number of 0.67. We also measure the *distances* of the units in terms of the number of EDUs from the beginning and end of the sentence (or text in the multi-sentential case). **Text structural** features capture the correlation between text structure and rhetorical structure by counting the number of *sentence* and *paragraph* boundaries in the units.

Discourse cues (e.g., *because, but*), when present, signal rhetorical relations between two text segments [106, 125]. However, recent studies [22, 84] suggest that an empirically acquired *lexical* **N-gram** dictionary is more effective than a fixed list of cue phrases, since this approach is domain independent and capable of capturing non-lexical cues such as punctuation. To build the lexical N-gram dictionary empirically from the training corpus we consider the first and last $N$ tokens ($N \in \{1,2,3\}$) of each unit and rank them according to their mutual information with the two labels, *Structure* and *Relation*.[8] Intuitively, the most informative cues are not only the most frequent, but also the ones that are indicative of the labels in the training data [28]. In addition to the lexical N-grams we also encode the *POS* tags of the first and last $N$ tokens ($N \in \{1,2,3\}$) in a unit as shallow-syntactic features.

Lexico-syntactic features **dominance sets** extracted from the Discourse Segmented Lexicalized Syntactic Tree (DS-LST) of a sentence has been shown to be extremely effective for intra-sentential discourse parsing in SPADE [191]. Figure 3.7(a) shows the DS-LST (i.e., lexicalized syntactic tree with EDUs identified) for a sentence with three EDUs from the RST-DT corpus, and Figure 3.7(b) shows the corresponding discourse tree. In a DS-LST, each EDU except the one with the root node must have a *head node* $N_H$ that is attached to an *attachment node* $N_A$ residing in a separate EDU. A dominance set $D$ (shown at the bottom of Figure 3.7(a)) contains these *attachment* points (shown in boxes) of the EDUs in a DS-LST. In addition to the syntactic and lexical information of the head and attachment nodes, each element in $D$ also includes a dominance relationship between the EDUs involved. The EDU with $N_A$ dominates (represented by '$>$') the EDU with $N_H$.

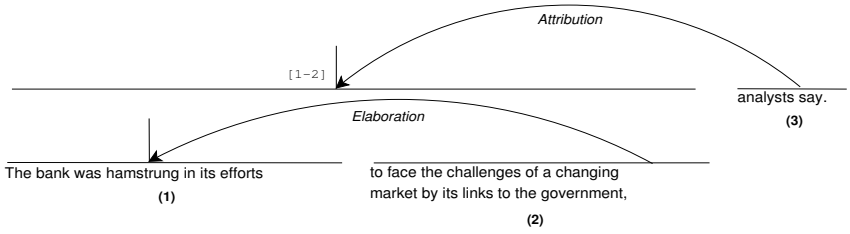
Soricut and Marcu [191] hypothesize that the dominance set (i.e., lexical heads,

[8] In contrast, HILDA [84] ranks the N-grams by their frequencies in the training corpus.




*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.




**(b)** Discourse tree for the above sentence.

**Figure 3.7:** Dominance set features for intra-sentential discourse parsing.

syntactic labels and dominance relationships) carries the most informative clues for intra-sentential parsing. For instance, the dominance relationship between the EDUs in our example sentence is $3 > 1 > 2$, which favors the DT structure $[1,1,2]$ over $[2,2,3]$. In order to extract dominance set features for two adjacent units $U_{t-1}$ and $U_t$, containing EDUs $e_{i:j}$ and $e_{j+1:k}$, respectively, we first compute $D$ from the DS-LST of the sentence. We then extract the element from $D$ that holds across the EDUs $j$ and $j+1$. In our example, for the two units, containing EDUs $e_1$ and $e_2$, respectively, the relevant dominance set element is *(1, efforts/NP)>(2, to/S)*. We encode the syntactic labels and lexical heads of $N_H$ and $N_A$ and the dominance relationship as features in our intra-sentential parsing model.

As described in Chapter 2, **Lexical chains** [143] are sequences of semantically related words that can indicate topical boundaries in a text. Features extracted from lexical chains are also shown to be useful for finding paragraph-level discourse structure [193]. For example, consider the text with 4 paragraphs ($P_1$ to $P_4$) in Figure 3.8(a). Now, let us assume that there is a lexical chain that spans the whole text, skipping paragraphs $P_2$ and $P_3$, while a second chain only spans $P_2$ and $P_3$. This situation makes it more likely that $P_2$ and $P_3$ should be linked in the DT before any of them is linked with another paragraph. Therefore, the DT structure in Figure 3.8(b) should be more likely than the structure in Figure 3.8(c).

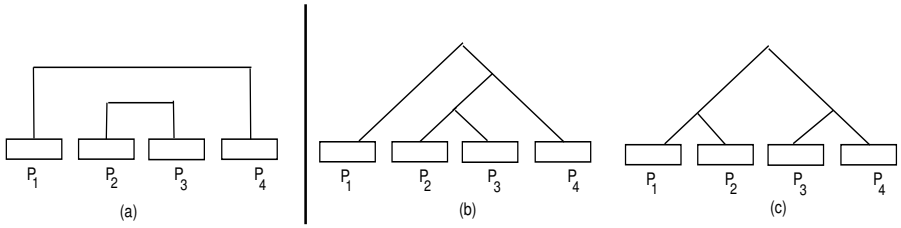



**Figure 3.8:** Correlation between lexical chains and discourse structure. (a) Lexical chains spanning paragraphs. (b) Two possible DT structures.

One challenge in computing lexical chains is that words can have multiple senses and semantic relationships depend on the sense rather than the word itself. Several methods have been proposed to compute lexical chains [17, 73, 86, 187]. We follow the approach proposed by Galley and McKeown [73], that extracts lexical chains after performing Word Sense Disambiguation (WSD). In the prepro-

cessing step, we extract the nouns from the document and lemmatize them using WordNet's built-in *morphy* function [67]. Then, by looking up in WordNet we expand each noun to all of its senses, and build a Lexical Semantic Relatedness Graph (LSRG). In a LSRG, the nodes represent noun-tokens with their candidate senses, and the weighted edges between senses of two different tokens represent one of the three semantic relations: *repetition, synonym* and *hypernym*. For example, Figure 3.9(a) shows a partial LSRG, where the token *bank* has two possible senses, namely *money bank* and *river bank*. Using the *money bank* sense, *bank* is connected with *institution* and *company* by hypernymy relations (edges marked with *H*), and with another *bank* by a repetition relation (edges marked with *R*). Similarly, using the *river bank* sense, it is connected with *riverside* by a hypernymy relation and with *bank* by a repetition relation. Nouns that are not found in WordNet are considered as proper nouns having only one sense, and are connected by a *repetition* relation.

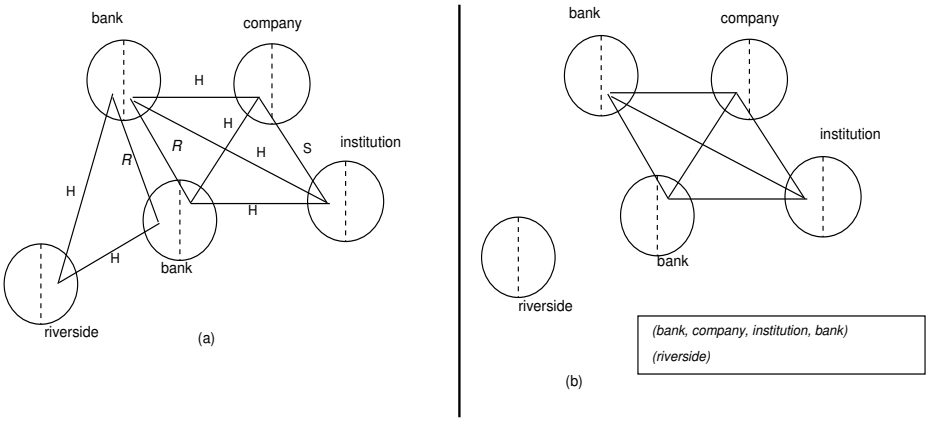



**Figure 3.9:** Extracting lexical chains. (a) A Lexical Semantic Relatedness Graph (LSRG) for five noun-tokens. (b) Resultant graph after performing WSD. The box at the bottom shows the lexical chains.

We use this LSRG first to perform WSD, then to construct lexical chains. For WSD, the weights of all edges leaving the nodes under their different senses are summed up and the one with the highest score is considered to be the right sense for the word-token. For example, if repetition and synonymy are weighted equally, and hypernymy is given half as much weight as either of them, the score of bank's two senses are: $1+0.5+0.5=2$ for the sense *money bank* and $1+0.5=1.5$ for

the sense *river bank*. Therefore, the selected sense for *bank* in this context is *river bank*. In case of a tie, we select the sense that is most frequent (i.e., the first sense in WordNet). Note that this approach to WSD is different from [193], which takes a greedy approach.

Finally, we prune the graph by only keeping the links that connect words with the selected senses. At the end of the process, we are left with the edges that form the actual lexical chains. For example, Figure 3.9(b) shows the result of pruning the graph in Figure 3.9(a). The lexical chains extracted from the pruned graph are shown in the box at the bottom. Following [193], for each chain element, we keep track of the location (i.e., sentence Id) in the text where that element was found, and exclude chains containing only one element. Given two discourse units, we count the number of chains that: hit the two units, exclusively hit the two units, skip both units, skip one of the units, start in a unit, and end in a unit.

We also consider more **contextual** information by including the above features computed for the neighboring adjacent unit pairs in the current feature vector. For example, the contextual features for units $U_{t-1}$ and $U_t$ includes the feature vector computed from $U_{t-2}$ and $U_{t-1}$ and the feature vector computed from $U_t$ and $U_{t+1}$.

We incorporate *hierarchical dependencies* between the constituents in a DT by rhetorical **sub-structural** features. For two adjacent discourse units $U_{t-1}$ and $U_t$, we extract the roots of the two rhetorical sub-trees. For our example in Figure 3.7(b), the root of the rhetorical sub-tree spanning over EDUs $e_{1:2}$ is *Elaboration-NS*. However, this assumes the presence of a labeled DT, which is not the case when we apply the parser to a new text (sentence or document). This problem can be easily solved by looping twice through building the parsing model and applying the parsing algorithm (see Section 3.4.2). We first build the model without considering the sub-structural features. Then we find the optimal DT employing our parsing algorithm. This intermediate DT will now provide labels for the sub-structures. Next we can build a new, more accurate model by including the sub-structure features, and run again the parsing algorithm to find the final optimal DT.

In addition to the above features, we also experimented with other features including *WordNet-based lexical semantics*, *subjectivity* and *TF.IDF-based cosine similarity*. However, since such features did not improve parsing performance on our development set, they were excluded from our final set of features.

### 3.4.2 Parsing Algorithm

Our parsing models assign a probability to every possible DT constituent in the intra-sentential and multi-sentential scenarios. The job of the parsing algorithm is to find the most probable DT for the whole text. Formally, this can be written as,

$$DT^* = \underset{DT}{\operatorname{argmax}}\, P(DT|\Theta) \tag{3.7}$$

where $\Theta$ specifies the parameters of the parsing model (intra-sentential or multi-sentential). We implement a probabilistic CKY-like bottom-up algorithm for computing the most likely parse using dynamic programming (see [100] for details). Specifically, with $n$ discourse units, we use the upper-triangular portion of the $n \times n$ dynamic programming table $D$, where cell $D[i,j]$ (for $i < j$) stores:

$$D[i,j] = P(r^*[U_i(0), U_{k^*}(1), U_j(1)]) \tag{3.8}$$

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

$$(k^*, r^*) = \underset{i \le k \le j\ ;\ R \in \{1 \cdots M\}}{\operatorname{argmax}} P(R[U_i(0), U_k(1), U_j(1)]) \times D[i,k] \times D[k+1,j] \tag{3.9}$$

Recall that the notation $R[U_i(0), U_k(1), U_j(1)]$ refers to a rhetorical relation $R$ between the discourse unit containing EDUs $U_i(0)$ through $U_k(1)$ and the unit containing EDUs $U_k(1)+1$ through $U_j(1)$. In addition to $D$, which stores the *probability* of the most probable constituents of a DT, we also simultaneously maintain two other $n \times n$ dynamic programming tables $S$ and $R$ for storing the structure (i.e., $U_{k^*}(1)$) and the relations (i.e., $r^*$) of the corresponding DT constituents, respectively. For example, given 4 EDUs $e_1 \cdots e_4$, the $S$ and $R$ tables at the left of Figure 3.10 together represent the DT shown at the right. To find the DT, we first look at the top-right entries in the two tables; here $S[1,4] = 2$ and $R[1,4] = r_2$ specify that the two discourse units $e_{1:2}$ and $e_{3:4}$ should be connected by the relation $r_2$ (the root in the DT). Then, we see how EDUs $e_1$ and $e_2$ should be connected by looking at the entries $S[1,2]$ and $R[1,2]$; here $S[1,2] = 1$ and $R[1,2] = r_1$ indicate that they should be connected by the relation $r_1$ (the left non-terminal in the DT). Finally, to

see how EDUs $e_3$ and $e_4$ should be linked, we look at the entries $S[3,4]$ and $R[3,4]$, which tell us that they should be linked by the relation $r_4$ (the right non-terminal).

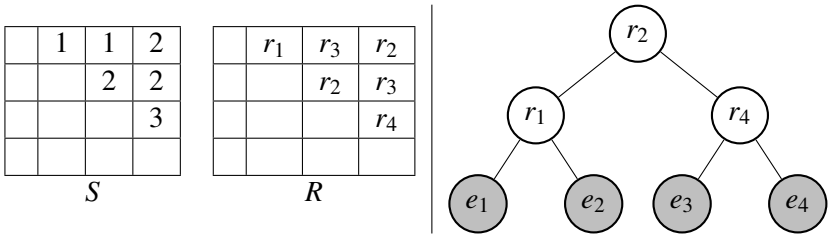



**Figure 3.10:** The $S$ and $R$ dynamic programming tables (left), and the corresponding discourse tree (right).

Note that, in contrast to previous studies on document-level discourse parsing [84, 126, 201], which use a greedy algorithm, our approach finds a discourse tree that is globally optimal. Our approach is also different from the sentence-level discourse parser SPADE [191]. SPADE first finds the *tree structure* that is globally optimal, then it assigns the most probable *relations* to the internal nodes. More specifically, the cell $D[i,j]$ in SPADE's dynamic programming table stores:

$$D[i,j] = P([U_i(0), U_k(1), U_j(1)]) \tag{3.10}$$

where $k = \underset{i \leq p \leq j}{\operatorname{argmax}} P([U_i(0), U_p(1), U_j(1)])$. Disregarding the relation label $R$ while populating $D$, this approach may find a discourse tree that is not globally optimal.

### 3.4.3 Document-level Parsing Approaches

Now that we have presented our intra-sentential and our multi-sentential parsers, we are ready to describe how they can be effectively combined to perform document-level rhetorical analysis. Recall that a key motivation for two-stage parsing is that it allows us to capture the strong correlation between text structure and discourse structure in a scalable, modular and flexible way. In the following, we describe two different approaches to model this correlation.

### 1S-1S (1 Sentence-1 Sub-tree)

A key finding from previous studies on sentence-level rhetorical analysis is that most sentences have a well-formed discourse sub-tree in the full DT [72, 191]. For example, Figure 3.11(a) shows 10 EDUs in 3 sentences (see boxes), where the DTs for the sentences obey their respective sentence boundaries. The 1S-1S approach aims to maximally exploit this finding. It first constructs a DT for every sentence using our intra-sentential parser, and then it provides our multi-sentential parser with the sentence-level DTs to build the rhetorical parse for the whole document.

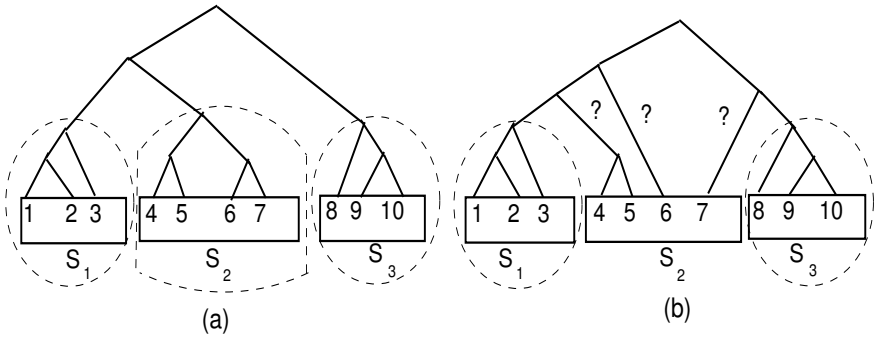



**Figure 3.11:** Two possible DTs for three sentences.

### Sliding Window

While the assumption made by 1S-1S clearly simplifies the parsing process, it totally ignores the cases where rhetorical structures violate sentence boundaries. For example, in the DT shown in Figure 3.11(b), sentence $S_2$ does not have a well-formed sub-tree because some of its units attach to the left (4-5, 6) and some to the right (7). Vliet and Redeker [218] call these cases 'leaky' boundaries. Even though less than 5% of the sentences have leaky boundaries in RST-DT, in other corpora this can be true for a larger portion of the sentences. For example, we observe over 12% of sentences with leaky boundaries in the Instructional corpus of Subba and Di-Eugenio [201]. However, we notice that in most cases where DT structures violate sentence boundaries, its units are merged with the units of its adjacent sentences, as in Figure 3.11(b). For example, this is true for 75% of cases in our development set containing 20 news articles from RST-DT and for 79% of cases

in our development set containing 20 how-to-do manuals from the Instructional corpus. Based on this observation, we propose a sliding window approach.

In this approach, our 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 3.11, our intra-sentential parser constructs a DT for $S_1$-$S_2$ and a DT for $S_2$-$S_3$. In this process, each sentence in a document except the first and the last will be associated with two DTs: one with the previous sentence (say $DT_p$) and one with the next (say $DT_n$). In other words, for each non-boundary sentence, we will have two decisions: one from $DT_p$ and one from $DT_n$. Our parser consolidates the two decisions and generates one or more sub-trees for each sentence by checking the following three mutually exclusive conditions one after another:

- *Same in both:* If the sentence under consideration has the same (in both structure and labels) well-formed sub-tree in both $DT_p$ and $DT_n$, we take this sub-tree. For example, in Figure 3.12(a), $S_2$ has the same sub-tree in the two DTs (one for $S_1$-$S_2$ and one for $S_2$-$S_3$). The two decisions agree on the DT for the sentence.

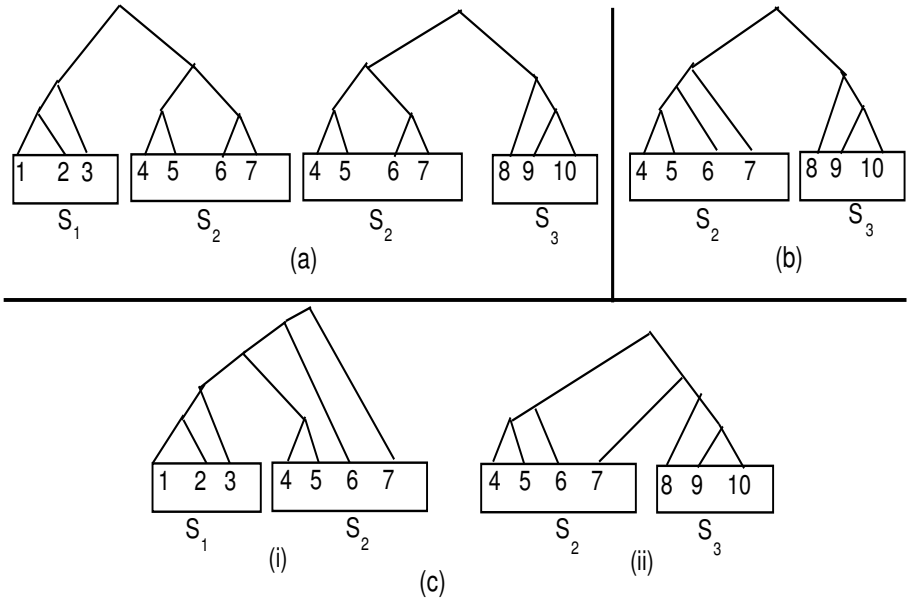



**Figure 3.12:** Extracting sub-trees for $S_2$.

- *Different but no cross:* If the sentence under consideration has a well-formed

sub-tree in both $DT_p$ and $DT_n$, but the two sub-trees vary either in structure or in labels, we pick the most probable one. For example, consider the DT for $S_1$-$S_2$ in Figure 3.12(a) and the DT for $S_2$-$S_3$ in Figure 3.12(b). In both cases $S_2$ has a well-formed sub-tree, but they differ in structure. We pick the sub-tree which has the higher probability in the two dynamic programming tables.

- *Cross:* If either or both of $DT_p$ and $DT_n$ segment the sentence into multiple sub-trees, we pick the one with more sub-trees. For example, consider the two DTs in Figure 3.12(c). In the DT for $S_1$-$S_2$, $S_2$ has three sub-trees (4-5,6,7), whereas in the DT for $S_2$-$S_3$, it has two (4-6,7). So, we extract the three sub-trees for $S_2$ from the first DT. If the sentence has the same number of sub-trees in both $DT_p$ and $DT_n$, we pick the one with higher probability in the dynamic programming tables.

At the end, the multi-sentential parser takes all these sentence-level sub-trees for a document, and builds a full rhetorical parse for the whole document.

## 3.5 The Discourse Segmenter

Our discourse parser above assumes that the input text has been already segmented into a sequence of EDUs. However, discourse segmentation is also a challenging problem, and previous studies [72, 191] have identified it as a primary source of inaccuracy for discourse parsing. Regardless of discourse parsing, segmentation itself can be useful in several NLP applications including sentence compression [194] and textual alignment in machine translation [198]. Therefore, we have developed our own discourse segmenter, that not only achieves state-of-the-art performance as shown later, but also reduces the time complexity by using fewer features.

### 3.5.1 Segmentation Model

Our discourse segmenter implements a binary classifier to decide for each word-token (except the last) in a sentence, whether to place an EDU boundary after that word. We use a **maximum entropy** model to build a discriminative classifier. Specifically, we use a Logistic Regression (LR) classifier with $l_2$ regularization:

$$P(y|w,\Theta) = \text{Ber}(y|\,\text{Sigm}(\Theta^T\,\mathbf{x})) + \lambda\Theta^T\Theta \tag{3.11}$$

where the output $y \in \{0,1\}$ denotes whether to put an EDU boundary ($y = 1$) or not ($y = 0$) after the word-token $w$, which is represented using a feature vector $\mathbf{x}$. In the equation, $\text{Ber}(\eta)$ and $\text{Sigm}(\eta)$ refer to the *Bernoulli* distribution and *Sigmoid* (also known as logistic) function, respectively. We learn the model parameters $\Theta$ using the L-BFGS fitting algorithm, which as described in Chapter 2, is time and space efficient. To avoid overfitting, we use 5-fold cross validation to learn the regularization strength parameter $\lambda$ from the training data. We also use a simple *bagging* technique [30] to deal with the sparsity of *boundary* (i.e., $y = 1$) tags. Note that our first attempt at the segmentation task implemented a linear-chain CRF model [110] to capture the sequential dependencies between the tags in a discriminative way. However, the binary LR classifier, using the same set of features, not only outperforms the CRF model, but also reduces the time and space complexities. This is not surprising because given the sparsity of the boundary (i.e., $y = 1$) tags, Markov dependencies between tags do not deliver additional improvement. Also, since we could not balance the data by using techniques like bagging in the CRF model, this further degrades the performance.

### 3.5.2 Features Used in the Segmentation Model

Our set of features for discourse segmentation are mostly inspired from previous studies but used in a novel way as we describe below.

