Chapter 1

MEETING STRUCTURE ANNOTATION

Annotations Collected with a General Purpose Toolkit

Alexander Gruenstein
Spoken Language Systems Group
Computer Science and Artificial Intelligence Laboratory
Massachusetts Institute of Technology
alexgru@csail.mit.edu

John Niekrasz
Center for the Study of Language and Information
Stanford University
niekrasz@csli.stanford.edu

Matthew Purver
Center for the Study of Language and Information
Stanford University
mpurver@csli.stanford.edu

Abstract We describe a generic set of tools for representing, annotating, and analyzing multi-party discourse, including: an ontology of multimodal discourse, a programming interface for that ontology, and *NOMOS* – a flexible and extensible toolkit for browsing and annotating discourse. We describe applications built using the *NOMOS* framework to facilitate a real annotation task, as well as for visualizing and adjusting features for machine learning tasks. We then present a set of of *hierarchical topic segmentations* and *action item subdialogues* collected over 56 meetings from the ICSI and ISL meeting corpora using our tools. These annotations are designed to support research towards automatic meeting understanding.

Keywords: topic segmentation, action items, annotation, media, discourse, dialogue, meetings, NOMOS

Introduction

The automatic processing and understanding of multi-party meetings has emerged recently as a major area of research. Technically, meetings present many interesting multidisciplinary challenges; for instance, they have multiple interacting participants and contain spontaneous speech, movement, and gesture. Commercially, they are interesting as they often involve important decisions, yet they are usually poorly documented. Several major projects studying meetings are underway, including Mapping Meetings,[1] M4,[2] AMI,[3] ISL,[4] IM2,[5] and CHIL.[6]

In this discussion, we view meetings from the perspective of building meeting understanding components which comprise part of the *cognitive personal office assistant* being designed for the CALO project.[7] The types of assistance envisioned include summarizing the meeting, actively bringing attention to relevant documents, and helping the collaborative creation of documents in the course of the meeting. Additionally, the content of meetings will be presented in a *meeting browser* which will allow a user to browse a top-level summary, locate pertinent portions, and "drill down" into more detailed structure as desired.

In order to summarize meeting structure in a useful way, it is therefore critical to first understand what sort of structure best assists humans in browsing or reviewing the contents of meetings. With this in mind, we describe an *application-driven* approach undertaken to annotate a set of meetings with relatively coarse structural annotations with the hopes of spurring development of automatic structural segmentation algorithms in this difficult domain. This approach encompasses both the development of a novel framework for manipulating and annotating recordings of multi-party discourse, and annotations performed on meeting corpora

In this chapter, we first describe the architecture developed in the course of the project for both collecting annotations over, and performing research tasks involving, multi-party discourse. While this architecture was developed in the context of working with meetings, it is more generally applicable to multi-party discourse. In particular, we discuss an *ontology of multimodal discourse*, along with its corresponding *ontology programming interface*. We then present *NOMOS*, an *audiovisual toolkit* built on top of this programming interface. *NOMOS*, in turn, was used to develop a tool used to perform annotations, as well as several other tools designed for manipulating meetings.

We then discuss how the tools developed were used to create a new set of annotations of the ICSI (Janin et al., 2003) and ISL (Burger et al., 2002) meeting corpora that mark *hierarchical topic segmentation* and *ac-*

tion items. Finally, we describe the characteristics of these annotations, and analyze inter-annotator agreement.

The annotations and tools described in this chapter, as well as technical documentation, can be downloaded from the world wide web at `http://godel.stanford.edu` under *Software*.

1. Architecture for Meeting Annotation, Research, and Browsing

We begin our discussion by describing the flexible architecture we have developed for working with multi-party discourse. The architecture has grown out of three major threads of research: (1) performing and viewing annotations of discourse, (2) working toward automatic discourse segmentation, and (3) integrating our work with other components comprising a digital office assistant – including components responsible for vision, gesture, and high-level reasoning. In this section, we discuss a *multimodal discourse ontology* (MMDO) which has resulted from these efforts, as well as *NOMOS* – an *audiovisual toolkit* for manipulating multi-party discourse and annotations of that discourse.

MMDO and Ontology Programming Interface

In order to generically represent both corpora and annotations of those corpora, we have devised a *multimodal discourse ontology* (MMDO). The MMDO is fully described in (Niekrasz et al., 2005; Niekrasz and Purver, 2006); here, we give a brief overview focusing on how the ontological framework allows us to unify several research threads. In accordance with our principles of *application-driven* annotations, the MMDO is a suitable representation on top of which to build agents capable of integrating with others into a digital personal assistant.

The MMDO follows recent trends in information technology which put *semantics* in the limelight of data-driven research, the most significant being the Semantic Web (Berners-Lee et al., 2001) which brings ontology and knowledge engineering in contact with the World Wide Web. Following this trend, research in annotation of both linguistic and multimedia resources has begun to shift away from the paradigm of *markup* toward that of *semantic annotation* (Farrar, fort; Geurts et al., 2003). While the former are commonly schematized in a manner similar to an XML DTD, the latter is grounded in a formal ontology, providing an expressive semantics to the annotation and allowing inference.