Our first subset of features which we call **SPADE features**, includes the lexico-syntactic patterns extracted from the lexicalized syntactic tree of the given sentence. These features replicate the features used in SPADE's segmenter, but used in a discriminative way. To decide on an EDU boundary after a word-token $w_k$, we find the lowest constituent in the lexicalized syntactic tree that spans over tokens $w_i \ldots w_j$ such that $i \leq k < j$. The production that expands this constituent in the tree, with the potential EDU boundary marked, forms the primary feature. For instance, to determine the existence of an EDU boundary after the word *efforts* in our sample sentence shown in Figure 3.7, the production *NP(efforts)* $\rightarrow$ *PRP$(its) NNS(efforts)* $\uparrow$ *S(to)* extracted from the lexicalized syntactic tree in Figure 3.7(a) constitutes the primary feature, where $\uparrow$ denotes the potential EDU boundary.

SPADE predicts an EDU boundary if the relative frequency (i.e., Maximum

Likelihood Estimate (MLE)) of a potential boundary given the production in the training data is greater than 0.5. If the production has not been observed frequently enough, the unlexicalized version of the production, e.g., $NP \rightarrow PRP\$\ NNS \uparrow S$ is used for prediction. If the unlexicalized version is also found to be rare, other variations of the production depending on whether they include the lexical heads and how many non-terminals (one or two) they consider before and after the potential boundary are examined one after another (see [72] for details). In contrast, we compute the MLE estimates for a primary production (feature) and its other variations, and use those directly as features with/without binarizing the values.

Shallow syntactic features like **Chunk** and **POS** tags have been shown to possess valuable clues for discourse segmentation [72, 194]. For example, it is less likely that an EDU boundary occurs within a chunk. We annotate the tokens of a sentence with chunk and POS tags using the state-of-the-art Illinois tagger[9] and encode these as features in our model. Note that the chunker assigns each token a tag using the *BIO* notation, where *B* stands for beginning of a particular phrase (e.g., noun phrase, verb phrase), *I* stands for inside of a particular phrase and *O* stands for outside of a phrase. The rationale for using the Illinois chunker is that it uses a larger set of tags (23 in total), thus more informative than most of the other existing taggers, which typically use only 5 tags (B-NP, I-NP, B-VP, I-VP and O).

EDUs are normally multi-word strings. Thus, a token near the beginning or end of a sentence is unlikely to be the end of a segment. Therefore, for each token we include its **relative position** (i.e., absolute position/total number of tokens) in the sentence and **distances** to the beginning and end of the sentence as features.

It is unlikely that two consecutive tokens are tagged with EDU boundaries. Therefore, we incorporate **contextual** information for a token into our model by including the above features computed for its neighboring tokens.

We also experimented with different N-gram ($N \in \{1,2,3\}$) features extracted from the token sequence, POS sequence and chunk sequence. However, since such features did not improve segmentation accuracy on the development set, they were excluded from our final set of features.

[9] Available at http://cogcomp.cs.illinois.edu/page/software

## 3.6 Experiments

This section presents our experimental results. First, we describe the corpora on which the experiments were performed and the evaluation metrics used to measure the performance of the discourse segmenter and the discourse parsers. Then we show the performance of our discourse segmenter followed by the performance of our discourse parser.

### 3.6.1 Corpora

While previous studies on rhetorical analysis only report their results on a particular corpus, to demonstrate the generality of our method, we experiment with texts from two very different genres: news articles and instructional how-to-do manuals.

Our first corpus is the standard *RST-DT* [38], which contains discourse annotations for 385 Wall Street Journal articles taken from the Penn Treebank corpus [128]. The corpus is partitioned into a training set of 347 documents and a test set of 38 documents. 53 documents, selected from both sets were annotated by two annotators, based on which we measure human agreement. In RST-DT, the original 25 rhetorical relations defined by Mann and Thompson [122] are further divided into a set of 18 coarser relation classes with 78 finer-grained relations.

Our second corpus is the *Instructional* corpus prepared by Subba and Di-Eugenio [201], which contains discourse annotations for 176 how-to-do manuals on home-repair. The corpus was annotated with 26 informational relations (e.g., *Preparation-Act, Act-Goal*).

For our experiments with the intra-sentential parser, we extracted a sentence-level DT from a document-level DT by finding the subtree that exactly spans over the sentence. In RST-DT, by our count, 7321 out of 7673 sentences in the training set, 951 out of 991 sentences in the test set, and 1114 out of 1208 sentences in the doubly-annotated set have a well-formed DT. On the other hand, 3032 out of 3430 sentences in the Instructional corpus have a well-formed DT. This forms the corpora for our experiments with intra-sentential discourse parsing. However, the existence of a well-formed DT in not a necessity for discourse segmentation, therefore, we do not exclude any sentence in our discourse segmentation experiments.

### 3.6.2 Evaluation (and Agreement) Metrics

This section describes the metrics used to measure both how much the annotators agree with each other, and how well the systems perform when their outputs are compared with human annotations for the discourse analysis tasks.

#### Metrics for Discourse Segmentation

Since sentence boundaries are considered to be also the EDU boundaries, we evaluate segmentation performance with respect to the intra-sentential segment boundaries, which is a standard method for measuring segmentation accuracy [72, 191]. Specifically, if a sentence contains $n$ EDUs, which corresponds to $n-1$ intra-sentential segment boundaries, we measure the segmenter's ability to correctly identify these $n-1$ boundaries. Let $h$ be the total number of intra-sentential segment boundaries in the human annotation, $m$ be the total number of intra-sentential segment boundaries in the model output, and $c$ be the total number of correct segment boundaries in the model output. Then, we measure Precision (P), Recall (R) and $F_1$-score for segmentation performance as follows:

$$P = \frac{c}{m}, \qquad R = \frac{c}{h}, \qquad and \qquad F_1 - score = \frac{2PR}{P+R} = \frac{2c}{h+m}$$

#### Metrics for Discourse Parsing

To evaluate parsing performance, we use the standard unlabeled and labeled precision, recall and $F_1$-score as proposed by Marcu [126]. The unlabeled metrics measure how accurate the parser is in finding the right structure of the DT, while the labeled metrics measure the parser's ability to find the right labels (i.e., nuclearity and relation) in addition to the right structure. Assume for example that given the two sentences of Figure 3.1, our system generates the DT shown in Figure 3.13.

Figure 3.14 shows the same human-annotated DT shown in Figure 3.1 (Figure 3.14(a)) and the same system-generated DT shown in Figure 3.13 (Figure 3.14(b)) when we align the two structures. For the sake of illustration, instead of showing the real EDUs, we only show their IDs. Notice that the automatic segmenter breaks the EDU marked 2-3 in the human annotation into two EDUs, and did not identify

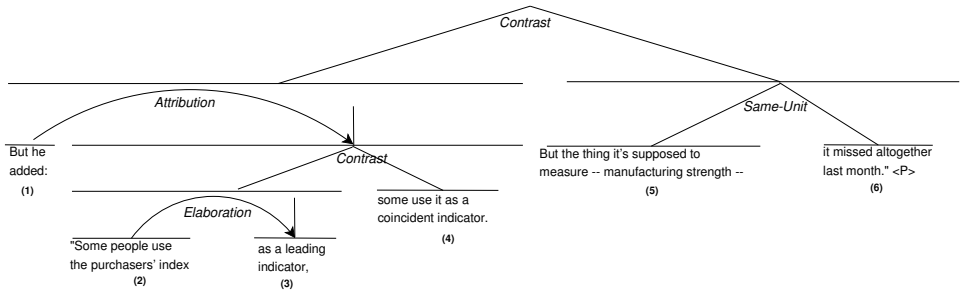



**Figure 3.13:** A hypothetical system-generated discourse tree for the two sentences in Figure 3.1.

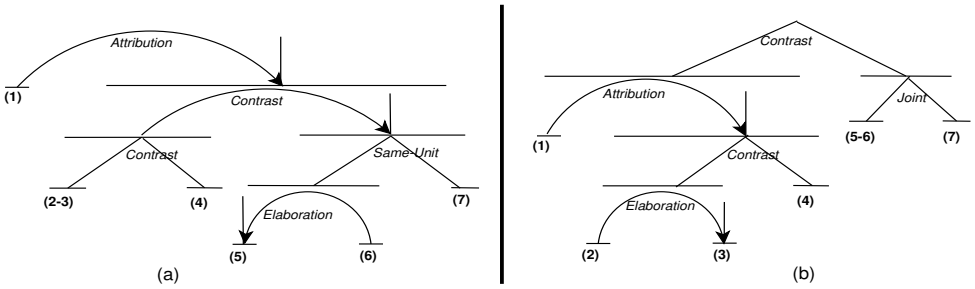



**Figure 3.14:** Measuring the accuracy of a rhetorical parser. (a) The human-annotated discourse tree. (b) The system-generated discourse tree.

the break between EDUs 5 and 6. In table 3.2, we list all constituents in the two DTs and their associated labels at the span, nuclei and relation levels. The recall (R) and precision (P) figures are shown at the bottom of the table. Note that following [126], the relation labels are assigned to the children nodes rather than to the parent nodes in the evaluation process to deal with non-binary trees in human annotations. From this discussion, this is easy to understand that if the number of EDUs are the same in the human and system annotations, and the discourse trees are binary, then we get the same figures for precision, recall and $F_1$-score.

### 3.6.3 Discourse Segmentation Evaluation

In this section we present our experiments on discourse segmentation.

| | Spans | | Nuclearity | | Relations | |
|---|---|---|---|---|---|---|
| Constituents | Human | System | Human | System | Human | System |
| 1-1 | * | * | S | S | Attribution | Attribution |
| 2-2 | | * | | S | | Elaboration |
| 3-3 | | * | | N | | Span |
| 4-4 | * | * | N | N | Contrast | Contrast |
| 5-5 | * | | N | | Span | |
| 6-6 | * | | S | | Elaboration | |
| 7-7 | * | * | N | N | Same-Unit | Joint |
| 2-3 | * | * | N | N | Contrast | Contrast |
| 5-6 | * | * | N | N | Same-Unit | Joint |
| 2-4 | * | * | S | N | Contrast | Span |
| 5-7 | * | * | N | N | Contrast | Contrast |
| 1-4 | | * | | N | | Contrast |
| 2-7 | * | | N | | Span | |
| | R = 7/10, P = 7/10 | | R = 6/10, P = 6/10 | | R = 4/10, P = 4/10 | |

**Table 3.2:** Measuring parsing accuracy (P = Precision, R = Recall).

**Experimental Setup for Discourse Segmentation**

We compare the performance of our segmenter with the performance of the two publicly available segmenters, namely the segmenters of HILDA [84] and SPADE [191]. We also compare our results with the state-of-the-art results reported by Fisher and Roark [72] on the RST-DT test set. We ran HILDA with its default settings. For SPADE, we applied the same modifications to its default settings as described in [72], which delivers significantly improved performance over its original version. Specifically, in our experiments on RST-DT, we trained SPADE using the human-annotated syntactic trees extracted from the Penn Treebank, and during testing, we replaced the Charniak parser [42] with a more accurate reranking parser [43]. However, due to the lack of gold syntactic trees in the Instructional corpus, we trained SPADE in this corpus using the syntactic trees produced by the reranking parser. To avoid using the gold syntactic trees, we used the reranking parser in all our systems for both training and testing purposes. This syntactic parser was trained on the sections of the Penn Treebank not included in our test set.

We applied the same canonical lexical head projection rules [47, 120] to lexicalize the syntactic trees as done in HILDA and SPADE.

Note that previous studies [72, 84, 191] on discourse segmentation only report their performance on the RST-DT test set. To compare our results with them, we evaluate our model on the RST-DT test set. In addition, we show a more general performance of SPADE and our system on the two corpora based on 10-fold cross validation. However, SPADE does not come with a training module for its segmenter. We reimplemented this module and verified it on the RST-DT test set.

**Results for Discourse Segmentation**

Table 3.3 shows the segmentation results of different systems in Precision, Recall, and $F_1$-score on the two corpora. On RST-DT, HILDA's segmenter delivers the weakest performance having a $F_1$-score of only 74.1.[10] SPADE performs much better than HILDA with an absolute $F_1$-score improvement of 11.1%. Our segmenter LR outperforms SPADE with an absolute $F_1$-score improvement of 4.9% (p $<$ 2.4e-06), and also achieves comparable results to the results of Fisher and Roark [72] (F&R), even though we use fewer features.[11] Notice that human agreement for this task is quite high, i.e., a $F_1$-score of 98.3 computed on the doubly-annotated portion of the RST-DT corpus mentioned in Section 3.6.1.

| | RST-DT | | | | | | | Instructional | |
|---|---|---|---|---|---|---|---|---|---|
| | Standard Test Set | | | | Doubly | 10-fold | | 10-fold | 10-fold |
| | HILDA | SPADE | F&R | LR | Human | SPADE | LR | SPADE | LR |
| Precision | 77.9 | 83.8 | **91.3*** | 88.0* | 98.5 | 83.7 | **87.5*** | 65.1 | **73.9*** |
| Recall | 70.6 | 86.8 | 89.7* | **92.3*** | 98.2 | 86.2 | **89.9*** | 82.8 | **89.7*** |
| $F_1$-score | 74.1 | 85.2 | **90.5*** | 90.1* | 98.3 | 84.9 | **88.7*** | 72.8 | **80.9*** |

**Table 3.3:** Segmentation results of different models on the two corpora. Performances significantly superior to SPADE are denoted by *.

Since Fisher and Roark [72] only report their results on the RST-DT test set and we did not have access to their system, we compare our approach with only SPADE when evaluating on a whole corpus based on 10-fold cross validation. On RST-DT,

[10] The high segmentation accuracy reported in [84] is due to a less stringent evaluation metric.

[11] Since we did not have access to the system or to the complete output/results of Fisher and Roark [72], we were not able to perform a significance test.

our segmenter delivers an absolute $F_1$-score improvement of 3.8%, which represents a more than 25% relative error rate reduction. The improvement is higher on the Instructional corpus with an absolute $F_1$-score improvement of 8.1%, which corresponds to a relative error reduction of 30%. The improvements for both corpora are statistically significant ($p < 3.0e\text{-}06$). When we compare our results on the two corpora, we observe a substantial decrease in performance on the Instructional corpus. This could be due to a smaller amount of data in this corpus and the inaccuracies in the syntactic parser and taggers, which are trained on news articles. A promising future direction would be to apply effective domain adaptation methods (e.g., `easyadapt` [53]) to improve segmentation performance in the Instructional domain by leveraging the rich data in the news domain (i.e., RST-DT).

### 3.6.4 Discourse Parsing Evaluation

This section presents our experiments on discourse parsing. First, we describe the experimental setup. Then, we present the results. While presenting the performance of our discourse parser, we show a breakdown of intra-sentential vs. inter-sentential results, in addition to the aggregated results at the document level.

#### Experimental Setup for Discourse Parsing

In our experiments on sentence-level (i.e., intra-sentential) parsing, we compare our approach with SPADE [191] on RST-DT, and with the ILP-based approach of Subba and Di-Eugenio [201] on the Instructional corpus, since they are the state-of-the-art on the respective genres. For SPADE, we applied the same modifications to its default settings as described in Section 3.6.3, which leads to improved performance. Similarly, in our experiments on document-level (i.e., multi-sentential) parsing, we compare our approach with HILDA [84] on RST-DT, and with the ILP-based approach [201] on the Instructional corpus. The results for HILDA were obtained by running the system with default settings on the same inputs we provided to our system. Since we could not run the ILP-based system (not publicly available), we report the performance presented in their paper.

Our experiments on RST-DT use the 18 coarser relations (see Figure 3.15) defined by Carlson and Marcu [37] and also used in SPADE and HILDA. After at-

taching the nuclearity statuses (NS, SN, NN) to these relations, we get 41 distinct relations.[12] Our experiments on the Instructional corpus consider the same 26 primary relations (e.g., *Goal:Act*, *Cause:Effect*) used by Subba and Di-Eugenio [201] and also treat the reversals of non-commutative relations as separate relations. That is, *Goal-Act* and *Act-Goal* are considered as two different relations. Attaching the nuclearity statuses to these relations gives 76 distinct relations.

**Evaluation of the Intra-sentential Parser**

This section presents our experimental evaluation on intra-sentential parsing. First, we show the performance of the sentence-level discourse parsers when they are provided with manual (or gold) segmentations. This allows us to judge the parsing performance independently of the discourse segmentation task. Then, we show the end-to-end performance, i.e., the performance based on automatic segmentation.

*Intra-sentential parsing results based on manual segmentation*

Table 3.4 presents the intra-sentential parsing results when manual segmentation is used.[13] Notice that our sentence-level parser DCRF consistently outperforms SPADE on the RST-DT test set in all three metrics, and the improvements are statistically significant ($p < 0.01$). Especially, on the relation labeling task, which is the hardest among the three tasks, we get an absolute $F_1$-score improvement of 9.6%, which represents a relative error rate reduction of 29.6%. Our $F_1$-score of 77.1 in relation labeling is also close to the human agreement of 83.0 on the doubly-annotated data. Our results on the RST-DT test set are also consistent with the mean scores over 10-folds on the whole RST-DT corpus.

The improvements are higher on the Instructional corpus, where we compare our mean results over 10-folds with the reported results of the ILP-based system [201], giving absolute $F_1$-score improvements of 4.8%, 15.5% and 10.6% in span, nuclearity and relations, respectively.[14] In other words, our parser reduces the

[12]Not all relations take all the possible nuclearity statuses. For example, *Elaboration* and *Attribution* are mono-nuclear relations, and *Same-Unit* and *Joint* are multi-nuclear relations.

[13]Recall from the discussion in Section 3.6.2 that precision, recall and $F_1$-score are the same when manual segmentation is used.

[14]Subba and Di-Eugenio [201] report their results based on an arbitrary split between training and test sets. Since we did not have access to their particular split, we compare our model's performance

| | RST-DT | | | | Instructional | |
|---|---|---|---|---|---|---|
| | Standard Test Set | | 10-fold | Doubly | Reported | 10-fold |
| Scores | SPADE | DCRF | DCRF | Human | ILP | DCRF |
| Span | 93.5 | **96.5*** | 95.3 | 95.7 | 92.9 | **98.3** |
| Nuclearity | 85.8 | **89.4*** | 88.2 | 90.4 | 71.8 | **89.2** |
| Relation | 67.6 | **79.8*** | 77.7 | 83.0 | 63.0 | **75.6** |

**Table 3.4:** Intra-sentential parsing results based on manual segmentation. Performances significantly superior to SPADE are denoted by *.

errors by 67.6%, 54.6% and 28.6% in span, nuclearity and relations, respectively.

If we compare the performance of our DCRF parser on the two corpora, we notice that our parser is more accurate in finding the right tree structure (see *Span* row in the table) on the Instructional corpus. This may be due to the fact that sentences in the Instructional domain are relatively short and contain fewer EDUs than sentences in the news domain, thus making it easier to find the right tree structure. However, when we compare the performance on the relation labeling task, we observe a decrease on the Instructional corpus. This may be due to the small amount of data available for training and the imbalanced distribution of a large number of discourse relations (i.e., 76 with nuclearity attached) in this corpus.

*Intra-sentential parsing results based on automatic segmentation*

In order to evaluate the performance of the end-to-end sentence-level discourse analysis systems, we feed the intra-sentential discourse parsers the output of their respective segmenters. Table 3.5 shows the (P)recision, (R)ecall, and ($F_1$)-score results for different metrics. We compare our results with SPADE on the RST-DT test set. We achieve absolute $F_1$-score improvements of 3.6%, 3.4% and 7.4% in span, nuclearity and relation, respectively. These improvements are statistically significant ($p<0.001$). Our system, therefore, reduces the errors by 15.5%, 11.4%, and 17.6% in span, nuclearity and relations, respectively. These results are also consistent with the mean results over 10-folds on the whole RST-DT corpus.

based on 10-fold cross validation with their reported results. Also, since we did not have access to their system/output, we could not perform a significance test on the Instructional corpus.

| | RST-DT | | | | | | | | | Instructional | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Test set | | | | | | 10-fold | | | 10-fold | | |
| Scores | SPADE | | | DCRF | | | DCRF | | | DCRF | | |
| | P | R | $F_1$ | P | R | $F_1$ | P | R | $F_1$ | P | R | $F_1$ |
| 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 |
| Nuc. | 69.8 | 70.5 | 70.2 | 75.2* | 78.1* | 76.6* | 73.9 | 76.5 | 75.2 | 64.6 | 71.0 | 67.6 |
| Rel. | 57.4 | 58.5 | 58.0 | 66.1* | 68.8* | 67.5* | 65.2 | 67.4 | 66.3 | 54.8 | 60.4 | 57.5 |

**Table 3.5:** Parsing results using automatic segmentation. Performances significantly superior to SPADE are denoted by *.

The rightmost column in the table shows our mean results over 10-folds on the Instructional corpus. We could not compare with the ILP-based approach [201] because no results were reported using an automatic segmenter. It is interesting to observe how much our end-to-end system is affected by an automatic segmenter on both RST-DT and the Instructional corpus (see Table 3.4 and Table 3.5). Nevertheless, taking into account the segmentation results in Table 3.3, this is not surprising because previous studies [191] have already shown that automatic segmentation is the primary impediment to high accuracy discourse parsing. This demonstrates the need for a more accurate segmentation model in the Instructional genre.

### Evaluation of the Complete Parser

We experiment with our full document-level discourse parser on the two corpora using the two parsing approaches described in Section 3.4.3, namely 1S-1S and the sliding window. On RST-DT, the standard split was used for training and testing. On the Instructional corpus, Subba and Di-Eugenio [201] used 151 documents for training and 25 documents for testing. Since we did not have access to their particular split, we took 5 random samples of 151 documents for training and 25 documents for testing, and report the average performance over the 5 test sets.

Table 3.6 presents $F_1$-scores for our parsers and the existing systems on the two corpora based on manual segmentation.[15] On both corpora, our parsers, namely, 1S-1S (TSP 1-1) and the sliding window (TSP SW), outperform existing systems by a wide margin ($p<7.1e-05$ on RST-DT).[16] On RST-DT, our parsers achieve ab-

[15] Recall that Precision, Recall and $F_1$-score are the same when manual segmentation is used.

[16] Since we did not have access to the output or to the system of Subba and Di-Eugenio [201], we

solute $F_1$-score improvements of 8%, 9.4% and 11.4% in span, nuclearity and relation, respectively, over HILDA. This represents relative error reductions of 32%, 23% and 21% in span, nuclearity and relation, respectively. Our results are also close to the upper bound, i.e. human agreement on this data set.

On the Instructional genre, our parsers deliver absolute $F_1$-score improvements of 10.5%, 13.6% and 8.14% in span, nuclearity and relations, respectively, over the ILP-based approach. Our parsers, therefore, reduce errors by 36%, 27% and 13% in span, nuclearity and relations, respectively.

| | RST-DT | | | | Instructional | | |
|---|---|---|---|---|---|---|---|
| Metrics | HILDA | TSP 1-1 | TSP SW | Human | ILP | TSP 1-1 | TSP SW |
| Span | 74.68 | **82.56*** | **82.84***† | 88.70 | 70.35 | **80.67** | **81.88**† |
| Nuclearity | 58.99 | **68.32*** | **68.30*** | 77.72 | 49.47 | **63.03** | **63.13** |
| Relation | 44.32 | **55.83*** | **55.81*** | 65.75 | 35.44 | **43.52** | **43.60** |

**Table 3.6:** Parsing results of different document-level parsers using manual (gold) segmentation. Performances significantly superior to HILDA (with p<7.1e-05) are denoted by *. Significant differences between TSP 1-1 and TSP SW (with p<0.01) are denoted by †.

If we compare the performance of our parsers on the two corpora, we observe higher results on RST-DT. This can be explained in at least two ways. First, the Instructional corpus has a smaller amount of data with a larger set of relations (76 when nuclearity attached). Second, some frequent relations are (semantically) very similar (e.g., Preparation-Act, Step1-Step2), which makes it difficult even for the human annotators to distinguish them [201].

Comparison between our two models reveals that TSP SW significantly outperforms TSP 1-1 only in finding the right structure on both corpora ($p$<0.01). Not surprisingly, the improvement is higher on the Instructional corpus. A likely explanation is that the Instructional corpus contains more leaky boundaries (12%), allowing the sliding window approach to be more effective in finding those, without inducing much noise for the labels. This clearly demonstrates the potential of TSP SW for datasets with even more leaky boundaries e.g., the Dutch [218] and the German Potsdam [197] corpora.

were not able to perform a significance test on the Instructional corpus.

Error analysis reveals that although TSP SW finds more correct structures, a corresponding improvement in labeling relations is not present because in a few cases, it tends to induce noise from the neighboring sentences for the labels. For example, when parsing was performed on the first sentence in Figure 3.1 in isolation using 1S-1S, our parser rightly identifies the *Contrast* relation between EDUs 2 and 3. But, when it is considered with its neighboring sentences by the sliding window, the parser labels it as *Elaboration*. A promising strategy to deal with this and similar problems that we plan to explore in future, is to apply both approaches to each sentence and combine them by consolidating three probabilistic decisions, i.e. the one from 1S-1S and the two from the sliding window.

*Analysis on the importance of the features*

To analyze the importance of the features, Table 3.7 presents the intra-sentential and multi-sentential parsing results based on manual segmentation on the RST-DT test set using different subsets of features. Every new subset of features appears to improve the performance. Specifically, for intra-sentential parsing, when we add the *Organizational* features with the *Dominance set* features (see $I_2$ column), we get about 2% absolute improvements in nuclearity and relations. With *N-gram* features, the gain is even higher; 6% in relations and 3.5% in nuclearity for intra-sentential parsing (see $I_3$), and 3.8% in relations and 3.1% in nuclearity for multi-sentential parsing (see $M_2$). This demonstrates the utility of the N-gram features, which is also consistent with the findings of [60, 181]. The features extracted from *Lexical chains* have also proved to be useful for multi-sentential parsing. They deliver absolute improvements of 2.7%, 2.9% and 2.3% in span, nuclearity and relations, respectively (see $M_3$). Including the *Contextual* features further gives improvements of 3% in nuclearity and 2.2% in relation for intra-sentential parsing (see $I_4$), and 1.3% in nuclearity and 1.2% in relation for multi-sentential parsing (see $M_4$). Notice that *Sub-structural* features are more beneficial for document-level parsing than they are for sentence-level parsing, i.e., an improvement of 2.2% vs. an improvement of 0.9%. This is not surprising because in general document-level discourse trees are much larger than sentence-level trees, making the sub-structural features more effective for document-level parsing.

| | Intra-sentential | | | | | Multi-sentential (TSP 1-1) | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| Scores | I1 | I2 | I3 | I4 | I5 | M1 | M2 | M3 | M4 | M5 |
| Span | 91.3 | 92.1 | 93.3 | 94.6 | 96.5 | 74.2 | 75.8 | 78.5 | 80.9 | 82.6 |
| Nuclearity | 78.2 | 80.3 | 83.8 | 86.8 | 89.4 | 60.6 | 63.7 | 65.6 | 66.9 | 68.3 |
| Relation | 66.2 | 68.1 | 74.1 | 76.3 | 79.8 | 46.3 | 50.1 | 52.4 | 53.6 | 55.8 |

**Table 3.7:** Parsing results using different subsets of features on RST-DT test set. Feature subsets for Intra-sentential parsing: I1 = {Dominance set}, I2 = {Dominance set, Organizational}, I3 = {Dominance set, Organizational, N-gram}, I4 = {Dominance set, Organizational, N-gram, Contextual}, I5 (all) = {Dominance set, Organizational, N-gram, Contextual, Sub-structural}. Feature subsets for Multi-sentential parsing: M1 = {Organizational, Text structural}, M2 = {Organizational, Text structural, N-gram}, M3 = {Organizational, Text structural, N-gram, Lexical chain}, M4 = {Organizational, Text structural, N-gram, Lexical chain, Contextual}, M5 (all) = {Organizational, Text structural, N-gram, Lexical chain, Contextual, Sub-structural}.

### Error Analysis on Relation Labeling and Future Directions

To further analyze the errors made by our parser on the hardest task of relation labeling, Figure 3.15 presents the confusion matrix for TSP 1-1 on the RST-DT test set. The relation labels are ordered according to their frequency in the RST-DT training set. In general, the errors are produced by two different causes acting together: (i) imbalanced distribution of the relations, and (ii) semantic similarity between the relations. The most frequent relation *Elaboration* tends to overshadow others, especially the ones which are semantically similar (e.g., *Explanation*, *Background*) and less frequent (e.g., *Summary*, *Evaluation*). Models sometimes fail to distinguish relations that are semantically similar (e.g., *Temporal:Background*, *Cause:Explanation*).

These observations suggest two ways to improve our parser. We would like to employ a more robust method (e.g., *ensemble* methods with *bagging*) to deal with the imbalanced distribution of relations, along with taking advantage of richer semantic knowledge (e.g., compositional semantics) to cope with the errors caused by semantic similarity between the relations.

In the future, we plan to investigate to what extent discourse segmentation and

| | T-C | T-O | T-CM | M-M | CMP | EV | SU | CND | EN | CA | TE | EX | BA | CO | JO | S-U | AT | EL |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| T-C | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |
| T-O | 0 | 2 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| T-CM | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 2 | 0 | 0 | 7 |
| M-M | 0 | 0 | 0 | 10 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 1 | 3 |
| CMP | 0 | 0 | 0 | 1 | 4 | 0 | 0 | 1 | 0 | 1 | 0 | 3 | 3 | 0 | 1 | 1 | 0 | 2 |
| EV | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 2 | 0 | 0 | 2 | 0 | 2 | 11 |
| SU | 0 | 0 | 0 | 0 | 0 | 0 | 8 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 12 |
| CND | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 22 | 0 | 0 | 0 | 0 | 1 | 3 | 0 | 0 | 3 | 2 |
| EN | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 24 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 7 |
| CA | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 2 | 3 | 0 | 4 | 2 | 2 | 7 | 0 | 3 | 11 |
| TE | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 2 | 0 | 7 | 1 | 9 | 1 | 9 | 0 | 3 | 4 |
| EX | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 1 | 5 | 0 | 12 | 0 | 1 | 3 | 0 | 3 | 12 |
| BA | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 0 | 1 | 4 | 1 | 19 | 2 | 6 | 1 | 5 | 12 |
| CO | 0 | 0 | 0 | 1 | 2 | 0 | 0 | 2 | 0 | 1 | 3 | 2 | 2 | 33 | 7 | 0 | 0 | 9 |
| JO | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 2 | 0 | 1 | 1 | 1 | 1 | 2 | 57 | 1 | 0 | 13 |
| S-U | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 85 | 1 | 0 |
| AT | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 3 | 272 | 9 |
| EL | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 14 | 6 | 1 | 8 | 1 | 0 | 8 | 2 | 2 | 359 |

**Figure 3.15:** Confusion matrix for relation labels on the RST-DT test set. Y-axis represents *true* and X-axis represents *predicted* relations. The relations are Topic-Change (T-C), Topic-Comment (T-CM), TextualOrganization (T-O), Manner-Means (M-M), Comparison (CMP), Evaluation (EV), Summary (SU), Condition (CND), Enablement (EN), Cause (CA), Temporal (TE), Explanation (EX), Background (BA), Contrast (CO), Joint (JO), Same-Unit (S-U), Attribution (AT) and Elaboration (EL).

discourse parsing can be jointly performed. We would also like to explore how our system performs on other genres like conversational (e.g., blogs, emails) and evaluative (e.g., customer reviews) texts. To address the problem of limited annotated data in various genres, we are planning to develop an online version of our system that will allow users to fix the parser's output and let the model learn from that feedback. A longer term goal is to extend our framework to also work with graph structures of discourse, as recommended by several recent discourse theories [230]. Once we achieve similar performance on graph structures, we will perform extrinsic evaluations to determine their relative utility for various NLP tasks.

## 3.7 Conclusion

In this chapter, we have presented a complete probabilistic discriminative framework for performing rhetorical analysis in the RST framework. Our discourse segmenter is a binary classifier based on a maximum entropy model. Our discourse parser applies an optimal parsing algorithm to probabilities inferred from two CRF models: one for intra-sentential parsing and the other for multi-sentential parsing. The CRF models effectively represent the structure and the label of discourse tree constituents jointly. Furthermore, the DCRF model for intra-sentential parsing captures the sequential dependencies between the constituents. The two separate models (i.e., one for intra-sentential parsing and the other for multi-sentential parsing) use their own informative feature sets and the distributional variations of the relations in the two parsing conditions.

We have also presented two novel approaches to effectively combine the intra-sentential and the multi-sentential parsing modules, that can exploit the strong correlation observed between the text structure and the structure of the discourse tree. The first approach 1S-1S builds a DT for every sentence using the intra-sentential parser, and then runs the multi-sentential parser on the resulting sentence-level DTs. On the other hand, to deal with leaky boundaries, our second approach builds sentence-level sub-trees by applying the intra-sentential parser on a sliding window covering two adjacent sentences and then consolidating the results produced by overlapping windows. After that, the multi-sentential parser takes all these sentence-level sub-trees and builds a full rhetorical parse for the whole document.

Empirical evaluations on two different genres demonstrate that our approach to discourse segmentation achieves state-of-the-art performance more efficiently using fewer features. A series of experiments on the discourse parsing task shows that both our intra-sentential and multi-sentential parsers significantly outperform the state-of-the-art, often by a wide margin. A comparison between our combination strategies reveals that the sliding window approach is more robust across domains.

An error analysis informs us that although the sliding window approach finds more correct tree structures, it tends to induce noise for the relation labels from the neighboring sentences in a few cases. Another analysis of the performance of our discourse parser on the relation labeling task tells us that the relations that are very

frequent tend to mislead the identification of the less frequent ones, and the models sometimes fail to distinguish relations that are semantically similar.

# Chapter 4

# Dialog Act Recognition

In addition to the coarse-grained topical structure discussed in Chapter 2 and the fine-grained rhetorical structure discussed in Chapter 3, asynchronous conversations exhibit another form of discourse structure, which comprises the dialog acts and the conversational structure. The Fragment Quotation Graph (FQG) discussed in Chapter 2 provides a fine-grain conversational structure of an asynchronous conversation by linking the text fragments in the messages based on their reply-to relations. In this chapter, we study dialog act recognition, which aims to identify the communicative acts (e.g., *question, request*) performed by the participants in the course of the conversation. We present three unsupervised approaches: a graph-theoretic deterministic framework and two probabilistic conversational models (namely HMM and HMM+Mix) for modeling dialog acts in asynchronous conversations. The deterministic models do not consider sequential dependencies between the act types, while the probabilistic models do. First we show that capturing sequential dependencies between the act types is important in asynchronous conversations as it is in synchronous domains (e.g., meetings, phone conversations). Then we demonstrate that the probabilistic models learn better sequential dependencies when they are trained on the sequences extracted from the conversational structure, rather than when they are trained on the sequences based on the temporal order. A comparison between the probabilistic models confirms that HMM+Mix is a better conversational model than the simple HMM model.[1]

[1]This chapter is based on the peer-reviewed conference paper Joty et al. [95] (**IJCAI-2011**).

## 4.1 Introduction

What makes an asynchronous conversation different from a monolog? Although participants communicate in writing (as opposed to speech), the nature of the interaction in asynchronous media is conversational, in the sense that once a participant initiated the communication all the other contributions are replies to previous ones. That is, the participants take turns, each of which consists of one or more utterances. The utterances in a turn perform certain communicative actions (e.g., asking a question, answering a question, requesting something, offering an apology, greeting), which are called **dialog acts**[2] (DAs) [12]. For instance, in the last email shown in Figure 4.1, the sentence *Yes - I could ...* answers the question posed in the third email. Two-part structures connecting two DAs (e.g., *Question-Answer, Request-Accept*) are called **adjacency pairs** [180].

Uncovering the complex dialog structure of an asynchronous conversation is an important step towards deep conversational analysis. Annotating utterances with DAs as shown in Figure 4.1 provides an initial level of structure — that has been shown to be useful for many applications in spoken dialog including meeting summarization [148, 151], collaborative task learning agents [5], artificial companions for people to use the Internet [229] and flirtation detection in speed-dates [170]. We believe that similar benefits will also hold for written asynchronous conversations.

While considerable progress has been made in DA recognition for synchronous conversations (e.g., [102, 232] for chats, [57, 107] for meetings, [14, 171, 200] for phone conversations), very little work has been conducted in asynchronous domains, especially at the sentence level. The dominant approaches to DA recognition in synchronous domains have been mostly supervised, and use either simple classifiers (binary or multi-class) or more structured models like Hidden Markov Models (HMMs), Maximum Entropy Markov Models (MEMMs), and Conditional Random Fields (CRFs). Since turns in synchronous conversations occur one after the other with minimal delay, the conversation flow in these conversations exhibits sequential dependencies between adjacency pairs (e.g., *question* followed by *answer*, *request* followed by *grant*). Sequence labelers like HMMs, MEMMs and CRFs, which are capable of capturing these inter-dependencies between the dialog

[2] Also known as *speech acts*.

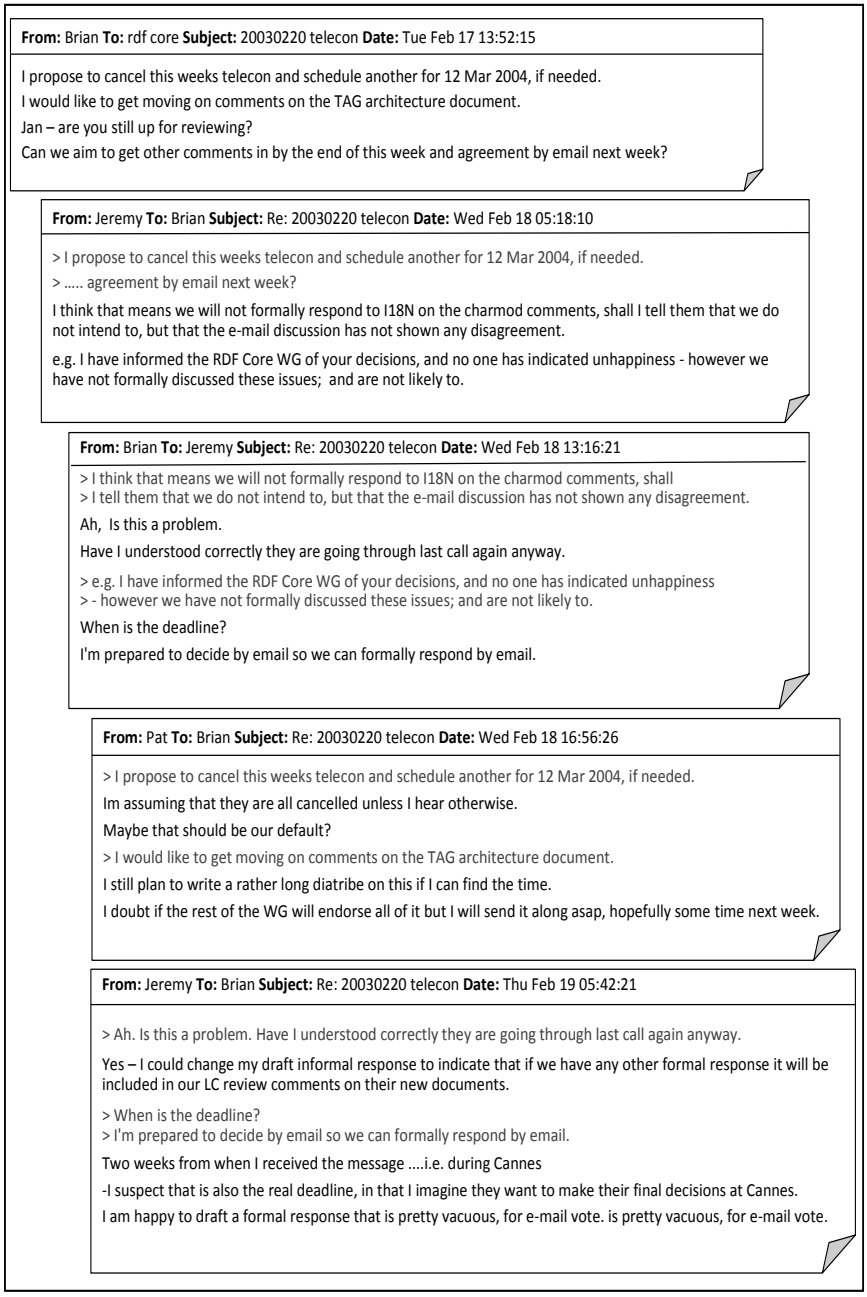



**Figure 4.1:** Sample truncated email conversation from our BC3 corpus. The right most column (i.e., DA) specifies the dialog act assignments for the sentences. The DA tags are defined in Table 4.1. The Fragment column specifies the fragments in the Fragment Quotation Graph (FQG) (see Figure 4.2).

acts, generally perform better than the simple classifiers (e.g., MaxEnt, SVMs).

However, the supervised learning strategy is very domain specific and requires considerable labeled data which can be labor intensive and expensive to acquire. Arguably, as the number of social media grows (e.g., email, blogs, Facebook) and the number of communicative settings in which people interact through these media also grows, the supervised paradigm of 'label-train-test' becomes too expensive and unrealistic. Every novel use of a new media may require not only new annotations, but possibly also new annotation guidelines and new DA tagsets. In contrast, the approach we present in this thesis adopts an unsupervised paradigm, where DA recognition is considered as a two step process: first, clustering the sentences based on their DA types, and then, assigning an appropriate DA label to each cluster. This approach is more robust across new forms of media and new domains. In this chapter, we investigate a graph-theoretic deterministic framework and two probabilistic conversational models for clustering sentences based on their DAs. The DA label for each cluster needs to be then determined through other means, which we do not explore in this study, but may include minimal supervision.

The **graph-theoretic framework**, used previously for topic segmentation in Chapter 2, clusters sentences into DAs based on their lexical and structural similarity, but ignores sequential dependencies between the DA types. The performance of this model crucially depends on how one measures the similarity between two sentences. We experimented with a wide range of similarity metrics, including TF.IDF-based cosine similarity, word subsequence kernel [32], extended string subsequence kernel [85] with part of speech (POS) tags, basic element-based dependency similarity [88] and tree kernel-based deep syntactic similarity [48].

Quite differently, the **probabilistic conversational models** frame DA clustering as a sequence-labeling problem that can be solved by variations of HMMs with the assumption that a conversation is a sequence of hidden DAs and each DA emits an observed sentence. The performance of a probabilistic model depends on the accuracies of the state transition distributions (i.e., sequence dependencies between DAs) and the act-emission (or observation) distributions. While the idea of using probabilistic models for sequence labeling to perform DA tagging is not new, we make several key contributions in showing how it can be effectively applied to asynchronous conversations by dealing with critical limitations in previous work.

Unlike synchronous conversations, the conversational flow in asynchronous conversations often lacks sequential dependencies between the act types in its temporal order. For example, consider the email conversation shown in Figure 4.1. If we arrange the sentences (excluding the quoted sentences) as they arrive in the conversation, it becomes hard to capture the sequential dependencies between the act types because the two components of the adjacency pairs can be far apart in the sequence. This could lead to inaccurate sequential dependencies in the sequence labelers, when they are applied to this sequence of the sentences. This example also demonstrates that the use of quotations (see the last email) can help us in putting the components of adjacency pairs close to each other. Therefore, we hypothesize that the sequences extracted from the conversational structure of the asynchronous conversation, e.g., the Fragment Quotation Graph (FQG), are rather more effective to accurately learn the sequence dependencies between the DA tags. That is, two neighboring sentences in the conversational structure are likely to be related (expressing related DAs), independently from their arrival time-stamp.

Among recent attempts on unsupervised DA modeling, Ritter et al. [174] propose HMMs to cluster the tweets in a Twitter conversation based on their dialog acts. They use a unigram language model as the act-emission distribution. However, other features like *speaker*, *relative position* and *sentence length* have also proved to be beneficial in supervised settings [93, 102]. In this work, we model each observation (sentence) not only by its unigrams but also by its speaker, relative position and length. Another crucial finding of Ritter et al. [174] is that their simple HMM conversational model tends to find some undesirable topical clusters in addition to the DA clusters. Without any supervision, distinguishing DAs from topics is in fact a common challenge, because many of features used for modeling DAs are also indicators of topics. Ritter et al. [174] address this problem by proposing an HMM+Topic model, which tries to separate the topic words from the DA indicators. In this work, we propose a more adequate HMM+Mix model, which not only explains away the topics, but also improves the act-emission distribution by defining it as a mixture model. We present the Expectation Maximization (EM) derivation to learn the parameters of this model.

We evaluate our models on two different datasets: email and forum. In what is to the best of our knowledge the first quantitative evaluation of unsupervised

DA tagging for asynchronous conversations, we show that (i) the graph-theoretic framework is not the right model for this task, (ii) the conversational models learn better sequential dependencies when they are trained on sequences extracted from the conversational structure rather than when they are trained on sequences based on the temporal order of the sentences and (iii) HMM+Mix is a better conversational model than the simple HMM model.

The rest of the chapter is organized as follows. After discussing related work in Section 4.2, we present our corpora in Section 4.3. In Section 4.4 we present the graph-theoretic framework and its performance on clustering sentences based on their DAs. We present our probabilistic conversational models in Section 4.5 and their evaluation in Section 4.6. Finally, we conclude with future directions in Section 4.7.

## 4.2 Related Work

There has been little work on DA recognition in asynchronous conversation. The approaches can be broadly classified into supervised, semi-supervised and unsupervised methods. Cohen et al. [46] first use the term *email speech act* for classifying *emails* based on their acts (e.g., deliver, meeting). However, their classifiers do not capture the sequential dependencies between the act types. In their follow-up work, Carvalho and Cohen [39] address this limitation by using two different classifiers – one for content and one for context – in an iterative collective classification algorithm. The content classifier only looks at the content of the message, whereas the context classifier takes into account both the content of the message and the dialog act labels of its parent and children in the thread structure of the email conversation. Note that their inventory of dialog acts is very specific to a work environment, and their approaches operate at the email level, not at the sentence level. Identification of adjacency pairs like *question-answer* pairs in email discussions using supervised learning methods was investigated in [173, 186]. Ferschke et al. [70] use DAs to analyze the collaborative process of editing Wiki pages, and apply supervised models to identify the DAs in Wikipedia Talk pages.

Several semi-supervised methods have been proposed for DA recognition in asynchronous conversation. Jeong et al. [93] use semi-supervised boosting to tag

the sentences in email and forum discussions with DAs by adapting knowledge from annotated spoken conversations (i.e., MRDA-tagged meeting and DAMSL-tagged telephone conversations). Given a sentence represented as a set of trees (i.e., dependency tree, n-gram tree and POS tag tree), the boosting algorithm iteratively learns the best feature set (i.e., sub-trees) that minimizes the errors in the training data. Note that this approach of adapting knowledge from synchronous domains to asynchronous domains could be problematic because domain adaptation methods generally work better when the distance between the source and the target domains is minimal. Another crucial limitation is that they do not consider the sequential dependencies between the act types, something we successfully exploit in our work. Zhang et al. [235] also employ semi-supervised methods for DA recognition in twitter. They use a transductive SVM and a graph-based label propagation framework to leverage the knowledge from abundant unlabeled data.