The MMDO can be found as part of the software architecture in figure 1.1. At the core is a general upper ontology called the Component Library (Barker et al., 2001), the core ontology used in the CALO project.

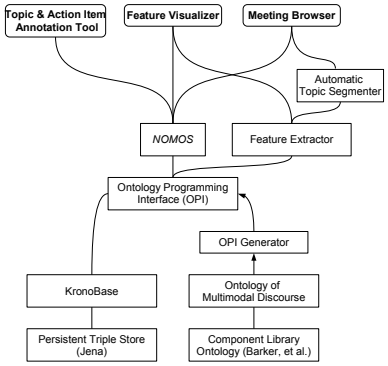


Figure 1.1. Architecture Diagram: The OPI Generator produces an Ontology Programming Interface through which applications can manipulate a knowledge base constrained by an arbitrary ontology. *NOMOS* is built on top of an extensible OPI generated from an ontology of multimodal discourse. *NOMOS*, in turn, serves as the audiovisual backbone of several tools, shown at the top. The OPI is also utilized by the feature extractor, which produces features for automatic topic segmentation.

This provides the most abstract level of semantics to the annotation schema such as events, entities, and roles. Building from these general concepts, we have designed an ontology of multimodal discourse. This layer encodes the concepts important to understanding discourse, such as utterances, words, speaking events, writing events, linguistic constituents, gesturing, etc. In its design, we place an emphasis on unifying our multiple research threads (e.g. human-computer dialogue, open-domain parsing, meeting modeling, and lexical semantics) both theoretically and pragmatically where possible, as well as on capturing as many of the commonly-held concepts in natural language research as possible.

Using this ontology, we create a custom-made Java API – which we call an *ontology programming interface* (OPI) – via an algorithm which encodes the hypernymic relations in the ontology as Java class inheritance and encodes the class relations (attributes) as Java methods. The OPI is written to interface with a triple-store database back-end, which supports persistent access to annotations, currently implemented using the Jena Semantic Framework. *Kronobase* is a layer we have developed for meta-annotation, which allows the recording of important aspects of annotation, including who performed it, when it was performed, and on which resources (other annotations) it is dependent.

NOMOS: An Audiovisual Toolkit for Meeting Annotation, Research, and Browsing

Leveraging the OPI is *NOMOS*,[8] a generic *audiovisual toolkit* for displaying and playing recorded discourses (or, in fact, any type of media recording), and for manipulating and visualizing associated annotations. *NOMOS* provides functionality for graphically displaying information stored in the annotation knowledge base, thus creating a generic platform in which any discourse can be loaded so long as it can be converted to the appropriate format. Moreover, since *NOMOS* is built using the OPI infrastructure, it can easily leverage the same set of underlying ontologies used internally by the CALO systems, including the MMDO. This makes it easy to use *NOMOS* as a platform underlying end-user components of the CALO systems.

NOMOS is a highly customizable environment, serving as the primary ingredient in building the annotation-related software tools discussed in section 2. In particular, both the *Feature Visualizer* and the *Topic and Action Item Annotation Tool* are composed entirely of a set of plugins and templates developed within the *NOMOS* framework; screenshots of these tools can be found in figures 1.5 and 1.6 later in this chapter. The latter is a tool for annotators, while the former is targeted at researchers. In addition, *NOMOS* serves as the basis for the *Meeting Browser* tool currently under development, with which end-users of the CALO systems will be able to browse through an automatically annotated meeting. Figure 1.1 shows the architectural hierarchy contributing to each piece of software. *NOMOS* is implemented entirely in Java, as are the tools built on top of it. Each has been used extensively under Windows, OS X, and Linux.

The rest of this section describes the core components which make up the *NOMOS* toolkit.

Query Editor. Since *NOMOS* is built on top of the OPI, all annotations are stored in a knowledge base accessible via the powerful programming interface exposed by the OPI. While this is an excellent interface for software development experts, it is not necessarily suitable for annotators or end-users of other applications built on top of *NOMOS*. In order to provide an intuitive mechanism for users to interact with this powerful programming tool, *NOMOS* provides a graphical *query editor*. The query editor provides a way for users to construct and edit *queries*, which are an intuitive means of extracting sets of annotations from the knowledge base – much in the same way that SQL queries are used to extract datasets from a database. For instance, as figure 1.2

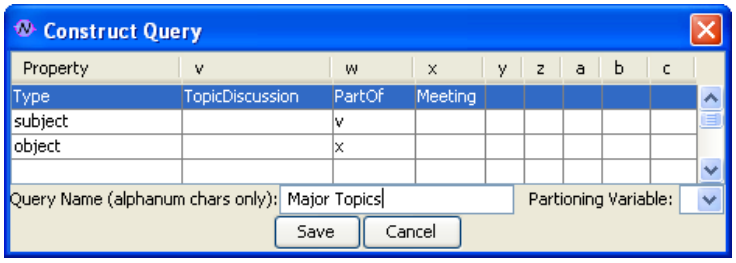


Figure 1.2. The graphical Query Editor

demonstrates, it is a simple matter to construct a query which extracts all of the *major topic* annotations of a particular meeting. Queries can be executed by *NOMOS*, with the results typically displayed on *tracks*, as described below. By providing such a query representation language, *NOMOS* itself can be, for the most part, agnostic with regard to the underlying ontology used to represent the annotations.