Among recent research on unsupervised DA modeling, Ritter et al. [174] propose two HMM-based unsupervised conversational models for modeling DAs in twitter. In particular, they use a simple HMM model and a HMM+Topic model to cluster the Twitter posts (not the sentences) into DAs. Since, they use a unigram language model to define the emission distribution, their simple HMM model tends to find some topical clusters in addition to the clusters that are based on DAs. The HMM+Topic model tries to separate the DA indicators from the topic words. By visualizing the type of conversations found by the two models, they show that the output of the HMM+Topic model is more interpretable than that of the HMM one, however, their classification accuracy is not empirically evaluated. Therefore, it is not clear whether these models are actually useful (i.e., beat the baseline), and which of the two models is a better DA tagger. Recently, Paul [160] proposes to use a mixed membership Markov model to cluster sentences based on their DAs, and show that this model outperforms a simple HMM.

Our conversation models were inspired by the models of Ritter et al. [174], but we improve on those by making the following four key contributions: (i) we model at the finer level of granularity (i.e., at the sentence-level as opposed to post-level) with a richer feature set including not only unigram but also sentence relative position, sentence length and speaker, (ii) our models exploit the graph structure of the conversations, (iii) our HMM+Mix model not only explains away the topics

(like HMM+Topic does), but also improves the emission distribution by defining it as a mixture model [23], (iv) we provide clustering accuracy of the models on two corpora (email and forum) by applying the one-to-one metric from [63].

## 4.3 Corpora

To show the generality of our methods, we experiment with two different types of asynchronous conversations: *email* and *forum*. Below we describe our datasets.

### 4.3.1 Dataset Selection and Clean Up

We used the same DA tagset and test datasets used by Jeong et al. [93]. The tagset, containing 12 act categories with their relative frequencies in the email and forum (test) corpora, is shown in Table 4.1. This inventory of DAs is originally adopted from the MRDA tagset [55]. We use this tagset because it is domain independent and suitable for sentence-level annotation [93]. Our test datasets include the 40 email conversations from our BC3 corpus [213] and 200 forum conversations from the travel forum site TripAdvisor[3]. As already mentioned in Chapter 2, the BC3 email conversations were originally selected from the W3C mailing list.[4]

Notice that the DA tags have similar distribution in the two corpora. *Statement* is the most frequent tag in both corpora covering about 66% to 70% of the sentences. *Polite mechanism, Yes-no question* and *Action motivator* have frequencies in the range 6% to 9% in the two corpora. Other DA tags have considerably lower frequencies in the two corpora. The $\kappa$ agreements between two human annotators are 0.79 and 0.73 for the email and forum corpora, respectively.

Due to privacy issues, there are only a few corpora of asynchronous conversations available for training an unsupervised system. Since it is preferable to train and test such a system on similar data, we have chosen the W3C email corpus (as opposed to Enron) to train our models on email conversations.[5] W3C contains $23,957$ email conversations. However, the raw data is too noisy (with system messages and signatures) to directly inform our models. We cleaned up the data with

[3] http://tripadvisor.com

[4] http://research.microsoft.com/en-us/um/people/nickcr/w3c-summary.html

[5] In contrast, Jeong et al. [93] train on Enron and test on BC3.

| Tag | Description | Email | Forum |
|---|---|---|---|
| S | Statement | 69.56% | 65.62% |
| P | Polite mechanism | 6.97% | 9.11% |
| QY | Yes-no question | 6.75% | 8.33% |
| AM | Action motivator | 6.09% | 7.71% |
| QW | Wh-question | 2.29% | 4.23% |
| A | Accept response | 2.07% | 1.10% |
| QO | Open-ended question | 1.32% | 0.92% |
| AA | Acknowledge and appreciate | 1.24% | 0.46% |
| QR | Or/or-clause question | 1.10% | 1.16% |
| R | Reject response | 1.06% | 0.64% |
| U | Uncertain response | 0.79% | 0.65% |
| QH | Rhetorical question | 0.75% | 0.08% |

**Table 4.1:** Dialog act tags and their relative frequencies in the two corpora.

the intention to keep only the headers, bodies and quotations. By processing the headers, we then reconstruct the thread structure of the email conversations.

In order to train our models on forum conversations, we crawled $25,000$ forum threads from the same travel forum site i.e., TripAdvisor. Our forum data is less noisy, but does not contain any thread structure (i.e., reply-to relations).

### 4.3.2 Dealing with Conversational Structure

In probabilistic conversational models, sequence dependencies between DA tags can be learned either from the simple temporal order of the utterances in a conversation, or from the sequences extracted from the graph structure of the conversation. We create a temporally ordered conversation by simply arranging its posts based on their arrival time. For the graph structure, we construct the Fragment Quotation Graph (FQG) [35] described in Chapter 2 as a fine-grained conversational structure of email conversations. For example, Figure 4.2 shows the FQG for the email conversation shown in Figure 4.1. Once again, for the sake of illustration, the real contents of the emails are abbreviated as a sequence of labels, each representing a text fragment (see the Fragment column in Figure 4.1). However, since the forum conversations in TripAdvisor do not contain any thread structure,

and participants hardly quote from others' posts in this forum, we could not build FQG for the forum threads. But we noticed that participants in this forum almost always respond to the initial post of the thread, and generally mention other participants' names (as a method of disentanglement) to respond to their post. Therefore, we create the graph structure of a forum conversation with the simple assumption that a post usually responds to the initial post unless it mentions other participants' names. If it mentions other participants' names, we consider the most recent post from those participants to be the post it replies to. One can also employ more sophisticated methods such as the ones described in [11, 220] to uncover the implicit thread structure of a forum conversation, which is beyond the scope of this thesis.

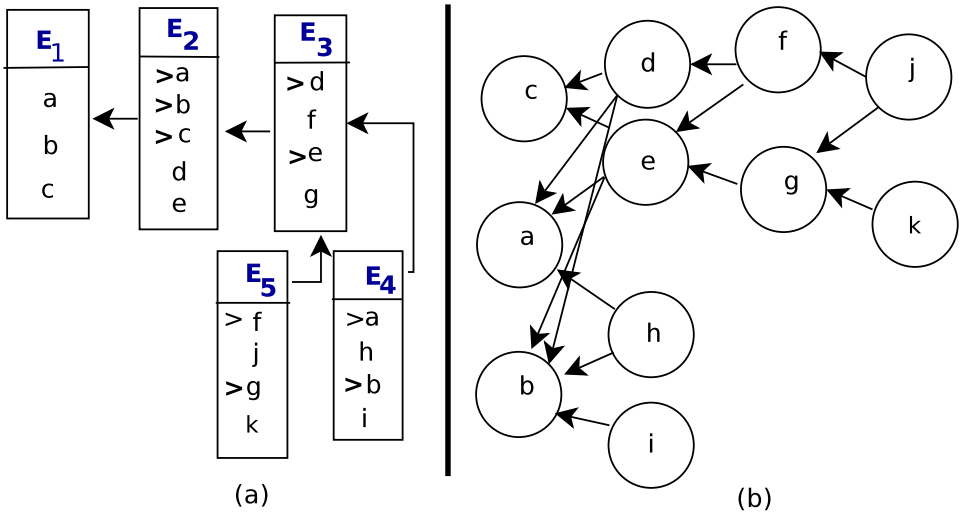



**Figure 4.2:** (a) The email conversation of Figure 4.1 with the fragments. Arrows indicate 'reply-to' relations. (b) The corresponding FQG.

Our assumption is that two neighboring sentences in the graph structure of a conversation are likely to express related DAs. Notice that the paths in the graph (e.g., *c-d-f-j* in Figure 4.2(b)) capture the adjacency relations (e.g., *question-answer, request-grant*) between text fragments. Therefore, we extract the sentences along each of the paths as a sequence, on which we train our probabilistic conversational models. However, by doing this the sentences in the nodes shared by multiple paths (e.g., $c, e, g$) are duplicated in multiple sequences. Section 4.5.4 describes how our conversational models deal with these duplicates.

## 4.4 Graph-theoretic Framework

Our first model for clustering sentences based on their DAs is built on the same graph-theoretic framework used earlier for topic segmentation in Chapter 2. As mentioned before, this framework has been successfully applied to many other NLP tasks including chat disentanglement [63] and coreference resolution [190]. In this work, we investigate whether the same framework can be adapted to clustering sentences of an asynchronous conversation based on their DAs.

### 4.4.1 Algorithm Description

In this framework, first we form a complete similarity graph $G = (V, E)$, where the nodes $V$ represent the sentences in a conversation and the edge-weights represent the *similarity* between the nodes (i.e., for an edge $(u, v) \in E$, edge-weight $w(u, v)$ represents how similar the sentences $u$ and $v$ are). Then we formulate the clustering problem as a **k-way-mincut** graph-partitioning problem with the intuition that sentences in a cluster should be similar to each other, while sentences in different clusters should be dissimilar, which we solve again by optimizing the **normalized cut (Ncut)** criterion (Equation 2.6) described in Section 2.3.1. Note that, depending on the task, the performance of this framework depends on how one measures the similarity between two sentences. In the following, we briefly describe the similarity metrics we experimented with in our work.

Unigrams or bag-of-words (**BOW**) have been quite extensively used in previous work on DA recognition ([102, 174]). To measure the BOW-based similarity between two sentences, we represent each sentence as a vector of TF.IDF [178] values of its words and compute the cosine similarity (Equation 2.5) between the vectors. However, recall from the discussion in Chapter 2 that this model has been quite successful for finding topical clusters [121]. Although we retain the stopwords and punctuation, which are arguably useful for recognizing DAs, it may still find clusters that are based on topics rather than DAs. In an attempt to abstract away the topic words, we also use a variation of the above metric, where we mask (**BOW-M**) the nouns in the sentences and measure the cosine similarity as before.[6]

The BOW and BOW-M similarity metrics do not consider the order of the

[6] Since nouns are arguably the most indicative for topics.

words. One can use n-gram co-occurrences to account for the order of the words. The Word Subsequence Kernel (**WSK**) [32], which is an improvement over n-gram co-occurrences, considers the order by transforming the sentences into higher dimensional spaces and then measuring sentence similarity in that space. Extended String Subsequence Kernel (**ESK**) [85], which is a simple extension of WSK, allows one to incorporate word-specific syntactic or semantic (e.g., word sense, POS tags) information into WSK. Since Jeong et al. [93] found n-grams and POS tags useful for DA recognition, we implement the WSK and the ESK with POS tags (**ESK-P**).

Jeong et al. [93] also found that sub-trees extracted from the dependency trees are important features for DA recognition. We measure the dependency-based similarity between two sentences by first extracting their Basic Elements (**BE**) (i.e., *head-modifier-relation* triples) [88] from their corresponding dependency trees, and then by counting the number of BE co-occurrences in the two trees. The BEs encode some syntactic and semantic information and one can quite easily decide whether any two units match considerably more easily than with longer units (e.g., sentences, parse trees) [89].

Like dependency tree, the sub-trees of the syntactic tree may also be important indicators for DA identification. To measure the syntactic similarity between two sentences, we first parse the sentences using Charniak parser [41], and then compute the similarity between pairs of parse trees using the Tree Kernel (**TK**) function as described in [48].

### 4.4.2 Evaluation of the Graph-theoretic Model

We wish to compare the DAs automatically identified by our models with the human-labeled DAs. However, since unsupervised clustering techniques do not assign any label to the clusters, metrics widely used in supervised classification, such as the $\kappa$ statistic or $F_1$ score, are not applicable in our case. In this work, we propose to use the **one-to-one** metric [63] used earlier for measuring the performance of the topic segmentation models in Chapter 2. Briefly, given a human annotation and a model annotation, one-to-one optimally pairs up the clusters from the two annotations by maximizing the total overlap, and then reports the percent-

age of overlap found.

The number of DA categories available to the systems was fixed to 12. Table 4.2 shows the one-to-one accuracy of the graph-theoretic framework with various similarity metrics. The right most column shows the performance of the majority class baseline, that considers all the utterances in a corpus as *statements*.

| Corpus | BOW | BOW-M | WSK | ESK-P | BE | TK | BASELINE |
|---|---|---|---|---|---|---|---|
| **Email** | 62.6 | 34.3 | 64.7 | 24.8 | 39.1 | 22.5 | 70.0 |
| **Forum** | 65.0 | 38.2 | 65.8 | 36.3 | 46.0 | 30.1 | 66.0 |

**Table 4.2:** One-to-one accuracy for different similarity metrics in the graph-theoretic framework.

A comparison between the performances of BOW and BOW-M (i.e., BOW with nouns masked) systems demonstrates that BOW performs way better than BOW-M. This indicates that masking the nouns in an attempt to abstract away the topic words degrades the performance substantially. The WSK system performs slightly better than the BOW system meaning that considering the order of the words in the similarity metric is useful. However, when we add the POS tags of the words in ESK (see ESK-P in Table 4.2), the performance degrades dramatically. This means that similarity metric based on POS tags has an adverse effect on clustering sentences into DAs. The results of the BE and TK systems indicate that the shallow (dependency) and the deep syntactic similarities between sentences also are not useful for recognizing DAs in this framework. More interestingly, notice that all the systems fail to beat the majority class baseline indicating that this framework is not the right model for recognizing DAs in asynchronous conversations.

## 4.5 Probabilistic Conversational Models

The graph-theoretic framework described above has three key limitations when applied to cluster the sentences of an asynchronous conversation based on their act types. First, this framework does not capture the potentially informative sequential structure of a conversation; for example, an *answer* usually follows a *question*, an *accept* usually follows a *request* and so on. Second, this framework seems to be

still confused by topics even when we mask the nouns, or consider the word order, POS tags and syntactic structures. Third, unlike our probabilistic conversational models (described below), this framework does not allow us to incorporate other important features (e.g., *speaker, position, length*) beyond lexical and syntactic similarity between the sentences in a principled way. To address these limitations we propose probabilistic conversation models, which assume that a conversation is a Markov sequence of hidden DAs, with each DA emitting an observed sentence.

### 4.5.1 HMM Conversational Model

Figure 4.3 shows our first probabilistic conversational model in plate notation. An asynchronous conversation $C_k$ is a sequence of hidden DAs $D_i$, and each DA generates an observed sentence $X_i$, represented by its: (i) bag-of-words (i.e., unigrams) shown in the $W_i$ plate, (ii) speaker $S_i$, (iii) relative position $P_i$, and (iv) length $L_i$. These features take discrete values, and are modeled as multinomial distributions. Following Ritter et al. [174], for unigrams, we limit our vocabulary to the $5,000$ most frequent words in the corpus. The relative position of a sentence is computed by dividing its position in the post by the number of sentences in the post. We then convert the relative positions and lengths to a sequence of natural numbers.

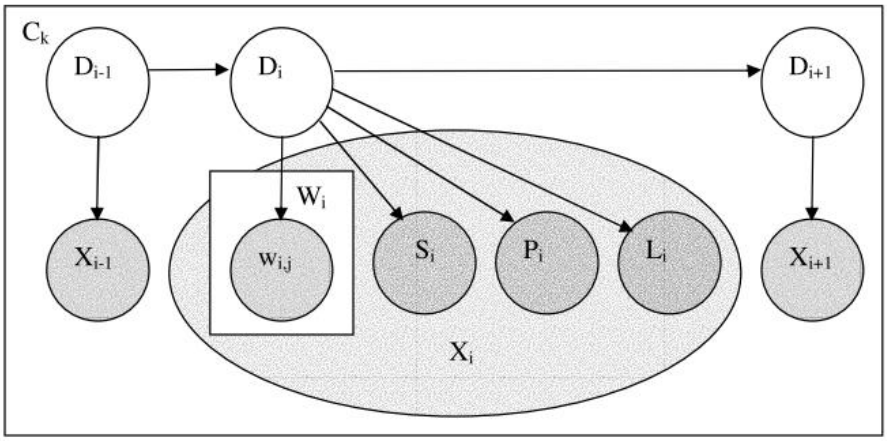



**Figure 4.3:** HMM conversational model

We place a symmetric Dirichlet prior (with hyperparameter $\alpha = 2$) on each of the multinomials (i.e., distributions over initial states, state transitions, unigrams,

speakers, positions and lengths)[7], and compute a maximum a posteriori (MAP) estimate of the parameters using the Baum-Welch (EM) algorithm with forward-backward providing the smoothed node and edge marginals for each sequence in the E-step. Specifically, given a sequence $X_{1:T}$, forward-backward computes:

$$\gamma_i(j) := P(D_i = j|X_{1:T}, \theta) \tag{4.1}$$

$$\xi_i(j,k) := P(D_{i-1} = j, D_i = k|X_{1:T}, \theta) \tag{4.2}$$

where the local evidence is given by:

$$P(X_i|D_i) = \left[\prod_j P(W_{i,j}|D_i)\right] P(S_i|D_i)P(P_i|D_i)P(L_i|D_i) \tag{4.3}$$

### 4.5.2 HMM+Mix Conversational Model

The HMM conversational model described above is similar to the conversational model of [174] except that, in addition to the unigrams of a sentence, we also use its relative position, speaker and length to model the emission distribution. However, as they point out, without additional guidance this unsupervised model may find some undesired clusters. For example, their HMM model with unigram feature tends to find some clusters that are based on topics rather than DAs. Since the features used in our conversational model are indicators of DAs as well as of topics (see Section 2.3.1 in Chapter 2), our model with the extended feature set may also find some unwanted clusters. Note that similar to Ritter et al. [174], we have also noticed in our graph-theoretic framework that masking nouns in an attempt to abstract away the topics even degrades the clustering performance. As a solution, Ritter et al. [174] propose the HMM+Topic model to separate the influence of the topic words from the DA indicators. In this model, a word is generated from one of the three hidden sources: (i) DA, (ii) Topic and (iii) General English. In contrast, we propose the HMM+Mix model where the emission distribution is defined as a mixture model. This model has two main advantages over the HMM+Topic

[7]The parameters were not shown in Figures 4.3 and 4.4 to reduce visual clutter.

model: first, by defining the emission distribution as a mixture model we not only indirectly explain away the topics but also enrich the emission distribution, since the mixture models can define multimodal distributions [23], and second, learning and inference in this model is much easier (using EM) without requiring approximate inference techniques such as Gibbs sampling.

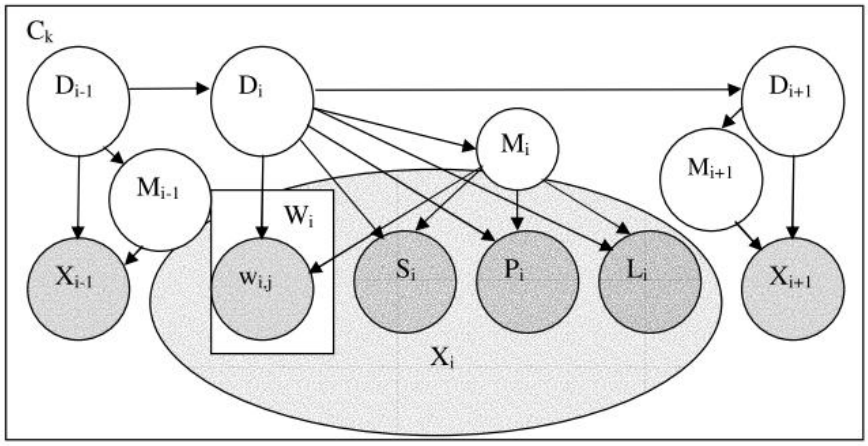



**Figure 4.4:** HMM+Mix conversational model

Figure 4.4 shows the extended HMM+Mix model, where emission distributions are modeled as mixtures of multinomials with $M_i \in \{1,\ldots,M\}$ representing the mixture component. Notice also that each observed sentence $X_i$ is now explained by two hidden causes (parents), i.e., its dialog act $D_i$ and the mixture component $M_i$. Therefore, the two causes will try to explain each other away to explain an observed sentence. Similar to the previous model, we put symmetric Dirichlet priors (with hyperparameter $\alpha = 2$) on the multinomials and compute MAP estimates of the parameters using EM, where the local evidence is given by:

$$P(X_i|D_i) = \sum_{M_i} p(M_i|D_i)P(X_i|D_i,M_i) \tag{4.4}$$

we define $P(X_i|D_i,M_i)$ in a way similar to Naive Bayes:

$$P(X_i|D_i,M_i) = \left[\prod_j P(W_{i,j}|D_i,M_i)\right] P(S_i|D_i,M_i)P(P_i|D_i,M_i)P(L_i|D_i,M_i) \quad (4.5)$$

In this model, for each sequence $X_{1:T}$, in addition to $\gamma_i(j)$ and $\xi_i(j,k)$ (Equation 4.1, 4.2), we also need to compute $\tau_i(j,k)$ in the E-step:

$$\tau_i(j,k) := P(D_i = j, M_i = k|X_{1:T},\theta) \quad (4.6)$$

One can show that this is given by the following expression:

$$\tau_i(j,k) := \gamma_i(j)\frac{P(M_i = k|D_i = j)P(X_i|D_i = j,M_i = k)}{\sum_m P(M_i = m|D_i = j)P(X_i|D_i = j,M_i = m)} \quad (4.7)$$

The details of the EM algorithm for this model is given in Appendix A.3.

### 4.5.3 Initialization in EM

In EM, we must ensure that we initialize the model parameters carefully to minimize the chance of getting stuck in poor local optima. We use multiple (10) restarts and pick the best solution based on the estimated likelihood of the data. For the first run, we ignore Markov dependencies between the DAs, and estimate the parameters of the emission distributions using the standard mixture model estimation method (i.e., using EM), and later use it to initialize other parameters (i.e., initial state distribution and state transition distributions). For the other 9 runs, we randomly initialize the parameters. We use this process of initialization in an attempt to ensure that our models are at least as good as the simple mixture model, which ignores the sequential structure of an asynchronous conversation.

### 4.5.4 Applying Conversational Models

We apply (i.e., train and test) the conversational models described above to the temporal order of the utterances and to the sequences extracted from the graph structure of the conversation. With the learned parameters, given a (test) sequence of sentences, we use Viterbi decoding to infer the most probable DA sequence. However, as described in Section 4.3.2, sequences extracted from the graph struc-

ture of a conversation may have duplicate sentences. As a result, when we run Viterbi decoding on it, for the same sentence we get multiple DA assignments (one for each position). We take the maximum vote to finally determine its DA.

## 4.6 Experiments

In our experiments with email and forum corpora, we create 50 training samples from a corpus, each sample containing 12,000 randomly selected conversations. When selecting the conversations, we ensure that each conversation contains at least two posts. For a corpus, we train our conversational models on each of the 50 training samples (each containing 12,000 conversations), and evaluate the performance on the test set each time. We train our models both on the temporal order of the utterances and on the order extracted from the graph structure of the conversation. The number of DAs available to the models was set to 12. In the HMM+Mix model, the number of mixture components M was empirically set to 3.[8]

| | **Email** | | **Forum** | |
|---|---|---|---|---|
| | Temporal | Graph | Temporal | Graph |
| BASELINE | 70.00 | 70.00 | 66.00 | 66.00 |
| HMM | 73.45 | 76.81 | 69.67 | 74.41 |
| HMM+Mix | 76.73 | **79.66** | 75.61 | **78.35** |

**Table 4.3:** Mean one-to-one accuracy for various models on the two corpora.

Table 4.3 presents the mean one-to-one accuracy of our conversational models and the majority class baseline on the two corpora. Our models beat the baseline in both corpora, proving their effectiveness in DA recognition. When we compare the performances of the models on the two corpora, we notice a similar trend. Both models benefit significantly from the graph-structure of the conversations ($p<0.05$). The finer conversational structure of email conversations in the form of FQG and the assumed conversational structure of forum conversations have been proved to be beneficial for recognizing DAs in these corpora. A comparison between our models shows that the HMM+Mix model outperforms the simple HMM

[8] We experimented with M=$\{1, 2, 3, 4, 5\}$, and achieved the highest accuracy with M=3.

($p<0.05$). This indicates that the mixture model acting as the emission distribution not only explains the topics away but also defines a finer observation model.

## 4.7 Conclusion and Future Work

In our investigation of approaches for modeling DAs in asynchronous conversations we have made several key contributions. We apply a graph-theoretic framework to the DA tagging task and compare it with probabilistic sequence-labeling models. Then, we show how in the probabilistic models the sequence dependencies can be more effectively learned by taking the conversational structure into account. After that, we successfully expand the set of sentence features considered in the act-emission distribution. Finally, we improve the act-emission distribution by applying a mixture model. Quantitative evaluation with human annotations shows that while the graph-theoretic framework is not the right model for this task, the probabilistic conversation models (i.e., HMM and HMM+Mix) are quite effective and their benefits are more pronounced with graph-structural conversational order as opposed to the temporal one. Comparison of the outputs of these models reveals that HMM+Mix model can predict DAs better than the HMM model. In the future, we wish to experiment with the Bayesian versions of these conversation models and also apply our models to other conversational modalities.

# Chapter 5

# Conclusion

As the Internet continues to grow, billions of people all over the world are now having conversations by writing in a growing number of social media in a seemingly unlimited variety of communicative settings. Effective processing of these asynchronous conversations can benefit both organizations and individuals. As a consequence, end-user applications (e.g., summarization, information extraction) targeting conversations in social media are now growing at an accelerating pace, and since these conversations are different in many aspects from the traditional media (e.g., news paper), NLP researchers are now facing new challenges to build tools (e.g., POS tagger, topic segmenter) to support these downstream applications.

In this thesis, we developed a new set of NLP tools for different discourse analysis tasks in asynchronous conversation, which can support many NLP applications including summarization and information extraction. In particular, we presented novel computational models for topic segmentation and labeling, rhetorical analysis and dialog act recognition in asynchronous conversation. Our initial hypothesis was that we can effectively address the technical challenges in these tasks by applying sophisticated graph-based methods and probabilistic graphical models. Graph-based methods allow us to encode different discourse-related information (e.g., conversational structure and other domain-specific features) in terms of the edge weights in the graph so that all the information can be simultaneously taken into account while performing discourse analysis tasks. Similarly, probabilistic graphical models allow for efficient and sound reasoning about multiple

interrelated random variables (i.e., structured output) in a compact but semantically intuitive graph representation. The successful application of these models to a complex task (e.g., a discourse analysis task in asynchronous conversations) is not a trivial matter. It requires identifying what aspects of the conversation should be represented for the target task, how they should be represented in the model, as well as how learning and inference in the resulting representations can be performed effectively. And this in-depth investigation was done in this thesis for each of the discourse analysis tasks. Our hypothesis was successfully verified by showing that all the models we devised in this thesis outperform state-of-the art techniques on widely accepted performance metrics.

In chapter 2, we studied topic segmentation and labeling in asynchronous conversation. We annotated two new corpora, measured inter-annotator agreement and presented a complete computational framework for performing these tasks. We first showed that existing off-the-shelf tools are insufficient for these tasks, because they do not consider the conversation specific features of asynchronous conversations. In response, we proposed two novel unsupervised topic segmentation models that take into account the fine-grained conversational structure, and a novel supervised topic segmentation model that combines lexical, conversational and topic features. We did so either by using a more informative prior or by using a graph-based clustering framework with no or little supervision. We demonstrated that unsupervised topic segmentation models benefit significantly when they consider the fine-grained conversational structure. Our comparison of the supervised segmentation model with the unsupervised models showed that the supervised method outperforms the unsupervised ones even using only a few labeled conversations.

For topic labeling, we proposed two novel graph-based ranking (i.e., random walk) models that respectively captures two forms of conversation specific information: the informative clues from the leading sentences and the fine-grained conversational structure. We demonstrated that the random walk model performs significantly better when it exploits the conversation specific clues from either of the two sources, and the random walk model that considers both sources of information performs consistently well across domains. The overall performance of the end-to-end system is also promising when compared with human annotations.

We made our topic annotated conversational corpora, annotation manual and

softwares available online. Researchers from several organizations including Microsoft Research Asia, Cisco Research and the Qatar Computing Research Institute (QCRI) are now using our corpora as well as softwares for research purposes. There is also an ongoing research project at the University of British Columbia on effective visualization of asynchronous conversations using our developed tools.[1]

In Chapter 3, we presented our complete discriminative framework for rhetorical analysis. We identified crucial limitations in the parsing model and in the parsing algorithm of the existing discourse parsers, and addressed them by proposing more adequate parsing models and an optimal parsing algorithm.

We employed probabilistic graphical models as our parsing models which allow efficient learning and reasoning in a compact graph representation. While existing discourse parsers use a single unified parsing model, we proposed to use two different parsing models – one for intra-sentential parsing and one for multi-sentential parsing. More specifically, we proposed a Dynamic Conditional Random Field (DCRF) as our intra-sentential parsing model which represents the structure and the labels of a discourse tree jointly and captures the sequential dependencies between the tree constituents. We demonstrated that our DCRF parsing model along with an optimal parsing algorithm outperforms the state-of-the-art by a wide margin in intra-sentential parsing. However, for multi-sentential parsing, we faced the challenge of scaling up the DCRF model to arbitrary long documents. So, we proposed to break the chain structure in the DCRF model for multi-sentential parsing, which also allowed us to balance the training data. We showed that using two separate parsing models result in a more effective parsing method, because the models could leverage their own informative feature sets and the distinct distributions of the coherence relations.

A two-stage parsing approach also allowed us to exploit the strong correlation between the text structure and the tree structure. We proposed to capture this correlation by combining our intra-sentential and multi-sentential parsers in two different ways. While our first approach did not consider the cases where rhetorical structures violate sentence boundaries (i.e., leaky boundaries), our second approach attempted to tackle them using a sliding window over two consecutive sentences.

[1]More on this research can be found at https://www.cs.ubc.ca/cs-research/lci/research-groups/intelligent-user-interfaces

We demonstrated that both of our approaches outperform the state-of-the-art by a large margin. A comparison between our approaches showed that the sliding window approach is more accurate in finding the right structure of the discourse trees, especially when it gets enough data to learn from.

Recently, we made a demo and the source code of our discourse parsing framework available online. Researchers from several universities including the University of Toronto and the University of Utah are now using our discourse parser.

In Chapter 4, we presented both deterministic and probabilistic unsupervised approaches to dialog act modeling in asynchronous conversation. While the deterministic (i.e., graph-based clustering) models did not consider the sequential dependencies between the act types, the probabilistic conversational (graphical) models (i.e., HMMs) did. First, we demonstrated that like synchronous conversations, it is crucial to capture the sequential dependencies between the acts in asynchronous conversations. Then, we showed that the probabilistic conversational models learn better sequential dependencies when trained on the sequences extracted from the conversational structure, rather than on the temporal order of the sentences.

However, it turns out that the simple unsupervised HMM model tends to find some undesired topical clusters in addition to dialog act clusters. We addressed this by proposing a novel HMM+Mix model, which not only explains away the topics, but also improves the act emission distribution by defining it as a mixture model.

## 5.1 Prospective Applications

There are many NLP applications that can be significantly benefited from our extracted discourse structures. We briefly describe the most important ones below.

### 5.1.1 Conversation Summarization

One of the most important applications of discourse structures is text summarization. A number of summarization methods utilizing topic structures have been proposed for monologs (e.g., [20, 56, 80, 115]). Topic-based summarization methods have also been explored for summarizing meeting transcripts [104, 221]. Murray et al. [148, 151] show the benefits of using dialog acts in meeting summarization. A number of research investigate the utility of rhetorical structure for measuring

text importance in single document summarization (e.g., [54, 118, 126]). Recently, Christensen et al. [45] propose to incorporate coherence structure into their multi-document summarization system for selecting and ordering sentences.

Most of the above approaches to summarization are **extractive**, where informative sentences are selected to form an abridged version of the source document(s). This approach has been quite successful for summarizing factual texts (e.g., news articles, biographies), and has also been applied to asynchronous conversation [35, 146, 169]. This approach has been by far the most popular in the field of summarization, largely because it does not require to generate novel sentences.

Another potential approach to summarization is **abstractive**, which first extracts key information (e.g., entities) and possibly the discourse structures, and then generates novel texts based on the extracted information and structures. Recent studies [133, 149] have shown that although users find the extractive summaries to be valuable for browsing documents, these summaries are less coherent than human written abstracts and users in general prefer abstracts over extracts. Thus, there is now a growing interest in abstractive summarization, even for conversations [151, 222]. Carenini et al. [36] postulate an abstractive summarization system for conversations that can take us closer to the human-style summarization of conversations. To our knowledge, with the exception of a partial proposal presented in [150], none has attempted to abstract asynchronous conversations yet, and no existing extractive methods use deep discourse structures (i.e., topic, coherence and dialog act structures).[2] We strongly believe that the discourse structures that we extract in this work will be very beneficial for both extractive and abstractive summarization approaches in asynchronous conversations. For an example of moving towards abstraction, see our recent paper [136] which describes how an abstract topic label can be generated for a given topical cluster.

### 5.1.2 Sentiment Analysis

Discourse structures can play an important role in sentiment analysis. A research problem in sentiment analysis is extracting fine-grained opinions about different aspects (or features) of a product. Several recent work (e.g., [112, 189]) have

[2] Carenini et al. [34, 35] use a fine-grained conversational structure to measure text importance in their summarization system.

already considered to exploit the rhetorical structure for this task.

Another challenging problem in sentiment analysis is assessing the overall opinion expressed in a review because not all sentences in a review contribute equally to the overall sentiment. For example, some sentences are subjective, while others are objective [157]; some express the main claims, while others support them [207]; some express opinions about the main entity, while others are about the peripherals. Discourse structures (i.e., rhetorical and topical structures) could be useful to capture the relative weights of the discourse units towards the overall sentiment. For example, the nucleus and satellite distinction along with the rhetorical relations could be useful to infer the relative weights of the connecting discourse units. Similarly, topical structures could be useful to distinguish between opinions expressed towards the main entity and opinions expressed towards the peripherals.

Topic models along with sentiment analysis could be useful in many interesting applications including analysis of perspectives (e.g., left vs. right, Palestinian vs. Israeli) [116, 152], prediction of election outcomes [211], and so on.

### 5.1.3 Information Extraction and Visualization

Topic models like LDAs coupled with effective user interfaces are now extensively used to facilitate efficient browsing and exploring a large document collection [25, 40, 58, 117, 156]. Similarly, our topic segmentation and labeling models would allow us to effectively browse a possibly complex and long conversation.

Topic and rhetorical structures have been used to select the parts of a document which are relevant to extract *named entities, the relations* that hold between them, and the *events* where they play a role (e.g., [130, 142, 209]). Rhetorical structure has also been used for extracting answers of *why* questions [215]. The dialog act structure has been used to detect action items in meetings [147, 166].

However, the extracted information (e.g., topics, sentiment, dialog acts) are of little use if these cannot be conveyed effectively to the user. Conversation visualization tools with interactive multimedia interfaces offer attractive solutions for conveying large and complex information using graphics and texts. Arguello and Rosé [8] developed such an interactive tool for visualizing topical segments in monolog and synchronous dialog (e.g., tutoring dialog). Recently, Rashid [172]

proposes such a tool for browsing and summarizing meeting conversations. Dork et al. [58] propose a visualization tool for monitoring events in Twitter. Effective visualization of the information extracted from asynchronous conversations is an active research area which can heavily benefit from our work.

### 5.1.4 Misc.

Among other applications of discourse structures, Machine Translation (MT) has received a resurgence of interest recently. Researchers believe that MT systems should consider discourse phenomena that go beyond the current sentence to ensure consistency in the choice of lexical items or referring forms, and the fact that source-language coherence relations between discourse units are also realized in the target language [82]. For a survey on discourse in MT see [81]. A workshop on discourse in MT was arranged recently at the ACL conference.[3]

Another application for research is Computer Supportive Collaborative Learning (CSCL). Automatic methods using NLP techniques to analyze online discussions for CSCL (e.g., argumentation) have been proposed [9, 144, 175, 227]. We believe, our discourse analysis tools can be also beneficial for these applications.

## 5.2 Future Directions

Moving forward, there are still many unsolved and fundamental challenges to be addressed in discourse analysis of asynchronous conversations. While specific venues for future work on each discourse analysis task were discussed at the end of the corresponding chapters, here we briefly outline a number of general future directions that arise from the work presented in this thesis.

- **Extrinsic Evaluation:** In all the evaluations in this thesis, we measured how the discourse analysis models perform with respect to human gold standard. It would be extremely interesting to see how effective our models are when the extracted discourse structures are exploited in the target end-user applications like conversation summarization and conversation visualization.

- **Interactive Discourse Analysis:** A key limitation researchers face working

[3] http://www.idiap.ch/workshop/DiscoMT/

on discourse, especially if the discourse is of a new kind (e.g., asynchronous conversation), is scarcity of annotated data. To address this problem, in the future we would like to develop an online version of our systems that would allow users to fix the output of the system and let the model learn from that feedback.

• **A Joint Model Integrating All Tasks:** In this thesis, we considered the three discourse analysis tasks (i.e., topic modeling, rhetorical analysis, dialog structure analysis) separately. A promising avenue of investigation would be to integrate all these tasks in a mutually beneficial way using a joint model. To what extent these tasks can be jointly modeled is an open question for future research [36].

• **Towards Coherent Summarization:** A significant challenge in summarization is producing not only informative but also coherent summaries. Existing approaches usually follow a two-step approach, where a *sentence ordering* step follows a *sentence selection* step to make the summary coherent. However, we believe coupling these two steps into a joint process would be more effective because if the sentences are selected first without paying any attention to coherence then an expected (coherent) ordering of the selected sentences may not exist.

Recently, Christensen et al. [45] attempt to perform sentence selection and ordering together using constraints on a discourse structure. However, they represent the discourse as an unweighted directed graph which is shallow and not sufficiently informative in most cases. Furthermore, their approach does not allow compression at the sentence level, which is often beneficial in summarization. In the future, we would like to investigate the utility of our discourse structures for performing sentence compression, selection and ordering in a joint process.

• **Combining Cohesion and Coherence for Summarization:** A line of previous research in unsupervised extractive summarization used *cohesion* as a measure of importance for sentence selection (e.g., [17, 35, 65]). These methods use simple representations that are based on lexical similarity. On the other hand, another line of research used *coherence* structure to measure the importance of a sentence (or clause) [54, 118, 126]. It would be interesting to see how a summarization approach that combines these two phenomena of discourse together in a single model performs. The work in this thesis would allow that because our rhetorical

structure (i.e., a discourse tree) provides the necessary coherence structure.

• **Towards Discourse Informed Sentiment Analysis:** We would like to investigate how discourse structures could play a role in assessing the overall sentiment expressed in a review by aggregating the sentiments expressed in its components (e.g., clauses). In doing so, we see the need to consider both the topical and the rhetorical structures. Topical structure would allow us to distinguish opinions expressed towards main vs. peripheral entities. Similarly, rhetorical structure would allow us to distinguish between main and supportive claims.

• **Applications in Social Media:** Finally, we would like to work on problems in social computing that involve conversation understanding and generation. For example, imagine an autonomous system that monitors social media and identifies conversations about humanitarian crisis (e.g., Hurricane Sandy, earthquake in Pakistan). The system would extract necessary information (e.g., requesting or offering food, shelter), and in cases it needs further information (or to verify the extracted information), the system would ask for it by generating a post in the conversation. The system would then map the offers to the corresponding requests and let the victims know about the offers.

Another research avenue in social computing that we would like to investigate is how to leverage the strengths of social media and traditional news media to provide a holistic view on a developing event. Users of social media as well as journalists of news articles may have very different perspectives (e.g., left vs. right, Democrat vs. Republican) on an event. We believe an aggregated view would be very useful for a general user.

# Bibliography


[1] P. H. Adams and C. H. Martell. Topic Detection and Extraction in Chat. In *Proceedings ICSC-2008*, pages 581–588. IEEE, 2008. → pages 23

[2] A. Alexandrescu and K. Kirchhoff. Graph-based learning for statistical machine translation. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 119–127, Boulder, Colorado, June 2009. Association for Computational Linguistics. URL http://www.aclweb.org/anthology/N/N09/N09-1014. → pages 28

[3] J. Allan. *Topic Detection and Tracking: Event-based Information Organization*, pages 1–16. Kluwer Academic Publishers, Norwell, MA, USA, 2002. → pages 8, 39

[4] J. Allan, C. Wade, and A. Bolivar. Retrieval and Novelty Detection at the Sentence Level. In *Proceedings of the 26th annual international ACM SIGIR conference on Research and development in informaion retrieval*, SIGIR '03, pages 314–321, Toronto, Canada, 2003. ACM. → pages 71

[5] J. Allen, N. Chambers, G. Ferguson, L. Galescu, H. Jung, and W. Taysom. PLOW: A Collaborative Task Learning Agent. In *Proceedings of the Twenty-Second Conference on Artificial Intelligence*, AAAI'07, pages 22–26. AAAI, 2007. → pages 22, 147

[6] D. Andrzejewski, X. Zhu, and M. Craven. Incorporating Domain Knowledge into Topic Modeling via Dirichlet Forest Priors. In *Proceedings of the 26th Annual International Conference on Machine Learning*, ICML '09, pages 25–32, Montreal, Quebec, Canada, 2009. ACM. → pages 58

[7] P. M. Aoki, M. H. Szymanski, L. D. Plurkowski, J. D. Thornton, A. Woodruff, and W. Yi. Where's the "party" in "multi-party"?: analyzing the structure of small-group sociable talk. In *CSCW '06*, pages 393–402, Banff, Canada, 2006. ACM. → pages 26

[8] J. Arguello and C. Rosé. Infomagnets: Making sense of corpus data. In *Proceedings of the Human Language Technology Conference of the NAACL, Companion Volume: Demonstrations*, pages 253–256, New York City, USA, June 2006. Association for Computational Linguistics. → pages 170

[9] J. Arguello, B. S. Butler, E. Joyce, R. Kraut, K. S. Ling, C. Rosé, and X. Wang. Talk to me: foundations for successful individual-group interactions in online communities. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '06, pages 959–968, Montréal, Québec, Canada, 2006. ACM. → pages 171

[10] N. Asher and A. Lascarides. *Logics of Conversation*. Cambridge University Press, 2003. → pages 17, 101

[11] E. Aumayr, J. Chan, and C. Hayes. Reconstruction of Threaded Conversations in Online Discussion Forums. In *Proceedings of the Fifth International AAAI Conference on Weblogs and Social Media*, ICWSM'11, pages 26–33. AAAI, 2011. → pages 48, 155

[12] J. L. Austin. How to do things with words. *Harvard University Press*, 1962. → pages 22, 147

[13] S. Banerjee and A. I. Rudnicky. Segmenting meetings into agenda items by extracting implicit supervision from human note-taking. In *Proceedings of the 12th international conference on Intelligent user interfaces*, IUI '07, pages 151–159, Honolulu, Hawaii, USA, 2007. ACM. → pages 11

[14] S. Bangalore, G. Di Fabbrizio, and A. Stent. Learning the Structure of Task-Driven Human-Human Dialogs. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics*, COLING-ACL'06, pages 201–208. ACL, 2006. → pages 8, 23, 39, 147

[15] N. Bansal, A. Blum, and S. Chawla. Correlation Clustering. *Machine Learning*, 56:89–113, 2004. ISSN 1-3. → pages 28, 48, 57

[16] N. S. Baron. Always On: Language in an Online and Mobile World. *Oxford ; New York : Oxford University Press*, 2008. → pages 1

[17] R. Barzilay and M. Elhadad. Using Lexical Chains for Text Summarization. In *Proceedings of the 35th Annual Meeting of the*

*Association for Computational Linguistics and the 8th European Chapter Meeting of the Association for Computational Linguistics, Workshop on Intelligent Scalable Test Summarization*, pages 10–17, Madrid, 1997. ACL. → pages 121, 172

[18] R. Barzilay and M. Lapata. Modeling local coherence: an entity-based approach. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, ACL '05, pages 141–148, Ann Arbor, Michigan, 2005. Association for Computational Linguistics. → pages 48

[19] R. Barzilay and L. Lee. Catching the Drift: Probabilistic Content Models, with Applications to Generation and Summarization. In *Proceedings of the Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics*, HLT-NAACL'04. ACL, 2004. → pages 31, 39

[20] R. Barzilay and K. R. McKeown. Sentence Fusion for Multidocument News Summarization. *Computational Linguistics*, 31:297–328, September 2005. ISSN 0891-2017. → pages 168

[21] D. Beeferman, A. Berger, and J. Lafferty. Statistical Models for Text Segmentation. In *Machine Learning*, volume 34, pages 177–210. Kluwer Academic Publishers, Feb. 1999. → pages 9, 10, 81

[22] O. Biran and O. Rambow. Identifying Justifications in Written Dialogs by Classifying Text as Argumentative. *International Journal of Semantic Computing*, 5(4):363–381, 2011. → pages 119

[23] C. M. Bishop. *Pattern Recognition and Machine Learning*. Springer, 2006. → pages 153, 161

[24] S. Blair-Goldensohn, K. McKeown, and O. Rambow. Building and Refining Rhetorical-Semantic Relation Models. In *Proceedings of the Human Language Technologies: The Annual Conference of the North American Chapter of the Association for Computational Linguistics*, HLT-NAACL'07, pages 428–435. ACL, 2007. → pages 20, 106

[25] D. Blei. Topic Modeling and Digital Humanities. *Journal of Digital Humanities*, 2, 2013. ISSN 1. → pages 170

[26] D. Blei, A. Ng, and M. Jordan. Latent Dirichlet Allocation. *The Journal of Machine Learning Research*, 3:993–1022, Mar. 2003. ISSN 1532-4435. → pages 10, 13, 32, 45, 50, 51

[27] D. M. Blei and P. J. Moreno. Topic Segmentation with an Aspect Hidden Markov Model. In *Proceedings of the 24th annual international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '01, pages 343–348, New Orleans, Louisiana, USA, 2001. ACM. → pages 9, 10, 45

[28] J. Blitzer. *Domain Adaptation of Natural Language Processing Systems*. PhD thesis, University of Pennsylvania, Pennsylvania, 2008. → pages 119

[29] J. Boyd-Graber and D. M. Blei. Syntactic Topic Models. In *Neural Information Processing Systems*, NIPS'08, 2008. → pages 52

[30] L. Breiman. Bagging Predictors. *Machine Learning*, 24(2):123–140, Aug. 1996. ISSN 0885-6125. → pages 65, 129

[31] S. Brin and L. Page. Tha anatomy of a large-scale hypertextual web search engine. In *Computer Networks*, volume 30(1-7), pages 107–117, 1998. → pages 29

[32] N. Cancedda, E. Gaussier, C. Goutte, and J. M. Renders. Word Sequence Kernels. *Journal of Machine Learning Research (JMLR)*, 3:1059–1082, 2003. → pages 149, 157

[33] J. Carbonell and J. Goldstein. The Use of MMR, Diversity-based Reranking for Reordering Documents and Producing Summaries. In *Proceedings of the 21st annual international ACM SIGIR conference on Research and development in information retrieval*, pages 335–336, Melbourne, Australia, 1998. ACM. → pages 46, 76

[34] G. Carenini, R. T. Ng, and X. Zhou. Summarizing Email Conversations with Clue Words. In *Proceedings of the 16th international conference on World Wide Web*, WWW'07, pages 91–100, Banff, Canada, 2007. ACM. → pages 26, 27, 49, 54, 169