Tracks. At the heart of the visual representation of *NOMOS* is the notion of a *track*. Tracks appear in a vertical stack in the center of the display, and can be clearly seen in figures 1.5 and 1.6. The x-axis of each track is measured in time: the start and end of a track correspond to the start and end of the discourse being displayed. A track, then, is appropriate for displaying annotations which are rooted at a particular time in the discourse, for instance: transcripts, topic segments, gestures, or groupings of utterances into linguistic units. *NOMOS* provides default functionality for displaying the properties of time-based annotations as text; in addition, there is extensive plugin support so that developers are free to write custom *plugins* for graphically displaying annotations in whatever means is most appropriate. This makes it possible to develop highly customized applications, such as the *Topic and Action Item Annotation Tool* discussed in the next section. Finally, users can easily zoom in and out on tracks.

Tabbed Panels. In addition to tracks, *NOMOS* also provides a generic mechanism for plugins to represent annotations graphically in any appropriate format as a *panel*. Any number of panels can be shown at once, each appearing as a *tab* which can be clicked. Like a *track*, a *panel* displays a set of annotations retrieved by executing a particular *query*. Unlike a track, a panel need not represent annotations *temporally*. Thus, panels serve primarily as a means of representing entities not associated with one particular time in a discourse – a typical example is the

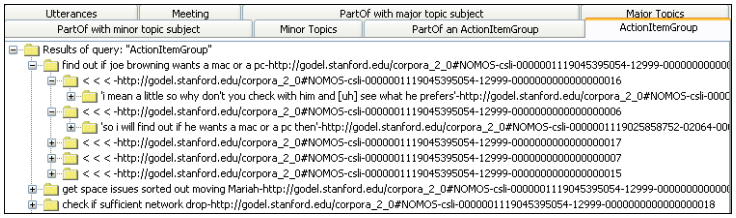


Figure 1.3. An example of a tree panel showing the results of a query

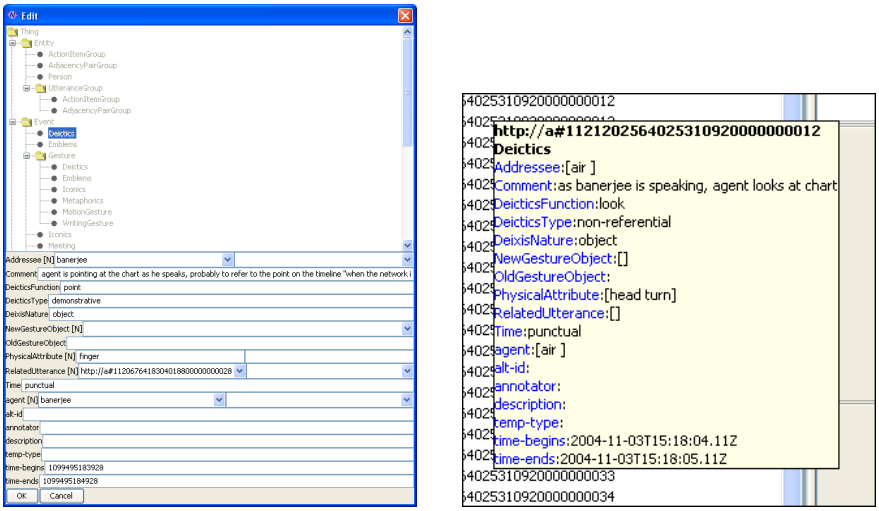
set of *Participants* of a particular discourse. The plugin infrastructure allows developers to create customized ways of displaying and editing entities with a panel. Distributed as part of *NOMOS* are core plugins which visualize any set of entities as a *tree*, where parent/child relationships denote that a *Relation* exists among entities; Figure 1.3 shows an example of such a tree. A tree display is particularly appropriate for non-time-based annotations, often in the form of *persistent annotations*. Persistent annotations are ones which "persist" across multiple discourses (or, more generally, across multiple media files); for instance, a list of discourse references might persist across several discourses, as the same entity might be referred to in the course of multiple discourses.