[35] G. Carenini, R. T. Ng, and X. Zhou. Summarizing Emails with Conversational Cohesion and Subjectivity. In *Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, ACL-HLT'08, pages 353–361, OH, 2008. ACL. → pages 17, 27, 49, 55, 72, 154, 169, 172

[36] G. Carenini, G. Murray, and R. Ng. *Methods for Mining and Summarizing Text Conversations*, volume 3. Morgan Claypool, 2011. → pages 1, 8, 25, 26, 48, 77, 169, 172

[37] L. Carlson and D. Marcu. Discourse Tagging Reference Manual. Technical Report ISI-TR-545, University of Southern California Information Sciences Institute, 2001. URL http://www.isi.edu/~marcu/discourse/tagging-ref-manual.pdf. → pages 136

[38] L. Carlson, D. Marcu, and M. Okurowski. RST Discourse Treebank (RST-DT) LDC2002T07. *Linguistic Data Consortium, Philadelphia*, 2002. → pages 17, 21, 102, 105, 131

[39] V. R. Carvalho and W. W. Cohen. On the Collective Classification of Email "Speech Acts". In *Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '05, pages 345–352, New York, NY, USA, 2005. ACM. → pages 25, 151

[40] A. Chaney and D. Blei. Visualizing Topic Models. In *Proceedings of the Sixth International AAAI Conference on Weblogs and Social Media*, ICWSM'12, pages 419–422, 2012. → pages 170

[41] E. Charniak. A Maximum-Entropy-Inspired Parser. In *Technical Report CS-99-12*, Brown University, Computer Science Department, 1999. → pages 157

[42] E. Charniak. A Maximum-Entropy-Inspired Parser. In *Proceedings of the 1st North American Chapter of the Association for Computational Linguistics Conference*, NAACL'00, pages 132–139, Seattle, Washington, 2000. ACL. → pages 134

[43] E. Charniak and M. Johnson. Coarse-to-Fine n-Best Parsing and MaxEnt Discriminative Reranking. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics*, ACL'05, pages 173–180, NJ, USA, 2005. ACL. → pages 134

[44] F. Y. Y. Choi, P. W. Hastings, and J. Moore. Latent Semantic Analysis for Text Segmentation. In *Proceedings of the 2001 Conference on Empirical Methods in Natural Language Processing*, EMNLP'01, pages 109–117, Pittsburgh, USA, 2001. ACL. → pages 9, 10, 43, 44, 64

[45] J. Christensen, Mausam, S. Soderland, and O. Etzioni. Towards Coherent Multi-Document Summarization. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, NAACL-HLT'13, pages 1163–1173, Atlanta, Georgia, June 2013. ACL. → pages 169, 172

[46] W. W. Cohen, V. R. Carvalho, and T. M. Mitchell. Learning to Classify Email into “Speech Acts”. In *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing*, EMNLP’04, pages 309–316, 2004. → pages 25, 151

[47] M. Collins. Head-Driven Statistical Models for Natural Language Parsing. *Computational Linguistics*, 29(4):589–637, Dec. 2003. ISSN 0891-2017. → pages 135

[48] M. Collins and N. Duffy. Convolution Kernels for Natural Language. In *Neural Information Processing Systems*, NIPS’01, pages 625–632, Vancouver, Canada, 2001. → pages 149, 157

[49] C. Cortes and V. N. Vapnik. Support Vector Networks. *Machine Learning*, 20:273–297, 1995. → pages 10, 61

[50] D. Cristea, N. Ide, and L. Romary. Veins theory: A model of global discourse cohesion and coherence. In *In Proceedings of the 36th Annual Meeting of the Association for Computational Linguistics and of the 17th International Conference on Computational Linguistics (COLING/ACL98)*, pages 281–285, 1998. → pages 101

[51] D. Crystal. *Language and the Internet*. Cambridge University Press, 2001. → pages 39

[52] L. Danlos. D-STAG: a Discourse Analysis Formalism based on Synchronous TAGs. *TAL*, 50(1):111–143, 2009. → pages 101

[53] H. Daume. Frustratingly Easy Domain Adaptation. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics*, ACL’07, pages 256–263, Prague, Czech Republic, 2007. ACL. → pages 136

[54] H. Daumé, III and D. Marcu. A noisy-channel model for document compression. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, ACL ’02, pages 449–456, Philadelphia, Pennsylvania, 2002. Association for Computational Linguistics. doi:10.3115/1073083.1073159. URL http://dx.doi.org/10.3115/1073083.1073159. → pages 18, 101, 169, 172

[55] R. Dhillon, S. Bhagat, H. Carvey, and E. Shriberg. Meeting Recorder Project: Dialog Act Labeling Guide. Technical report, ICSI Tech. Report, 2004. URL http:

//www.icsi.berkeley.edu/ftp/global/pub/speech/papers/MRDA-manual.pdf. → pages 153

[56] G. Dias, E. Alves, and J. G. P. Lopes. Topic Segmentation Algorithms for Text Summarization and Passage Retrieval: an Exhaustive Evaluation. In *Proceedings of the 22nd national conference on Artificial intelligence - Volume 2*, pages 1334–1339, Vancouver, BC, Canada, 2007. AAAI. → pages 8, 39, 168

[57] A. Dielmann and S. Renals. DBN based Joint Dialogue Act Recognition of Multiparty Meetings. In *Proceedings of the IEEE International Conference on Acoustics Speech and Signal Processing*, ICASSP '07, 2007. → pages 23, 147

[58] M. Dork, D. Gruen, C. Williamson, and S. Carpendale. A Visual Backchannel for Large-Scale Events. *IEEE Transactions on Visualization and Computer Graphics*, 16(6):1129–1138, Nov. 2010. → pages 170, 171

[59] G. Durrett, D. Hall, and D. Klein. Decentralized entity-level modeling for coreference resolution. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 114–124, Sofia, Bulgaria, August 2013. Association for Computational Linguistics. → pages 31

[60] D. duVerle and H. Prendinger. A Novel Discourse Parser based on Support Vector Machine Classification. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, pages 665–673, Suntec, Singapore, 2009. ACL. → pages 111, 118, 141

[61] J. Eisenstein. Hierarchical Text Segmentation from Multi-scale Lexical Cohesion. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, NAACL '09, pages 353–361, Boulder, Colorado, 2009. ACL. → pages 9, 10, 31, 45

[62] J. Eisenstein and R. Barzilay. Bayesian Unsupervised Topic Segmentation. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, EMNLP '08, pages 334–343, Honolulu, Hawaii, 2008. ACL. → pages 10, 15, 45, 60

[63] M. Elsner and E. Charniak. Disentangling Chat. *Computational Linguistics*, 36:389–409, 2010. ISSN 3. → pages 23, 26, 28, 38, 48, 60, 64, 81, 82, 153, 156, 157

[64] M. Elsner and E. Charniak. Disentangling Chat with Local Coherence Models. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies - Volume 1*, HLT '11, pages 1179–1189, Portland, Oregon, 2011. ACL. → pages 48

[65] G. Erkan and D. Radev. LexRank: Graph-based Lexical Centrality as Salience in Text Summarization. *Journal of Artificial Intelligence Research*, 22:457–479, 2004. → pages 28, 29, 172

[66] A. Eshghi and P. G. Healey. What is conversation? distinguishing dialogue contexts. In *In THE ANNUAL MEETING OF THE COGNITIVE SCIENCE SOCIETY (CogSci 2009)*, pages 1240–1245, 2009. → pages 48

[67] C. Fellbaum. WordNet - An Electronic Lexical Database. Cambridge, MA, 1998. MIT Press. → pages 122

[68] D. Feng, J. Kim, E. Shaw, and E. Hovy. Towards modeling threaded discussions using induced ontology knowledge. In *proceedings of the 21st national conference on Artificial intelligence - Volume 2*, AAAI'06, pages 1289–1294. AAAI Press, 2006. ISBN 978-1-57735-281-5. URL http://dl.acm.org/citation.cfm?id=1597348.1597393. → pages 46

[69] V. Feng and G. Hirst. Text-level Discourse Parsing with Rich Linguistic Features. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics*, ACL '12, pages 60–68, Jeju Island, Korea, 2012. ACL. → pages 21, 31, 103, 108

[70] O. Ferschke, I. Gurevych, and Y. Chebotar. Behind the Article: Recognizing Dialog Acts in Wikipedia Talk Pages. In *Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics*, EACL '12, pages 777–786, Avignon, France, 2012. ACL. → pages 151

[71] J. Finkel, A. Kleeman, and C. Manning. Efficient, Feature-based, Conditional Random Field Parsing. In *Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics*, ACL'08, pages 959–967, Columbus, Ohio, USA, 2008. ACL. → pages 113

[72] S. Fisher and B. Roark. The Utility of Parse-derived Features for Automatic Discourse Segmentation. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics*, ACL'07, pages 488–495, Prague, Czech Republic, 2007. ACL. → pages 19, 102, 108, 126, 128, 130, 132, 134, 135

[73] M. Galley and K. McKeown. Improving Word Sense Disambiguation in Lexical Chaining. In *Proceedings of the 18th International Joint Conference on Artificial Intelligence*, IJCAI'03, pages 1486–1488, Acapulco, Mexico, 2003. → pages 121

[74] M. Galley, K. McKeown, E. Fosler-Lussier, and H. Jing. Discourse Segmentation of Multi-party Conversation. In *Proceedings of the 41st Annual Meeting on Association for Computational Linguistics - Volume 1*, ACL '03, pages 562–569, Sapporo, Japan, 2003. ACL. → pages 9, 10, 11, 13, 32, 38, 43, 44, 50, 60, 62, 64, 65

[75] S. Ghosh, R. Johansson, G. Riccardi, and S. Tonelli. Shallow Discourse Parsing with Conditional Random Fields. In *Proceedings of the 5th International Joint Conference on Natural Language Processing*, IJCNLP'11, pages 1071—1079, Chiang Mai, Thailand, 2011. AFNLP. → pages 31, 113

[76] T. L. Griffiths, M. Steyvers, D. M. Blei, and J. B. Tenenbaum. Integrating topics and syntax. In *Advances in Neural Information Processing Systems*, NIPS'05, pages 537–544. MIT Press, 2005. → pages 52

[77] A. Gruenstein, J. Niekrasz, and M. Purver. Meeting structure annotation – annotations collected with a general purpose toolkit. *In Recent Trends in Discourse and Dialogue*, 39:247–274, 2008. → pages 11

[78] A. D. Haghighi, A. Y. Ng, and C. D. Manning. Robust textual inference via graph matching. In *Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing*, HLT '05, pages 387–394, Vancouver, British Columbia, Canada, 2005. Association for Computational Linguistics. → pages 28

[79] M. Halliday and R. Hasan. *Cohesion in English*. Longman, London, 1976. → pages 5

[80] S. Harabagiu and F. Lacatusu. Topic Themes for Multi-document Summarization. In *Proceedings of the 28th annual international ACM*

*SIGIR conference on Research and development in information retrieval*, pages 202–209, Salvador, Brazil, 2005. ACM. → pages 39, 168

[81] C. Hardmeier. Discourse in statistical machine translation: A survey and a case study. *Discours*, 2012. → pages 171

[82] C. Hardmeier, J. Nivre, and J. Tiedemann. Document-wide decoding for phrase-based statistical machine translation. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, EMNLP-CoNLL '12, pages 1179–1190, Jeju Island, Korea, 2012. Association for Computational Linguistics. → pages 171

[83] M. A. Hearst. TextTiling: Segmenting Text into Multi-paragraph Subtopic Passages. *Computational Linguistics*, 23(1):33–64, March 1997. ISSN 0891-2017. → pages 8, 9, 43, 51, 62

[84] H. Hernault, H. Prendinger, D. duVerle, and M. Ishizuka. HILDA: A Discourse Parser Using Support Vector Machine Classification. *Dialogue and Discourse*, 1(3):1—33, 2010. → pages 19, 21, 103, 108, 109, 117, 119, 125, 134, 135, 136

[85] T. Hirao, , J. Suzuki, H. Isozaki, and E. Maeda. Dependency-based Sentence Alignment for Multiple Document Summarization. In *Proceedings of the 20th international conference on computational linguistics*, COLING'04, pages 446–452, Geneva, Switzerland, 2004. ACL. → pages 149, 157

[86] G. Hirst and D. St-Onge. Lexical Chains as Representation of Context for the Detection and Correction of Malapropisms. In *Christiane Fellbaum, editor, WordNet: An Electronic Lexical Database and Some of its Applications*, pages 305–332. MIT press, 1997. → pages 121

[87] J. Hobbs. Coherence and coreference. *Cognitive Science*, 3:67–90, 1979. ISSN 1. → pages 5

[88] E. Hovy, C. Y. Lin, and L. Zhou. A BE-based Multi-document Summarizer with Query Interpretation. In *Proceedings of Document Understanding Conference*, DUC'05, Vancouver, Canada, 2005. → pages 149, 157

[89] E. Hovy, C. Y. Lin, L. Zhou, and J. Fukumoto. Automated Summarization Evaluation with Basic Elements. In *Proceedings of the Fifth Conference on Language Resources and Evaluation*, LREC'06, Genoa, Italy, 2006. → pages 157

[90] P. Hsueh, J. D. Moore, and S. Renals. Automatic Segmentation of Multiparty Dialogue. In *the Proceedings of the 11th Conference of the European Chapter of the Association for Computational Linguistics*, EACL'06, Trento, Italy, 2006. ACL. → pages 44

[91] A. Hulth. Improved Automatic Keyword Extraction Given More Linguistic Knowledge. In *Proceedings of the 2003 conference on Empirical methods in natural language processing*, EMNLP '03, pages 216–223. ACL, 2003. → pages 16, 47, 75

[92] A. Janin, D. Baron, J. Edwards, D. Ellis, D. Gelbart, N. Morgan, B. Peskin, T. Pfau, E. Shriberg, A. Stolcke, and C. Wooters. The ICSI Meeting Corpus. In *Proceedings of IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP-03)*, pages 364–367, 2003. → pages 11, 38

[93] M. Jeong, C.-Y. Lin, and G. G. Lee. Semi-supervised speech act recognition in emails and forums. In *Proceedings of the 2009 conference on Empirical methods in natural language processing*, EMNLP'09, 2009. → pages 25, 150, 151, 153, 157

[94] S. Joty, G. Carenini, G. Murray, and R. T. Ng. Exploiting Conversation Structure in Unsupervised Topic Segmentation for Emails. In *Proceedings of the conference on Empirical Methods in Natural Language Processing*, EMNLP'10, pages 388–398, Massachusetts, USA, 2010. ACL. → pages 37, 56

[95] S. Joty, G. Carenini, and C.-Y. Lin. Unsupervised Modeling of Dialog Acts in Asynchronous Conversations. In *Proceedings of the twenty second International Joint Conference on Artificial Intelligence*, IJCAI'11, Barcelona, 2011. → pages 49, 55, 146

[96] S. Joty, G. Carenini, G. Murray, and R. T. Ng. Supervised Topic Segmentation of Email Conversations. In *Proceedings of the Fifth International AAAI Conference on Weblogs and Social Media*, ICWSM'11, pages 530–533, Barcelona, Spain, 2011. AAAI. → pages 37, 65

[97] S. Joty, G. Carenini, and R. T. Ng. A Novel Discriminative Framework for Sentence-Level Discourse Analysis. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, EMNLP-CoNLL '12, pages 904–915, Jeju Island, Korea, 2012. ACL. → pages 100

[98] S. Joty, G. Carenini, and R. T. Ng. Topic Segmentation and Labeling in Asynchronous Conversations. *Journal of Artificial Intelligence Research (JAIR)*, 47:521–573, 2013. → pages 37

[99] S. Joty, G. Carenini, R. T. Ng, and Y. Mehdad. Combining Intra- and Multi-sentential Rhetorical Parsing for Document-level Discourse Analysis. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics*, ACL '13, Sofia, Bulgaria, 2013. ACL. → pages 100

[100] D. Jurafsky and J. Martin. *Speech and Language Processing*, chapter 14. Prentice Hall, 2008. → pages 4, 31, 124

[101] S. Kim, T. Baldwin, and M. Kan. Evaluating N-gram Based Evaluation Metrics for Automatic Keyphrase Extraction. In *Proceedings of the 23rd International Conference on Computational Linguistics*, COLING'10, pages 572–580, Beijing, China, 2010. ACL. → pages 83

[102] S. N. Kim, L. Cavedon, and T. Baldwin. Classifying Dialogue Acts in One-on-one Live Chats. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, EMNLP'10. ACL, 2010. → pages 23, 31, 147, 150, 156

[103] S. N. Kim, O. Medelyan, M.-Y. Kan, and T. Baldwin. SemEval-2010 Task 5 : Automatic Keyphrase Extraction from Scientific Articles. In *Proceedings of the 5th International Workshop on Semantic Evaluation*, pages 21–26, Uppsala, Sweden, July 2010. ACL. → pages 92

[104] T. Kleinbauer, S. Becker, and T. Becker. Combining Multiple Information Layers for the Automatic Generation of Indicative Meeting Abstracts. In *Proceedings of the Eleventh European Workshop on Natural Language Generation*, ENLG'07, pages 151–154, Stroudsburg, PA, USA, 2007. ACL. → pages 8, 39, 168

[105] J. M. Kleinberg. Authoratative sources in a hyperlinked environment. In *ACM*, volume 46(5), 1999. → pages 29

[106] A. Knott and R. Dale. Using Linguistic Phenomena to Motivate a Set of Coherence Relations. *Discourse Processes*, 18(1):35–62, 1994. → pages 119

[107] J. Kolář. A Comparison of Language Models for Dialog Act Segmentation of Meeting Transcripts. In *Proceedings of the 11th international*

*conference on Text, Speech and Dialogue*, TSD '08, pages 117–124, Berlin, Heidelberg, 2008. Springer-Verlag. → pages 23, 147

[108] D. Koller and N. Friedman. *Probabilistic Graphical Models Principles and Techniques*. The MIT Press, 2009. → pages 3, 29

[109] B. Krishnapuram, L. Carin, M. A. T. Figueiredo, S. Member, and E. J. Hartemink. Sparse Multinomial Logistic Regression: Fast Algorithms and Reneralization Bounds. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27:957–968, 2005. ISSN 6. → pages 61

[110] J. Lafferty, A. McCallum, and F. Pereira. Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data. In *Proceedings of the Eighteenth International Conference on Machine Learning*, pages 282–289, San Francisco, CA, USA, 2001. Morgan Kaufmann Publishers Inc. → pages 10, 113, 129

[111] J. Lau, K. Grieser, D. Newman, and T. Baldwin. Automatic Labelling of Topic Models. In *Proceedings of the 49th annual meeting on Association for Computational Linguistics*, ACL'11, pages 1536–1545, Portland, USA, 2011. ACL. → pages 46, 68

[112] A. Lazaridou, I. Titov, and C. Sporleder. A Bayesian Model for Joint Unsupervised Induction of Sentiment, Aspect and Discourse Representations. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics*, ACL '13, Sofia, Bulgaria, 2013. ACL. → pages 18, 101, 169

[113] H. LeThanh, G. Abeysinghe, and C. Huyck. Generating Discourse Structures for Written Texts. In *Proceedings of the 20th international conference on Computational Linguistics*, COLING '04, Geneva, Switzerland, 2004. ACL. doi:10.3115/1220355.1220403. URL http://dx.doi.org/10.3115/1220355.1220403. → pages 19

[114] C.-Y. Lin. ROUGE: A Package for Automatic Evaluation of Summaries. In *Proceedings of Workshop on Text Summarization Branches Out*, pages 74–81, Barcelona, 2004. → pages 83

[115] C. Y. Lin and E. Hovy. The Automated Acquisition of Topic Signatures for Text Summarization. In *Proceedings of the 18th conference on Computational linguistics*, pages 495–501. ACL, 2000. → pages 168

[116] W.-H. Lin, E. Xing, and A. Hauptmann. A joint topic and perspective model for ideological discourse. In *Proceedings of the European conference on Machine Learning and Knowledge Discovery in Databases - Part II*, ECML PKDD '08, pages 17–32, Antwerp, Belgium, 2008. Springer-Verlag. ISBN 978-3-540-87480-5. → pages 170

[117] S. Liu, M. X. Zhou, S. Pan, Y. Song, W. Qian, W. Cai, and X. Lian. TIARA: Interactive, Topic-based Visual Text Summarization and Analysis. *ACM Trans. Intell. Syst. Technol.*, 3(2):25:1–25:28, Feb. 2012. ISSN 2157-6904. → pages 39, 170

[118] A. Louis, A. Joshi, and A. Nenkova. Discourse Indicators for Content Selection in Summarization. In *Proceedings of the 11th Annual Meeting of the Special Interest Group on Discourse and Dialogue*, SIGDIAL '10, pages 147–156, Tokyo, Japan, 2010. ACL. → pages 18, 101, 169, 172

[119] W. Magdy. Tweetmogaz: a news portal of tweets. In *Proceedings of the 36th international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '13, pages 1095–1096. ACM, 2013. → pages 2

[120] D. Magerman. Statistical Decision-tree Models for Parsing. In *Proceedings of the 33rd annual meeting on Association for Computational Linguistics*, ACL'95, pages 276–283, Cambridge, Massachusetts, 1995. ACL. → pages 135

[121] I. Malioutov and R. Barzilay. Minimum Cut Model for Spoken Lecture Segmentation. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics*, COLING-ACL'06, pages 25–32, Sydney, Australia, 2006. ACL. → pages 9, 10, 28, 44, 57, 86, 156

[122] W. Mann and S. Thompson. Rhetorical Structure Theory: Toward a Functional Theory of Text Organization. *Text*, 8(3):243–281, 1988. → pages 17, 101, 131

[123] C. D. Manning, P. Raghavan, and H. Schutze. *Introduction to Information Retrieval*. Cambridge University Press, 2008. → pages 46

[124] D. Marcu. A Decision-based Approach to Rhetorical Parsing. In *Proceedings of the 37th annual meeting of the Association for Computational Linguistics on Computational Linguistics*, ACL'99, pages 365–372, Morristown, NJ, USA, 1999. ACL. → pages 20, 106, 109

[125] D. Marcu. The Rhetorical Parsing of Unrestricted Texts: A Surface-based Approach. *Computational Linguistics*, 26:395–448, 2000. → pages 20, 105, 119

[126] D. Marcu. *The Theory and Practice of Discourse Parsing and Summarization*. MIT Press, Cambridge, MA, USA, 2000. → pages 18, 101, 125, 132, 133, 169, 172

[127] D. Marcu and A. Echihabi. An Unsupervised Approach to Recognizing Discourse Relations. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, ACL'02, pages 368–375. ACL, 2002. → pages 20, 105, 106

[128] M. Marcus, B. Santorini, and M. Marcinkiewicz. Building a Large Annotated Corpus of English: The Penn Treebank. *Computational Linguistics*, 19(2):313–330, 1994. → pages 131

[129] J. Martin. *English Text: System and Structure*. John Benjamins, Philadelphia/Amsterdam, 1992. → pages 101

[130] M. Maslennikov and T.-S. Chua. A multi-resolution framework for information extraction from free text. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 592–599, Prague, Czech Republic, June 2007. Association for Computational Linguistics. URL http://www.aclweb.org/anthology/P07-1075. → pages 18, 170

[131] E. Mayfield, D. Adamson, and C. P. Rosé. Hierarchical Conversation Structure Prediction in Multi-party Chat. In *Proceedings of the 13th Annual Meeting of the Special Interest Group on Discourse and Dialogue*, SIGDIAL '12, pages 60–69. ACL, 2012. → pages 23, 26, 48

[132] A. McCallum. MALLET: A Machine Learning for Language Toolkit. http://mallet.cs.umass.edu, 2002. → pages 117

[133] K. Mckeown, J. Hirschberg, M. Galley, and S. Maskey. From Text to Speech Summarization. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing*, ICASSP'05, 2005. → pages 169

[134] O. Medelyan. *Human-Competitive Automatic Topic Indexing*. PhD thesis, The University of Waikato, Hamilton, New Zealand, 2009. → pages 47, 68, 75

[135] O. Medelyan, E. Frank, and I. H. Witten. Human-Competitive Tagging using Automatic Keyphrase Extraction. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*, EMNLP'09, pages 1318–1327, Singapore, 2009. ACL. → pages 16, 47, 83, 91

[136] Y. Mehdad, G. Carenini, R. T. Ng, and S. Joty. Towards topic labeling with phrase entailment and aggregation. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 179–189, Atlanta, Georgia, June 2013. Association for Computational Linguistics. → pages 96, 169

[137] Q. Mei, X. Shen, and C. Zhai. Automatic labeling of Multinomial topic models. In *Proceedings of the 13th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 490–499, California, USA, 2007. ACM. → pages 16, 45, 67, 68, 71, 75

[138] R. Mihalcea. Unsupervised large-vocabulary word sense disambiguation with graph-based algorithms for sequence data labeling. In *Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing*, HLT '05, pages 411–418, Vancouver, British Columbia, Canada, 2005. Association for Computational Linguistics. doi:10.3115/1220575.1220627. URL http://dx.doi.org/10.3115/1220575.1220627. → pages 28

[139] R. Mihalcea and D. Radev. *Graph-based Natural Language Processing and Information Retrieval*. Cambridge University Press, 2011. → pages 3, 16, 17, 29, 39, 47, 68

[140] R. Mihalcea and P. Tarau. TextRank: Bringing Order into Text. In *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing*, EMNLP'04, pages 404–411, Barcelona, Spain, 2004. → pages 16, 28, 29, 47, 69, 71, 74, 83, 91

[141] T. Minka. The Dirichlet-tree Distribution. Technical report, Justsystem Pittsburgh Research Center, 1999. URL http://research.microsoft.com/~minka/papers/dirichlet/minkadirtree.pdf. → pages 58

[142] Y. Mizuta, A. Korhonen, T. Mullen, and N. Collier. Zone analysis in biology articles as a basis for information extraction. *I. J. Medical Informatics*, pages 468–487, 2006. → pages 170

[143] J. Morris and G. Hirst. Lexical Cohesion Computed by Thesaural Relations as an Indicator of Structure of Text. *Computational Linguistics*, 17(1): 21–48, 1991. → pages 9, 43, 50, 121

[144] J. Mu, K. Stegmann, E. Mayfiled, C. Rosé, and F. Fischer. The acodea framework: Developing segmentation and classification schemes for fully automatic analysis of online discussions. *International Journal Of Computer-supported Collaborative Learning*, 7(2):285–305, 2012. → pages 171

[145] K. Murphy. *Machine Learning A Probabilistic Perspective*. The MIT Press, 2012. → pages 30, 31, 89, 113

[146] G. Murray and G. Carenini. Summarizing Spoken and Written Conversations. In *Proceedings of EMNLP*, Honolulu, Hawaii, 2008. → pages 169

[147] G. Murray and S. Renals. Detecting Action Items in Meetings. In *Proceedings of the 5th international workshop on Machine Learning for Multimodal Interaction*, MLMI '08, pages 208–213, Utrecht, The Netherlands, 2008. Springer-Verlag. → pages 170

[148] G. Murray, S. Renals, J. Carletta, and J. Moore. Incorporating Speaker and Discourse Features into Speech Summarization. In *Proceedings of the Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics*, HLT-NAACL'06, 2006. → pages 22, 147, 168

[149] G. Murray, T. Kleinbauer, P. Poller, T. Becker, S. Renals, and J. Kilgour. Extrinsic Summarization Evaluation: A Decision Audit Task. *ACM Trans. Speech Lang. Process.*, 6:2:1–2:29, October 2009. ISSN 1550-4875. → pages 169

[150] G. Murray, G. Carenini, and R. Ng. Interpretation and Transformation for Abstracting Conversations. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, HLT '10, pages 894–902, Los Angeles, California, 2010. ACL. → pages 169

[151] G. Murray, G. Carenini, and R. T. Ng. Generating and Validating Abstracts of Meeting Conversations: a User Study. In *Proceedings of the 6th International Natural Language Generation Conference*, INLG'10, 2010. → pages 22, 147, 168, 169

[152] D. Nguyen, E. Mayfield, and C. P. Rosé. An analysis of perspectives in interactive settings. In *Proceedings of the First Workshop on Social Media Analytics*, SOMA '10, pages 44–52, Washington D.C., District of Columbia, 2010. ACM. → pages 170

[153] V.-A. Nguyen, J. Boyd-Graber, and P. Resnik. SITS: A Hierarchical Nonparametric Model using Speaker Identity for Topic Segmentation in Multiparty Conversations. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Long Papers - Volume 1*, ACL '12, pages 78–87, Jeju Island, Korea, 2012. ACL. → pages 10, 45

[154] J. Otterbacher, G. Erkan, and D. R. Radev. Using Random Walks for Question-focused Sentence Retrieval. In *Proceedings of Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing*, pages 915–922, Vancouver, Canada, 2005. → pages 28, 29

[155] L. Page, S. Brin, R. Motwani, and T. Winograd. The PageRank Citation Ranking: Bringing Order to the Web. Technical Report 1999-66, Stanford InfoLab, 1999. → pages 16, 46, 69, 72

[156] S. Pan, M. X. Zhou, Y. Song, W. Qian, F. Wang, and S. Liu. Optimizing Temporal Topic Segmentation for Intelligent Text Visualization. In *Proceedings of the 2013 international conference on Intelligent user interfaces*, IUI '13, pages 339–350, Santa Monica, California, USA, 2013. ACM. → pages 170

[157] B. Pang and L. Lee. A sentimental education: sentiment analysis using subjectivity summarization based on minimum cuts. In *Proceedings of the 42nd Annual Meeting on Association for Computational Linguistics*, ACL '04, Barcelona, Spain, 2004. Association for Computational Linguistics. → pages 28, 170

[158] K. Papineni, S. Roukos, T. Ward, and W.-J. Zhu. BLEU: A Method for Automatic Evaluation of Machine Translation. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, ACL'02, pages 311–318, Philadelphia, Pennsylvania, 2002. ACL. → pages 83

[159] R. J. Passonneau and D. J. Litman. Discourse Segmentation by Human and Automated Means. *Computational Linguistics*, 23(1):103–139, Mar. 1997. ISSN 0891-2017. → pages 60

[160] M. J. Paul. Mixed Membership Markov Models for Unsupervised Conversation Modeling. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, EMNLP-CoNLL '12, pages 94–104, Stroudsburg, PA, USA, 2012. ACL. → pages 152

[161] T. Pedersen, S. Patwardhan, and J. Michelizzi. WordNet::Similarity - Measuring the Relatedness of Concepts. In *Proceedings of Fifth Annual Meeting of the North American Chapter of the Association for Computational Linguistics (NAACL-04)*, pages 38–41, Boston, MA, 2004. → pages 84

[162] L. Pevzner and M. A. Hearst. A Critique and Improvement of an Evaluation Metric for Text Segmentation. *Computational Linguistics*, 28 (1):19–36, Mar. 2002. ISSN 0891-2017. → pages 81

[163] R. Prasad, A. Joshi, N. Dinesh, A. Lee, E. Miltsakaki, and B. Webber. The Penn Discourse TreeBank as a Resource for Natural Language Generation. In *Proceedings of the Corpus Linguistics Workshop on Using Corpora for Natural Language Generation*, pages 25–32, Birmingham, U.K., 2005. → pages 18, 101

[164] R. Prasad, N. Dinesh, A. Lee, E. Miltsakaki, L. Robaldo, A. Joshi, and B. Webber. The Penn Discourse TreeBank 2.0. In *Proceedings of the Sixth International Conference on Language Resources and Evaluation (LREC)*, pages 2961—2968, Marrakech, Morocco, 2008. ELRA. ISBN 2-9517408-4-0. → pages x, 31, 113

[165] M. Purver. Topic Segmentation. In G. Tur and R. de Mori, editors, *Spoken Language Understanding: Systems for Extracting Semantic Information from Speech*, pages 291–317. Wiley, 2011. ISBN 978-0-470-68824-3. → pages 9, 11, 38, 43, 50, 77

[166] M. Purver, P. Ehlen, and J. Niekrasz. Detecting Action Items in Multi-party Meetings: Annotation and Initial Experiments. In *Proceedings of the Third international conference on Machine Learning for Multimodal Interaction*, MLMI'06, pages 200–211, Bethesda, MD, 2006. Springer-Verlag. → pages 170

[167] M. Purver, K. P. Kording, T. L. Griffiths, and J. B. Tenenbaum. Unsupervised Topic Modelling for Multi-Party Spoken Discourse. In *Proceedings of the 21st International Conference on Computational*

*Linguistics and the 44th annual meeting of the Association for Computational Linguistics*, COLING-ACL'06, pages 17–24, Sydney, Australia, 2006. ACL. → pages 9, 10, 45

[168] L. R. Rabiner. A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition. In *Proceedings of the IEEE*, pages 257–285, 1989. → pages 212

[169] O. Rambow, L. Shrestha, J. Chen, and C. Lauridsen. Summarizing Email Threads. In *Proceedings of the Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics*, HLT-NAACL-Short '04, pages 105–108. ACL, 2004. → pages 169

[170] R. Ranganath, D. Jurafsky, and D. Mcfarland. It's Not You, it's Me: Detecting Flirting and its Misperception in Speed-Dates. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*, EMNLP'09, pages 334–342, Singapore, 2009. ACL. → pages 147

[171] V. K. Rangarajan Sridhar, S. Bangalore, and S. Narayanan. Combining Lexical, Syntactic and Prosodic Cues for Improved Online Dialog Act Tagging. *Comput. Speech Lang.*, 23:407–422, October 2009. ISSN 0885-2308. → pages 147

[172] S. Rashid. A Visual Interface for Browsing and Summarizing Conversations. Master's thesis, University of British Columbia, Vancouver, 2012. → pages 170

[173] S. Ravi and J. Kim. Profiling Student Interactions in Threaded Discussions with Speech Act Classifiers. In *Proceedings of AI in Education Conference*, AIED'07, 2007. → pages 151

[174] A. Ritter, C. Cherry, and B. Dolan. Unsupervised Modeling of Twitter Conversations. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, HLT '10, pages 172–180, LA, California, 2010. ACL. → pages 25, 35, 150, 152, 156, 159, 160

[175] C. Rosé, Y. chia Wang, J. Arguello, K. Stegmann, A. Weinberger, and F. Fischer. Analyzing collaborative learning processes automatically: Exploiting the advances of computational linguistics in computer-supported

collaborative learning. *International Journal Of Computer-supported Collaborative Learning*, 3(3):237–271, 2008. → pages 171

[176] C. P. Rosé, B. Di Eugenio, L. S. Levin, and C. Van Ess-Dykema. Discourse processing of dialogues with multiple threads. In *Proceedings of the 33rd annual meeting on Association for Computational Linguistics*, ACL '95, pages 31–38, Stroudsburg, PA, USA, 1995. Association for Computational Linguistics. doi:10.3115/981658.981663. URL http://dx.doi.org/10.3115/981658.981663. → pages 26

[177] H. Sacks, A. Schegloff, and G. Jefferson. A Simplest Systematics for the Organization of Turn-taking for Conversation. *Language*, 50:696—735, 1974. → pages 22, 78

[178] G. Salton and M. J. McGill. *Introduction to Modern Information Retrieval*. McGraw-Hill, Inc., New York, NY, USA, 1986. ISBN 0070544840. → pages 9, 62, 71, 156

[179] H. Schauer and U. Hahn. Anaphoric Cues for Coherence Relations. In *Proceedings of the Conference on Recent Advances in Natural Language Processing*, RANLP '01, pages 228–234, 2001. → pages 105

[180] A. Schegloff. Sequencing in conversational openings1. *American Anthropologist*, 70(6):1075–1095, 1968. ISSN 1548-1433. → pages 22, 147

[181] F. Schilder. Robust Discourse Parsing via Discourse Markers, Topicality and Position. *Natural Language Engineering*, 8(3):235–255, June 2002. ISSN 1351-3249. → pages 141

[182] E. Seneta. *Non-negative Matrices and Markov Chains*. Springer-Verlag, 1981. → pages 72

[183] F. Sha and F. Pereira. Shallow Parsing with Conditional Random Fields. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology - Volume 1*, NAACL-HLT'03, pages 134–141, Edmonton, Canada, 2003. ACL. → pages 113

[184] D. Shen, Q. Yang, J.-T. Sun, and Z. Chen. Thread detection in dynamic text message streams. In *Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '06, pages 35–42, Seattle, Washington, USA, 2006. ACM. ISBN 1-59593-369-7. → pages 23, 26, 47

[185] J. Shi and J. Malik. Normalized Cuts and Image Segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(8): 888–905, 2000. ISSN 0162-8828. → pages 28, 44, 56, 57

[186] L. Shrestha and K. McKeown. Detection of Question-Answer Pairs in Email Conversations. In *Proceedings of the 20th international conference on Computational Linguistics*, COLING '04, Morristown, NJ, USA, 2004. ACL. → pages 151

[187] H. G. Silber and K. F. McCoy. Efficiently Computed Lexical Chains As an Intermediate Representation for Automatic Text Summarization. *Computational Linguistics*, 28(4):487–496, 2002. → pages 121

[188] N. A. Smith. *Linguistic Structure Prediction*. Synthesis Lectures on Human Language Technologies. Morgan and Claypool, May 2011. → pages 21

[189] S. Somasundaran. *Discourse-Level Relations for Opinion Analysis*. PhD thesis, University of Pittsburgh, Pittsburgh, 2010. → pages 18, 101, 169

[190] W. M. Soon, H. T. Ng, and D. C. Y. Lim. A Machine Learning Approach to Coreference Resolution of Noun Phrases. *Computational Linguistics*, 27 (4):521–544, Dec. 2001. ISSN 0891-2017. → pages 28, 60, 156

[191] R. Soricut and D. Marcu. Sentence Level Discourse Parsing Using Syntactic and Lexical Information. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology - Volume 1*, NAACL'03, pages 149–156, Edmonton, Canada, 2003. ACL. → pages 19, 21, 103, 105, 107, 110, 111, 117, 119, 125, 126, 128, 132, 134, 135, 136, 139

[192] C. Sporleder. Manually vs. Automatically Labelled Data in Discourse Relation Classification. Effects of Example and Feature Selection. *LDV Forum*, 22(1):1–20, 2007. → pages 106

[193] C. Sporleder and M. Lapata. Automatic Paragraph Identification: A Study across Languages and Domains. In *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing*, EMNLP '04, pages 72–79, 2004. → pages 105, 110, 121, 123

[194] C. Sporleder and M. Lapata. Discourse Chunking and its Application to Sentence Compression. In *Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing*, HLT-EMNLP'05, pages 257–264, Vancouver, British Columbia, Canada, 2005. ACL. → pages 18, 19, 101, 107, 117, 128, 130

[195] C. Sporleder and A. Lascarides. Exploiting Linguistic Cues to Classify Rhetorical Relations. In *Proceedings of Recent Advances in Natural Langauge Processing (RANLP)*, Bulgaria, 2005. → pages 20, 106

[196] C. Sporleder and A. Lascarides. Using Automatically Labelled Examples to Classify Rhetorical Relations: An Assessment. *Natural Language Engineering*, 14(3):369–416, 2008. ISSN 1351-3249. → pages 20, 105, 106

[197] M. Stede. The Potsdam Commentary Corpus. In *Proceedings of the ACL-04 Workshop on Discourse Annotation*, Barcelona, 2004. ACL. → pages 105, 140

[198] M. Stede. *Discourse Processing*. Synthesis Lectures on Human Language Technologies. Morgan And Claypool Publishers, 2011. → pages 7, 19, 20, 104, 128

[199] M. Steyvers and T. Griffiths. *Latent Semantic Analysis: A Road to Meaning*, chapter Probabilistic Topic Models. Laurence Erlbaum, 2007. → pages 51, 86

[200] A. Stolcke, N. Coccaro, R. Bates, P. Taylor, C. Van Ess-Dykema, K. Ries, E. Shriberg, D. Jurafsky, R. Martin, and M. Meteer. Dialogue Act Modeling for Automatic Tagging and Recognition of Conversational Speech. *Computational Linguistics*, 26:339–373, 2000. → pages 23, 31, 147

[201] R. Subba and B. Di-Eugenio. An Effective Discourse Parser that Uses Rich Linguistic Information. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, HLT-NAACL'09, pages 566–574, Boulder, Colorado, 2009. ACL. → pages 21, 103, 105, 108, 109, 125, 126, 131, 136, 137, 139, 140

[202] A. Subramanya and J. Bilmes. Semi-supervised learning with measure propagation. *J. Mach. Learn. Res.*, 12:3311–3370, Nov. 2011. ISSN 1532-4435. → pages 28

[203] A. Subramanya, S. Petrov, and F. Pereira. Efficient graph-based semi-supervised learning of structured tagging models. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, EMNLP '10, pages 167–176, Cambridge, Massachusetts, 2010. Association for Computational Linguistics. → pages 28

[204] C. Sutton and A. McCallum. An Introduction to Conditional Random Fields. *Foundations and Trends in Machine Learning*, 4(4):267–373, 2012. → pages 30, 116

[205] C. Sutton, A. McCallum, and K. Rohanimanesh. Dynamic Conditional Random Fields: Factorized Probabilistic Models for Labeling and Segmenting Sequence Data. *Journal of Machine Learning Research (JMLR)*, 8:693–723, 2007. ISSN 1532-4435. → pages 34, 103, 112

[206] M. Taboada. Discourse Markers as Signals (or Not) of Rhetorical Relations. *Journal of Pragmatics*, 38(4):567–592, 2006. → pages 105

[207] M. Taboada, J. Brooke, M. Tofiloski, K. Voll, and M. Stede. Lexicon-based methods for sentiment analysis. *Comput. Linguist.*, 37(2):267–307, June 2011. ISSN 0891-2017. → pages 170

[208] P. P. Talukdar and K. Crammer. New regularized algorithms for transductive learning. In *Proceedings of the European Conference on Machine Learning and Knowledge Discovery in Databases: Part II*, ECML PKDD '09, pages 442–457, Bled, Slovenia, 2009. Springer-Verlag. → pages 28, 29

[209] S. Teufel and M. Moens. Summarizing scientific articles: experiments with relevance and rhetorical status. *Comput. Linguist.*, 28(4):409–445, Dec. 2002. ISSN 0891-2017. → pages 18, 170

[210] M. Tofiloski, J. Brooke, and M. Taboada. A syntactic and lexical-based discourse segmenter. In *Proceedings of the ACL-IJCNLP 2009 Conference Short Papers*, ACLShort '09, pages 77–80, Stroudsburg, PA, USA, 2009. Association for Computational Linguistics. URL http://dl.acm.org/citation.cfm?id=1667583.1667609. → pages 19

[211] A. Tumasjan, T. O. Sprenger, P. G. Sandner, and I. M. Welpe. Predicting elections with twitter: What 140 characters reveal about political sentiment. In *Proceedings of the Fourth International AAAI Conference on Weblogs and Social Media*, pages 178–185, 2010. → pages 170

[212] P. D. Turney. Learning Algorithms for Keyphrase Extraction. *Information Retrieval*, 2(4):303–336, May 2000. ISSN 1386-4564. → pages 83

[213] J. Ulrich, G. Murray, and G. Carenini. A Publicly Available Annotated Corpus for Supervised Email Summarization. In *EMAIL-2008 Workshop*, pages 428–435. AAAI, 2008. → pages xiii, 12, 77, 153

[214] I. Varga, M. Sano, K. Torisawa, C. Hashimoto, K. Ohtake, T. Kawai, J.-H. Oh, and S. D. Saeger. Aid is Out There: Looking for Help from Tweets during a Large Scale Disaster. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics*, ACL '13, Sofia, Bulgaria, 2013. ACL. → pages 1

[215] S. Verberne, L. Boves, N. Oostdijk, and P. Coppen. Evaluating Discourse-based Answer Extraction for Why-question Answering. In *Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval*, SIGIR'07, pages 735–736, Amsterdam, The Netherlands, 2007. ACM. → pages 18, 101, 170

[216] P. Verna. The Blogosphere: Colliding with Social and Mainstream Media. *eMarketer*, 2010. URL http://www.emarketer.com/Reports/All/Emarketer_2000708.aspx. → pages 1

[217] D. Vesset, B. McDonough, and M. Wardley. Worldwide business analytics software 2010-2014 forecast and 2009 vendor shares. Technical report, idc, Stanford InfoLab, 2010. → pages 2

[218] N. Vliet and G. Redeker. Complex Sentences as Leaky Units in Discourse Parsing. In *Proceedings of Constraints in Discourse*, Agay-Saint Raphael, September 2011. → pages 104, 126, 140

[219] H. M. Wallach. Topic Modeling: Beyond Bag-of-Words. In *Proceedings of the 23rd international conference on Machine learning*, ICML '06, pages 977–984, Pittsburgh, Pennsylvania, 2006. ACM. → pages 52

[220] H. Wang, C. Wang, C. Zhai, and J. Han. Learning Online Discussion Structures by Conditional Random Fields. In *Proceedings of the 34th international ACM SIGIR conference on Research and development in Information Retrieval*, SIGIR '11, pages 435–444, Beijing, China, 2011. ACM. → pages 26, 48, 155

[221] L. Wang and C. Cardie. Unsupervised Topic Modeling Approaches to Decision Summarization in Spoken Meetings. In *Proceedings of the 13th Annual Meeting of the Special Interest Group on Discourse and Dialogue*, SIGDIAL '12, pages 40–49, Seoul, South Korea, 2012. ACL. → pages 168

[222] L. Wang and C. Cardie. Domain-Independent Abstract Generation for Focused Meeting Summarization. In *Proceedings of the 51st Annual*

*Meeting of the Association for Computational Linguistics*, ACL'13, Sofia, Bulgaria, 2013. ACL. → pages 169

[223] L. Wang and D. W. Oard. Context-based Message Expansion for Disentanglement of Interleaved Text Conversations. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, NAACL '09, pages 200–208, Boulder, Colorado, 2009. ACL. → pages 26, 47, 48

[224] Y.-C. Wang, M. Joshi, W. Cohen, and C. Rosé. Recovering Implicit Thread Structure in Newsgroup Style Conversations. In *Proceedings of the International AAAI Conference on Weblogs and Social Media*, ICWSM'08. AAAI, 2008. → pages 26, 48

[225] B. Webber. D-LTAG: Extending Lexicalized TAG to Discourse. *Cognitive Science*, 28(5):751–779, 2004. → pages 17, 101

[226] B. Webber, M. Egg, and V. Kordoni. Discourse structure and language technology. *Natural Language Engineering*, 18:437–490, 10 2012. ISSN 1469-8110. → pages 2, 7

[227] M. Wen and C. P. Rose. Understanding participant behavior trajectories in online health support groups using automatic extraction methods. In *Proceedings of the 17th ACM international conference on Supporting group work*, GROUP '12, pages 179–188, Sanibel Island, Florida, USA, 2012. ACM. → pages 171

[228] D. Widdows and B. Dorow. A graph model for unsupervised lexical acquisition. In *Proceedings of the 19th international conference on Computational linguistics - Volume 1*, COLING '02, pages 1–7, Taipei, Taiwan, 2002. Association for Computational Linguistics. doi:10.3115/1072228.1072342. URL http://dx.doi.org/10.3115/1072228.1072342. → pages 28

[229] Y. Wilks. Artificial Companions as a New Kind of Interface to the Future Internet. *OII Research Report No. 13*, 2006. → pages 147

[230] F. Wolf and E. Gibson. Representing Discourse Coherence: A Corpus-Based Study. *Computational Linguistics*, 31:249–288, June 2005. ISSN 0891-2017. → pages 143

[231] D. H. Wolpert. The lack of a priori distinctions between learning algorithms. *Neural Comput.*, 8(7):1341–1390, Oct. 1996. → pages 31

[232] T. Wu, F. M. Khan, T. A. Fisher, L. A. Shuler, and W. M. Pottenger. Posting Act Tagging Using Transformation-Based Learning. In *Proceedings of the Workshop on Foundations of Data Mining and Discovery, IEEE International Conference on Data Mining*, 2002. → pages 23, 147

[233] J. Yamron, I. Carp, L. Gillick, S. Lowe, and P. van Mulbregt. A hidden markov model approach to text segmentation and event tracking. In *Acoustics, Speech and Signal Processing, 1998. Proceedings of the 1998 IEEE International Conference on*, volume 1, pages 333–336 vol.1, 1998. doi:10.1109/ICASSP.1998.674435. → pages 9, 10

[234] T. Zesch and I. Gurevych. Approximate Matching for Evaluating Keyphrase Extraction. In *Proceedings of the 7th International Conference on Recent Advances in Natural Language Processing*, RANLP'09, pages 484–489, Borovets, Bulgaria, 2009. → pages 83

[235] R. Zhang, D. Gao, and W. Li. Towards Scalable Speech Act Recognition in Twitter: Tackling Insufficient Training Data. In *Proceedings of the Workshop on Semantic Analysis in Social Media*, pages 18–27, Avignon, France, 2012. ACL. → pages 152

[236] W. X. Zhao, J. Jiang, J. He, Y. Song, P. Achananuparp, E.-P. Lim, and X. Li. Topical Keyphrase Extraction from Twitter. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies - Volume 1*, HLT '11, pages 379–388, Portland, Oregon, 2011. ACL. → pages 46

[237] W. X. Zhao, J. Jiang, J. Weng, J. He, E.-P. Lim, H. Yan, and X. Li. Comparing Twitter and Traditional Media using Topic Models. In *Proceedings of the 33rd European conference on Advances in information retrieval*, ECIR'11, pages 338–349, Dublin, Ireland, 2011. Springer-Verlag. → pages 46

[238] D. Zhou, S. A. Orshanskiy, H. Zha, and C. L. Giles. Co-ranking Authors and Documents in a Heterogeneous Network. In *Proceedings of the 2007 Seventh IEEE International Conference on Data Mining*, ICDM '07, pages 739–744, Washington, DC, USA, 2007. IEEE Computer Society. → pages 17, 29, 73, 74

# Appendix A

# Supporting Materials

## A.1 Metrics for Topic Segmentation

### A.1.1 One-to-One Metric

Consider the two annotations of the same conversation having 10 sentences (denoted by colored boxes) in Figure A.1(a). In each annotation, the topics are distinguished by different colors. For example, the *model output* has four topics, whereas the *human annotation* has three topics. To compute one-to-one accuracy, we take the model output and map its segments optimally (by computing the optimal max-weight bipartite matching) to the segments of the gold-standard human annotation. For example, the red segment in the model output is mapped to the green segment in the human annotation. We transform the model output based on this mapping and compute the percentage of overlap as the one-to-one accuracy. In our example, seven out of ten sentences overlap, therefore, the one-to-one accuracy is 70%.