Templates. In order to customize how a particular set of annotations are displayed, *NOMOS* provides a mechanism for creating and editing *templates*. A template consists of the following:

- A set of queries to be executed
- The types of tracks to be used to display the results of each query – where the type is determined by the type of plugins used to display the track itself and the entities on each track
- The set of plugins used to display annotations as tabbed panels
- Values for configuration parameters for each plugin – used to further customize behavior according to plugin-specific parameters

Given this set of information, templates define how the user will view the annotations: the same set of annotations can be visualized in quite different ways, depending on the subset of the annotations defined through the queries and the plugins used to display the results of these queries.

Transcription. Capabilities for both displaying and editing transcripts are packaged as core *NOMOS* plugins, as these capabilities are de-



(a) Editing an entity's properties – the potential values for each slot are constrained by ontological constraints over the current domain

(b) A tooltip brought up by hovering the mouse over an entity allows for quick interrogation

Figure 1.4. Screenshots of *NOMOS* capabilities for viewing and editing entities

sired in many annotation tasks. In the GUI, each conversational participant is assigned a *track*, in which the transcribed utterances (or speech recognition hypotheses) of that participant are displayed – moving from left to right moves along the time axis. In the screenshot shown in figure 1.5, for example, each of the top seven horizontal tracks are dedicated to the transcripts of the seven meeting participants. Each small box on a track shows the transcription of a single utterance, where the left- and right-hand sides of each box are time-aligned with the start and stop time of the utterance. Zooming in and out allows the user to adjust how much of the transcript is viewed at once; this makes it easy to move from a microscopic view of the discourse to a global one, and back. For instance, while figure 1.5 displays about a minute of discourse, figure 1.6 shows about an hour.

Creating and Editing Annotations. The plugin architecture implemented in *NOMOS* allows tool designers to create arbitrary mechanisms for users to interact with, modify, and create new annotations – the *Topic and Action Item Annotation Tool* described below provides an excellent example of how plugins can lead to such specialization. However, many annotation tasks share a common flavor, so *NOMOS* includes a core set of capabilities for defining new sets of annotations, as well as

modifying existing sets. Using the core architecture, time-dependent annotations (or annotations relating time-based entities to one another) are typically made on additional *tracks*; for example, in a gesture-annotation task, gestures might be shown as *events* on a track so that the start and end time of each gesture can be pinpointed. The *relation* of each gesture to a particular utterance can then be annotated via *drag and drop*: users can drag one entity on to another to set a particular entity as a value for a particular *slot* in another entity. In addition, any entity can be *interrogated* by bringing up a dialog box like the one shown in figure 1.4(a) which shows the slots and values that define that entity, allowing users to directly modify the knowledge base. In both mechanisms for relating entities to each other, *ontological constraints* are enforced by *NOMOS*. For example, if the value of a particular slot can only be of a certain type, *NOMOS* will use tools associated with the OPI to do subclass inference and only allow entities of that type (or subtypes of that type) to be set as the value of a particular slot. Similarly, when editing the properties of a particular entity, only valid slot-fillers in the current domain are presented as options to fill that slot; for example, when choosing the value for a *Participant* slot on an utterance, an annotator will only be able to choose an available entity of type *Person*, since this slot can only take values of this type. These inference capabilities mean that highly customized tools can be developed quickly with little or no programming; instead, ontological constraints directly "customize" the tool.

Audio and Video. A red vertical line overlaying the tracks represents the audio and/or video cursor. It indicates the current position of playback: as playback proceeds, it moves from left to right and the track display is automatically scrolled. Buttons along the bottom can be used to pause playback, or skip forward and back a few seconds – allowing users to quickly replay a bit of the conversation, or quickly fast forward through parts of it. The *focus* button is used to center the display around the current media location; conversely, clicking in a particular location in a track will move the cursor to that location. An arbitrary number of audio and video streams can be synchronized at once; for instance, a video of a discourse can be played back with a separate audio track for each participant mixed together in real time.

Annotation Comparison Capabilities. It is often quite important to be able to see each annotator's annotations of a single discourse side-by-side. Built into *NOMOS* is the capability to partition a set of annotations based on the *annotator* who created each annotation,

laying out each annotator's contributions on a separate track. This capability facilitates easy comparison of multiple annotations made to the same discourse, by stacking each distinct set of annotations on tracks one above another. When comparing topic segmentations, for example, loading each set of annotations one above another and then zooming out allows annotators to get a rough idea of where areas of disagreement and agreement lie; these areas can then be zoomed in on for more detailed discussion. The same techniques can be used to compare the output of annotations automatically generated by, for instance, machine learning techniques. Visually comparing similarities and differences lends powerful (though perhaps anecdotal) insight into differences among algorithms.

Comparison to Similar Efforts

The architecture described in this section provides similar functionality to toolkits in development elsewhere. Of particular note, is the NITE XML Toolkit (Carletta et al., 2004; Carletta and Kilgour, 2004), which is a generic toolkit for performing linguistic annotation tasks. At a basic audiovisual level, NITE provides fairly similar funtionality to the NOMOS architecture described in this chapter: synchronized audio and video playback and a plugin architecture. A key difference in the visual display is that by default transcripts in NOMOS are displayed along *tracks*, while in the NITE system they are presented as linearized text. While both approaches have their advantages in different applications, the tracked presentation provides the most natural means for emphasizing *overlap*, a feature of multiparty discourse we believe is often ignored in natural language processing applications.

Greater differences between NOMOS and NITE arise as a result of fundamental differences in the way the two represent the underlying annotations. The NOMOS architecture is centered around semantic annotation, which results in annotations made according to particular schemata, and a uniform user interface built around editing the fields of entities and their relationships to one another. Moreover, the semantic framework made use of by NOMOS is meant to be interoperable with high level reasoning components currently in development, making it quite straightforward to transfer knowledge in the form of annotations to agents capable of reasoning. NITE, on the other hand, stores all annotations as XML.

Finally, our framework stands out in that the OPI is "compiled" from the annotation schema. The availability of this Java API makes it straightforward to manipulate and analyze annotations, as we have

done in section 4. The OPI makes it possible to write scripts which are forced at *compile-time* to conform to the annotation schema. In addition, since annotations are stored in a standard triples format, other knowledge-base tools can easily manipulate them.

2. Tools

In this section, we describe several distinct tools we have developed using *NOMOS* as the core platform, backed by the multimodal discourse ontology and its associated ontology programming interface. All of the tools described in this section are made up entirely of a set of *NOMOS* plugins, laid out using the standard *template* mechanism described in the previous section.

The *Topic and Action Item Annotation Tool* was developed for the use of the annotators performing the annotations which will be described later in this chapter. The *Feature Visualizer* is a tool we have developed in the course of our preliminary automatic segmentation work. And the *Meeting Browser* is a tool currently under development which is intended to be an end-user component of the CALO digital personal office assistant. Taken together, these tools demonstrate the flexibility of the architecture we have developed, showing how it can play a cross-cutting role across the tasks of meeting annotation, browsing, and research.

Topic and Action Item Annotation Tool

A screenshot of the *Topic and Action Item Annotation Tool* is shown in figure 1.5. It leverages the full features of *NOMOS*, complementing them via plugins to allow for additional annotation capabilities specialized to annotating topic segments and action items. The tool is an excellent example of how the generic capabilities provided by *NOMOS* can be further specialized via plugins to make performing a specific set of annotations particularly efficient.

We briefly note here features developed in the tool (as well as in *NOMOS* in general) which particularly decrease the high cognitive load demanded by the annotation task. Notably, key capabilities revolve around simultaneously providing global and local insight into the meeting and annotations, as well as the capability to easily revise draft annotations.

Topic and action items (see section 3) are annotated via context menus available on the tracks displaying the utterance of the discourse. Specialized tabbed *panels* (see above) show a topic hierarchy and a list of action items (shown in the upper left of figure 1.5), giving an overview of the annotations at a global level. During the pilot period of annota-

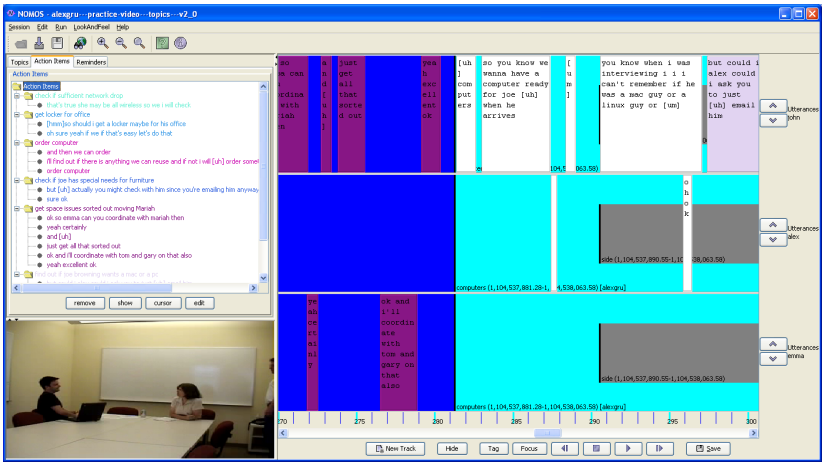


Figure 1.5. Screenshot of the *Topic and Action Item Annotation Tool*

tion, it became clear how important it was to be able to easily modify annotations after making an initial rough pass through a discourse. As a result, capabilities for *renaming* and *deleting* both topics and action items exist, as well as the ability to *promote*, *demote*, or *merge* major and minor topics as appropriate. These capabilities provide single-click *shortcuts* to what would otherwise be somewhat involved tasks in the default *NOMOS* framework. In addition, "reminders" can be inserted at particular time points, allowing annotators to make notes to refer back to in a subsequent pass.