### A.1.2 $Loc_k$ Metric

Consider the model output (at the left most column) and the human annotation (at the right most column) of the same conversation having 5 sentences (denoted by colored boxes) in Figure A.2. Similar to Figure A.1, the topics in an annotation are distinguished using different colors. Suppose we want to measure the $loc_3$ score for

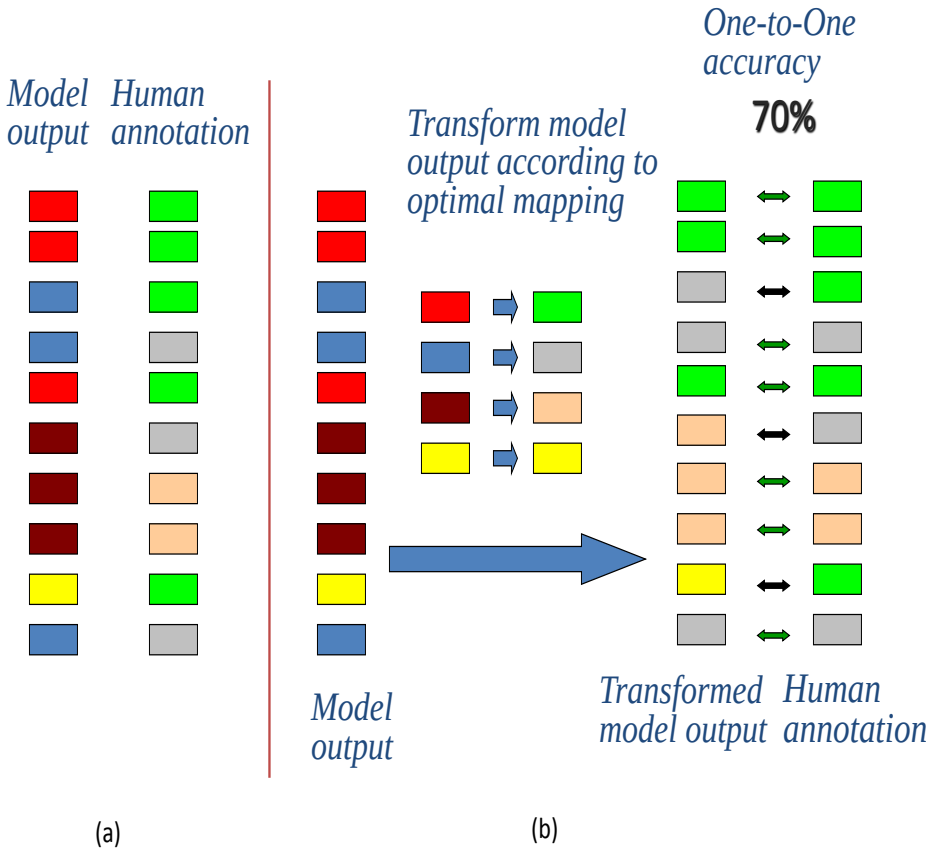



**Figure A.1:** Computing one-to-one accuracy.

the fifth sentence (marked with yellow arrows at the bottom of the two annotations). In each annotation, we look at the previous 3 sentences and transform them based on whether they have *same* or *different* topics. For example, in the model output one of the previous three sentences is *same* (red), and in the human annotation two of the previous three sentences are *same* (green), when compared with the sentence under consideration. In the transformed annotations, *same* topics are denoted by gray boxes and *different* topics are denoted by black boxes. We compute $loc_3$ by measuring the overlap of the *same* or *different* judgments in the 3-sentence window. In our example, two of three overlap, therefore, the $loc_3$ agreement is 66.6%.

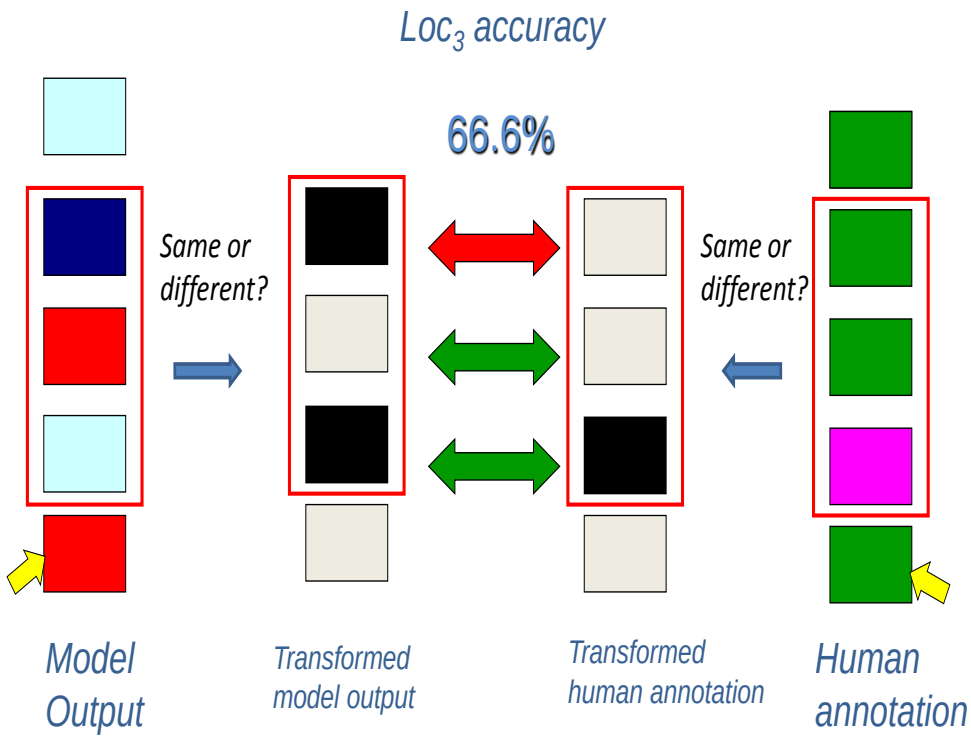



**Figure A.2:** Computing $loc_3$ accuracy.

## A.2 Annotation Manual for Topic Segmentation and Labeling

### A.2.1 Instructions for Finding Topics in Emails

#### Abstract

This document is a manual that instructs annotators on how to find topics in email conversations. After introducing some general ideas on the task, it explains how to perform the annotation step by step.

#### Introduction

The ultimate goal of this research is to be able to automatically generate summaries of email conversations. Finding topics is the first step towards this goal. It involves clustering the sentences of a conversation into a set of clusters which reveal the topics discussed in the conversation. Our assumption is that knowing the topic structure of conversations simplifies extraction of information and summarization.

#### A. Task Overview

The task of finding topics can be divided into two subtasks as described below:

*A.1. Finding the discussed topics*
You will be given a set of email conversations (or threads). Each conversation is associated with a human written summary that will give you a brief overview of the corresponding conversation. Read carefully the conversation first, then read the summary and make sure that you understand the conversation.

At this point, you should be able to find the underlying topics/issues discussed in the conversation. Our definition of 'topic' is something about which the participant(s) discuss or argue or express their opinions. For example, an email thread about an upcoming meeting may contain a discussion about the 'location and schedule', another discussion about the 'meeting agenda', etc.

Here, you need to list the topics in the following format:
< Topic number = X, a short description of the topic >

For example,

< Topic number = 1, extending the meeting duration >

< Topic number = 2, scheduling the meeting >

The short description should provide a high-level overview on that topic. This can usually be based on a few keywords from the discussion, and needs to be detailed enough that someone else could figure out later what the topic was. In order to come up with a segment description, try to fill in the following statement with a specific phrase:

In this topic, people talk about ———————————-

For example, an email thread about a meeting could have topic descriptions such as 'extending the meeting', 'scheduling the meeting', 'meeting agenda', etc. An email thread about arranging a conference can have topics such as 'location and time', 'registration', 'food menu', 'workshops', etc. An email thread about building a webpage that will contain a list of people working in a particular research area (e.g., User Modeling) all over the world, can have discussions about 'people who are working', 'creating a map', 'design of the map' and so on.

The target number of topics for each conversation **will not be given** to you in advance, so you will need to find as many topics as you see fit and natural to convey the overall content structure of the conversation. You might be expecting us to tell you exactly how many topics each thread should have, but the truth is it is subjective and varies considerably. In theory each sentence says something different from the previous sentence and therefore it should be possible to mark a new topic, but, as we mentioned previously, we **don't want this level of detail**. There could be threads that discuss one topic extensively, and others that discuss a very large number of topics, all briefly. There is also no optimal length for an topic; there is a fine, yet subjective balance as to how many topics one could detect in a single thread, before it all seems too fragmented. For this reason, you should divide the thread into topics in the way that you find **most natural**. We will provide you with some examples to help you initially.

Sometimes you won't be sure whether to mark part of a conversation as one topic or two, because there are really two topics but they are related to each other. For instance, if a group was talking about an upcoming meeting, they might talk

first about the 'location' and then they might talk about the 'time'. If they talked about these two things separately, then they would be separate topics. However, if they discuss both location and time at the same time then the topic description should be something like 'location and time'.

*A.2. Assigning topics to sentences*

Here the task is, for each sentence (which is separated by a line break) in the thread you have to identify the most appropriate topic to which it belongs. In general, one sentence should be labeled with only one topic, however if you find sentences that you think cover more than one topic, please do label them with all the relevant topics. Again, if you find any sentence that doesn't fit into any topic, just label those as the predefined topic 'OFF-TOPIC'.

Wherever appropriate you should also make use of 2 other predefined topic labels: 'INTRO' and 'END'. INTRO (e.g., 'hi', 'hello X', etc.) signifies the section (usually at the beginning) of an email that people use to begin their email. Likewise, END (e.g., 'Cheers', 'Best', etc.) signifies the section (usually at the end) that people use to end their email.

In some emails you will find people quote (usually preceded by $>$ sign) from other's email(s). You do not need to label the quoted texts if you have already labelled these texts in any of the previous emails. However, you may find some quotes that are new in the current email. You should label those. These quotes actually come from emails not in this thread (also called hidden emails). An email thread may contain three to ten emails and it may take up to 25 minutes to find the topics in a thread, so allocate enough time to be able to do so without interruptions.

*B. Examples*

To help you through the process, in this section we have included two example annotations.[1] Please study these carefully. Here, we use tab or indentation to give the thread view. If you are facing any problem with this view and prefer to have an annotation tool (i.e., software that can be used to help humans to annotate), please let us know. If you have any questions/concerns while you are performing

[1]The examples are not shown here to save space.

the annotation, do not hesitate to ask. Thanks for your help. Good luck!

### A.2.2 Instructions for Finding Topics in Blogs

#### Abstract

This document is a manual that instructs annotators on how to find topics in blog conversations and summarize them. After introducing some general ideas on the task, it explains how to perform the annotation step by step.

#### Introduction

The ultimate goal of this research is to be able to automatically generate summaries of blog conversations. Finding topics is the first step towards this goal. It involves clustering the sentences of a blog discussion into a set of clusters which reveal the topics discussed in the conversation. Our assumption is that knowing the topic structure of conversations simplifies extraction of information and summarization.

#### A. Task Overview

The task of finding topics and summarizing a blog discussion can be divided into four different subtasks which are described below:

*A.1. Writing a short summary ($\leq$ 3 sentences) of each thread:*
You will be given a set of blog discussions from Slastdot[2]. Each discussion consists of an article followed by a number of threads and single comments. For example see the sample blogs provided in pages 4 and 8 (ignore texts in color).

A **thread** is a sequence of comments organized hierarchically according to their 'reply-to' relation. A **single comment** is a comment to the article nobody replied to. The sentences in a comment were separated using an automatic segmentation tool which is not perfect in many cases (don't worry about it). At the end of each such sentence we have put a space to enter a number (i.e., Topic id ——) that will be required in step 3 (as described in A.3).

[2]http://slashdot.org/

In this initial A.1 step we ask you to first read through the article and then read carefully the threads and the single comments (ignoring [Topic id ——]). For each thread once you finish reading it, write a short summary ($\leq$3 sentences) of the thread in the space provided just below the respective thread. Your summary should be short but as informative as possible. We provide examples of what we consider good and bad short summaries in example 2 (pages 11-19). You need to read the single comments but do not need to summarize them.

To ease the process of finding topics in step 2 (Section A.2) and labelling them in step 3 (Section A.3) you can keep notes in the provided scratch paper as you discover new topics while reading through the article, threads and single comments in this step. This point will become clear once you read Sections A.2 and A.3.

Please ask the experimenter if you have any question about this step.

*A.2. Finding the discussed topics:*

At this point, as you will have read the whole blog conversation and summarized all the threads, you should have a pretty good understanding of the blog content and should be able to find the underlying topics covered in the whole discussion. Our definition of 'topic' is something about which the participant(s) discuss or argue or express their opinions. For example, a discussion about a new iphone may contain topics such as "date of arrival in market", "touch screen", "music application","charging and power", "outlook", "industrial espionage" etc. Note that even though a thread may have a single title, the sentences may discuss different topics. Even the sentences in the same comment may discuss different topics.

Here, you need to list the topics discussed in the following format:

< Topic number = X, a short description of the topic >

For example,

< Topic id 1: date of arrival >

< Topic id 2: touch screen >

< Topic id 3: music application >

And so on.

The short description should provide a high-level overview on that topic. This can usually be based on a few keywords from the discussion, but needs to be detailed enough that someone else could figure out later what the topic was. In order

to come up with a segment description, try to fill in the following statement with a specific phrase:

Here, people talked about ————————————-

For example, a discussion about different issues of a country may include "security", "economy", "personnel", "industries and companies", "foreign policy", etc. A discussion about a new scientific contribution (e.g., a proof of P is not equal to NP) may have topics such as "the inventor", "the contribution itself", "objections from other researchers", "new ideas generated by the contribution", "possible applications", "implications for theoretical computer science" and so on. See B.1 and B.2 on pages 4 and 8 for examples of topics identified in sample blogs.

The target number of topics for each discussion **will not be given** to you in advance, so you will need to find as many topics as you see fit and natural to convey the overall content structure of the discussion. You might be expecting us to tell you exactly how many topics each discussion should have, but the truth is it is rather subjective and may vary considerably. In theory each sentence says something different from the previous sentence and therefore it should be possible to mark a new topic, but, as mentioned before, we **don't want this level of detail**. There could be conversations that discuss one or two topics extensively, and others that handle a very large number of topics, all briefly. There is also no optimal length for a topic; there is a fine, yet subjective balance as to how many topics one could detect in a blog discussion, before it all seems too fragmented. For this reason, you should divide the discussion into topics in the way that you **find most natural**. We will provide you with some good and bad examples to help you initially.

Sometimes you won't be sure whether to mark part of a discussion as one topic or two, because there are really two topics but they are related to each other. For instance, if a group was talking about a country's issues, they might talk first about the "security" and then they might talk about the "economy". If they talked about these two things separately, then they would be separate topics. However, if they discuss both "security" and "economy" at the same time possibly exploring how the two can be related then a more appropriate topic description could be something like "security and economy". Studying Example 1 (Section B.1) and 2 (Section B.2) will help you with the task.

Ask the experimenter if you have any question at this point.

*A.3. Assigning topics to sentences:*

Now that you have identified the topics covered in the blog, for each sentence (which is separated by a line break) in the conversation, you have to identify the most appropriate topic to which it belongs. In general, one sentence should be labelled with only one topic, however if you find a sentence that you think cover two topics almost equally (i.e., 50-50, 40-60, 60-40), please do label them with the two relevant topics and also mention the percentage of coverage. Again, if you find any sentence that is not related to the original article, just label those as the predefined label 'OFF-TOPIC'. In step 1 (Section A.1), when summarizing the conversations if you find all the sentences in that conversation are OFF-TOPIC, you can write the summary something like: "the whole discussion is off the topic".

Wherever appropriate you should also use two other predefined topic labels: 'INTRO' and 'END'. INTRO (e.g., 'hi', 'hello X', etc.) signifies the section (usually at the beginning) of a comment that people use to begin their contribution. Likewise, END (e.g., 'Cheers', 'Best', etc.) signifies the section (usually at the end) that people use to end their comment.

In some comments you will find people quote (usually preceded by $>$ sign) from other's comment(s). You do not need to label the quoted texts if you have already labelled these texts in any of the previous comments.

Finally, while you are annotating the sentences, if you feel you need to revise your topic list (e.g., adding a new topic, renaming an existing one), please do not hesitate to do so.

At this point please ask any questions you have to the experimenter.

*A.4. Writing a 250 words summary of the whole conversation:*

As a final step you will author a single high level 250 words summary for the whole blog conversation. The summary should be around 250 words. It is therefore critical to capture the important information of the discussion in a concise manner. For example please see the 250 words summaries of the blogs on pages 4 and 8.

A blog conversation may contain thirty (30) to one hundred (100) comments and depending on the length and the number of comments it may take 40-60 minutes to complete the above four tasks, so allocate enough time to be able to do so

without interruptions.

*B. Examples:*

To help you through the process, in this section we have included two example annotations.[3] Please study these very carefully and ask questions if anything is unclear. Here, we use tab/indentation to give the conversations a thread view. Black represents the original content, Blue represents annotation, and Orange represents our comments. If you are facing any problem with this view, please let us know.

If you have any questions/concerns while you are performing the annotation, do not hesitate to ask. Thanks for your help. Good luck!

## A.3 EM for HMM+Mix model

The expected complete data log likelihood can be written as:

$$Q(\theta,\theta^{old}) = \sum_{k=1}^{K} E[N_k^1] log\pi_k + \sum_{j=1}^{K}\sum_{k=1}^{K} E[N_{jk}] logA_{jk} + \sum_{k=1}^{K}\sum_{m=1}^{M} E[N_{km}] logC_{km} + \sum_{k=1}^{K}\sum_{m=1}^{M} E[N_{kml}] logB_{kml} \tag{A.1}$$

where the expected counts are given by:

$$E[N_k^1] = \sum_{n=1}^{N} p(D_{n,1} = k|X_n,\theta^{old}) \tag{A.2}$$

$$E[N_{jk}] = \sum_{n=1}^{N}\sum_{i=1}^{T_n} p(D_{n,i} = j, D_{n,i+1} = k|X_n,\theta^{old}) \tag{A.3}$$

$$E[N_{km}] = \sum_{n=1}^{N}\sum_{i=1}^{T_n} p(D_{n,i} = k, M_{n,i} = m|X_n,\theta^{old}) \tag{A.4}$$

$$E[N_{kml}] = \sum_{n=1}^{N}\sum_{i=1}^{T_n} p(D_{n,i} = k, M_{n,i} = m|X_n,\theta^{old}) I(X_{n,i} = l) \tag{A.5}$$

[3]The examples are not shown here to save space.

### A.3.1 E step:

In E step, we compute the expected sufficient statistics mentioned above. Specifically, by running forwards-backwards algorithm on each sequence we get the smoothed node and edge marginals:

$$\gamma_{n,i}(j) := p(D_{n,i} = j | X_{n,1:T_n}, \theta) \tag{A.6}$$

$$\xi_{n,i}(j,k) := p(D_{n,i} = j, D_{n,i+1} = k | X_{n,1:T_n}, \theta) \tag{A.7}$$

Rabiner [168] (page 267) shows that the joint probability

$$\begin{aligned} \tau_{n,i}(j,k) &:= p(D_{n,i} = j, M_{n,i} = k | X_{n,1:T_n}, \theta) \\ &= \gamma_{n,i}(j) \frac{p(M_{n,i} = k | D_{n,i} = j) p(X_{n,i} | D_{n,i} = j, M_{n,i} = k)}{\sum_m p(M_{n,i} = m | D_{n,i} = j) p(X_{n,i} | D_{n,i} = j, M_{n,i} = m)} \end{aligned} \tag{A.8}$$

### A.3.2 M step:

The Maximum Likelihood Estimates (MLE) of the parameters are given by:

$$\hat{\pi}_k = \frac{E[N_k^1]}{N} \tag{A.9}$$

$$\hat{A}_{jk} = \frac{E[N_{jk}]}{\sum_k E[N_{jk}]} \tag{A.10}$$

$$\hat{C}_{km} = \frac{E[N_{km}]}{\sum_m E[N_{km}]} \tag{A.11}$$

$$\hat{B}_{kml} = \frac{E[N_{kml}]}{E[N_{km}]} \tag{A.12}$$