Specialized *track* plugins provide a task-specific visualization of both topic segmentations and action items. *Major* topics are signaled graphically on the tracks containing the utterances by alternating the background color. The *minor* breaks are indicated by the narrower bands of alternating light and dark gray centered vertically in the track. For instance, in figure 1.5 there are two major topics visible in the time slice shown; in addition, the second major topic is a parent to one visible child minor topic. Brief descriptions assigned to each major and minor topic are displayed in each track. Finally, the entire hierarchy of topics can be shown by clicking on the appropriate tab in the upper left hand corner; clicking on any topic in this list will shift the track display to the start of that topic.

An example of annotations for *action items* is also displayed in figure 1.5. Several utterances by the top and bottom speaker in the first major topic have been shaded the same color to indicate that they are related to the same action item; similarly, an utterance on the top right has

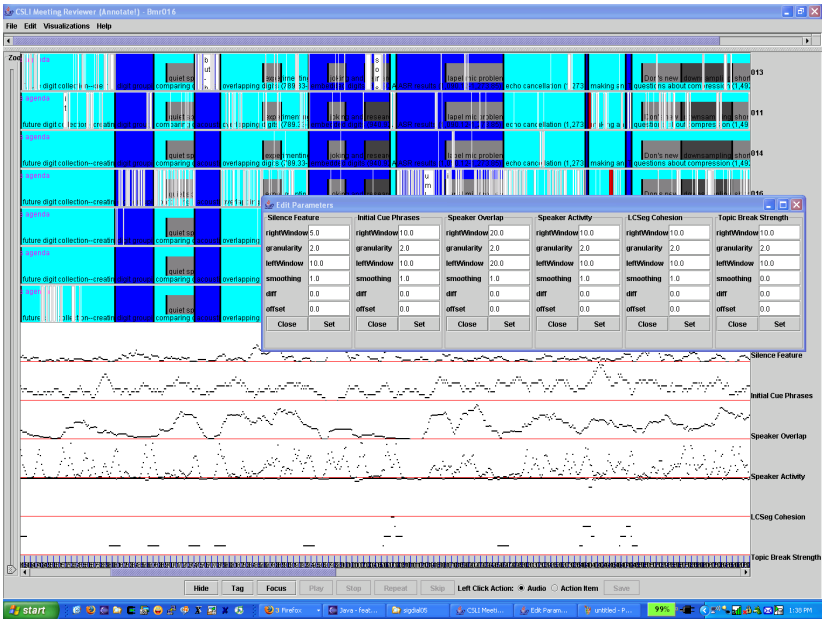


Figure 1.6. Screenshot of the *Feature Visualizer*

also been highlighted a different color to indicate it is part of a *different* action item. Moreover, the panel in the upper left corner lists all of the action items marked in the entire discourse. A brief description of each appears, followed below by the transcript of each utterance comprising that action item. Clicking on an utterance will scroll the track display to show that utterance. Each action item is assigned a color, shown both in the summary in the upper right and in the highlighted utterances in the display.

Feature Visualizer

We have developed a generic *Feature extractor* and *Feature Visualizer* using the ontology programming interface and *NOMOS* audiovisual toolkit, as the architecture digram in figure 1.1 shows. We mean *feature* here in the sense of features which can be computed from discourse as input to machine learning algorithms for classification tasks such as topic segmentation. The *Feature Extractor* is simply a set of Java classes which provide core functionality for processing discourse, as represented by the OPI. Functionalities include: extracting sets of utterances in a given time window, turning these utterances into bags of words per speaker, smoothing feature values, and calculating their derivatives.

Moreover, generic tools are provided for iterating over discourses, processing them, and extracting sets of feature values at regular intervals which can then be piped directly into learners like decision trees, neural nets or support vector machines.

The *Feature Visualizer* is built on top of the extraction architecture, using a set of plugins to create the GUI using *NOMOS*. It displays calculated feature values alongside an annotated discourse, as shown in figure 1.6. Moreover, as the popup window in figure 1.6 shows, it allows the user to dynamically modify each feature's parameters (for example: window size, smoothing, or other feature-specific parameters) and immediately observe the results. We have found the visualizer to be invaluable in debugging algorithms for feature extractors, tweaking parameter values, and hypothesizing new, interesting features.

Meeting Browser

We are currently developing a *Meeting Browser* tool, which will sit on top of both the audiovisual toolkit and the feature extractor. The eventual development of this tool is the motivation that has driven our annotations and associated schema. The browser is meant to allow users to "drill down" through the structure of the meeting, easily pinpointing segments of interest.

3. Annotation Motivations and Schema

We now turn to describing an annotation task performed using the *Topic and Action Item Annotation Tool* described in the previous section, providing a real world example of both the sort of anntations which may be performed in the NOMOS architecture, and the type of analyses which are straightforward to perform using the OPI compiled from the annotation schema. We focus on two types of discourse structure annotations. The first, *topic segmentation*, breaks the discourse up into a (hierarchical) sequence of topics. The second, *action item subdialogues*, marks particular utterances as being relevant to the discussion or assignment of action items. In this section, we describe our motivations in studying these phenomena, related work, and the iterative process by which we refined an application-driven annotation schema.

We worked with the ICSI Meeting corpus (Janin et al., 2003) and the ISL Meeting Corpus (Burger et al., 2002) because both contain high-quality close-talking microphone recordings of conversational speech in a meeting environment, as well as word-level transcriptions and utterance-level timing information. We focused mainly on the ICSI corpus because its contents most closely matched our task of processing fairly informal,

office-style meetings. In addition, extensive annotations have already been completed on the ICSI corpus, including: dialogue acts (Shriberg et al., 2004), "hot spots" (Wrede and Shriberg, 2003), and some work on topic segmentation (Galley et al., 2003; Carletta and Kilgour, 2004).

Topic Segmentations

A significant challenge in spoken discourse segmentation is providing a concrete definition of the problem – the desired concepts of both *topic* and *segmentation*. To that end, we first briefly discuss the conceptualizations – and motivations behind those conceptualizations – that have arisen in the related fields of segmenting text and monologue. We then discuss previous work in segmenting discourse, our own motivations, and finally outline an annotation schema derived from these motivations.

Text and Monologues. The segmenting of text documents is often motivated by information retrieval tasks – for instance, so that a single appropriate segment can be returned matching a query. In some cases, topic boundaries are hand-annotated, as in (Hearst, 1994). However, topic boundaries are often artificially created by concatenating multiple articles together, as in (Galley et al., 2003; Choi, 2000). Moreover, since text is written linearly, usually with clearly punctuated boundaries in the form of sentences and paragraphs, it is natural to assume that topic boundaries will occur at such places. Thus, such "natural" boundaries both define and limit the search space. In addition to text, there has been much research in segmenting *non-conversational speech*; essentially monologues or series of monologues. For example, much work has been done on automatically segmenting broadcast news, *e.g.* (Tür et al., 2001; Beeferman et al., 1999; Allan et al., 1998).

The tasks of segmenting text and monologue are similar in that both tend to have fairly well defined topic structure. In the case of artificial text corpora created through concatenation, topic boundaries can be objectively defined over the concatenated article boundaries. News broadcasts tend to consist primarily of scripted speech – with little spontaneity – produced by highly practiced professionals (though some work has also been done on more spontaneous monologues, see (Passonneau and Litman, 1997)). Topic boundaries in news broadcasts are designed to be obvious, with unambiguous shifts from one story to the next. In both domains, automatic segmentation algorithms tend to rely primarily on lexical co-occurrence statistics to calculate a measure of *lexical cohesion* between chunks of text (Hearst, 1994; Hearst, 1997). In the case of monologue, prosodic cues are often utilized as well (Tür et al., 2001; Hirschberg and Nakatani, 1998).

Discourse. When turning to spontaneous discourse, most previous work has followed this text/monologue approach: for example, when (Galley et al., 2003) annotated 25 meetings in the ICSI Meeting corpus for topics, the discourse was represented *linearly* as a series of non-overlapping utterances, topics were represented as a linear sequence of segments, and topic boundaries were allowed only at *speaker changes*. Although we are aware of one project in which *hierarchical* topic annotations are being used (on the ICSI corpus using the NITE XML toolkit (Carletta and Kilgour, 2004)), no annotations are yet publicly available.

Rather than adapting the task of discourse segmentation to make it look more like a text segmentation task, we took an *application-driven* approach to segmenting discourse. Our motivation for topic segmentation was to enable broad understanding of a discourse, providing a coarse summary segmentation for broad-perspective user browsing capabilities, and allowing for selective "drill-down" and replay; for more detailed discussion of the utility of high-level segmentations, see (Banerjee et al., 2005). We therefore wanted to collect annotations which can be leveraged specifically to provide such capabilities for a digital personal office assistant. Specifically, we instructed the annotators to look at the problem of providing a topic segmentation from the perspective of utility: if they were reviewing a meeting they might not have attended, what segmentation would help them quickly "drill down" to portions they might be particularly interested in reviewing. While a bit vague, this description of the task avoids biasing the annotators toward relying on particular discourse phenomena or restricting them to particular boundary locations; (Ries, 2001) argues that such an application-driven approach, with linguistically naive coders, may help best represent end-users of meeting browser systems.

This application-driven approach proved difficult at first, resulting in low inter-annotator agreement among the two undergraduate annotators in the first five meetings that were annotated. However, through discussions of the annotations (often using annotation comparison capabilities discussed in section 1) – discussions in which no actual concrete annotation criteria for what always must constitute a topic break were discussed – an acceptable level of inter-annotator agreement was reached for the majority of meetings (see section 4). Agreement results eventually reached a plateau, at which point further discussion of the annotation guidelines was terminated. At this point, guidelines were then drawn summarizing the result of these discussion: see (Gruenstein et al., 2004). The resulting schema is discussed below.

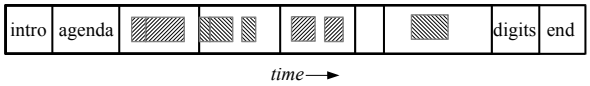


Figure 1.7. A sample hierarchical meeting segmentation

Topic Segmentation Schema. Meetings were segmented according to a two-level hierarchical segmentation schema. In the top (*major*) level of the hierarchy, the entire meeting is wholly and contiguously segmented, where segment boundaries symbolize highly salient breaks in discourse structure and/or distinguish parts of the discourse between which there is an obvious difference in subject matter. In the second (*minor*) level of the schema, major segments are optionally subsegmented without a requirement for contiguity, but with overlapping segments forbidden. Minor segments signify either a temporary digression or a more focused discussion of the subject matter, while still remaining directly relevant to the encompassing major segment. Our pilot annotation work indicated that restricting topic breaks to speaker changes was an unnatural restriction. Instead, our schema allows topics to start and end at any point in the discourse, even in the middle of a single speaker's utterance. Some ramifications of this choice are discussed in section 4. We note that while our choice to allow topic breaks at any time point may be "permissive," it may in fact not be permissive enough when considering multiparty discourse. In such discourses, it may sometimes be the case that while some speakers have moved along to a new topic, others may still linger on an old topic; or, a few speakers may discuss one topic amongst themselves while others discuss another. While such phenomena are interesting, we felt that given the *application-driven* nature of our annotations (with the application being a meeting reviewer tool), capturing such granularity was not necessary. Figure 1.7 depicts a meeting segmented according to the schema, with vertical lines separating major topics, and shaded areas representing minor topics.

Annotators also gave brief descriptive names to topics, though no standards were set as to the format or content of the assigned names, with the exception of the following *reserved* topic names:

- *AGENDA*: the portion of the meeting in which the agenda is presented and discussed
- *INTRO*: speech before the meeting "officially" begins (appears in every meeting, though may have zero length)
- *END*: speech after the meeting "officially" ends (appears in every meeting, though may have zero length)
- *TECHNICAL DIFFICULTIES*: a period in which there are technical difficulties with recording equipment

- *DIGITS*: the digits task in the ICSI meeting corpus [see (Janin et al., 2003)]

Except for *AGENDA*, the reserved names simply serve the purpose of highlighting portions of the recording which might not be considered part of the meeting proper; below we discuss how they play a role in defining a reference segmentation. In addition, if a new topic is a continuation of a discussion of a previous topic left off earlier, the convention is used that the same descriptive text is given for both topics – implicitly linking them.

Action Items

Though the focus of the annotation work was hierarchical topic segmentation, annotators also marked *action items*. Previously, we have shown how simple task-assignment charts can be inferred from highly scripted, multimodal meetings (Kaiser et al., 2004). In moving to free-form meetings, identifying *decision points* like action items follows as a natural first step in extending this work.

For the purposes of annotation, we define an action item loosely as a task which is discussed in the meeting and then assigned to a participant (or participants) to complete at some point after the completion of the meeting. In our schema, action items are defined as sets of *utterances*, rather than start and end times: this is possible because action items are usually discussed only briefly, so it is feasible for an annotator to pinpoint particular utterances in which the discussion occurred. Moreover, it is useful to identify as specifically as possible the utterances in which action items were discussed, as not all speech within a time window may be relevant due to the high levels of speech overlap in multi-party conversations.

Note also that while identifying the general regions of action item discussion could be useful for logging and browsing by a user, it is only by identifying the relevant utterances themselves that we will be able to move towards automatic interpretation of the action items, where interpretation might include: identifying the person it has been assigned to, its deadline and the information about the task it involves. With this goal in mind, we plan future annotation passes to further classify each utterance into specific categories such as *task*, *deadline* and *person* assignment. Furthermore, it may be useful to mark information particular to the task, such as: the person it has been assigned to, its deadline, and its relation to other tasks.

4. Analysis of Collected Annotations

We collected annotations for a total of 65 meetings, however 9 of those meetings were not annotated by both annotators, were annotated during our preliminary annotation sessions, or had other problems. Excluding this set of meetings leaves a total of 56 annotated meetings: 40 meetings from the ICSI corpus and 16 from the ISL corpus, totalling 45.9 hours. In this section, we provide a statistical analysis of our annotations of this set, along with some more qualitative observations. We describe multiple algorithms which have been applied to the data to make our analysis possible. We also provide an analysis of inter-annotator agreement using multiple metrics. Last, we compare our annotations to other similar datasets.

Pre-processing

Every meeting recording has a beginning and end which do not actually contain meeting dialogue and which are not relevant to an analysis of topic structure. Before analysis, we therefore perform pre-processing of our annotations to produce a segmentation that does not contain these sections of the discourse. Because our annotators were asked to annotate these special cases, our pre-processing algorithm simply takes the union of the set of *INTRO* and *END* segments from both annotators and removes those portions of the discourse from both annotations. All the analyses presented below were done after this pre-processing step. While pre-processing of *DIGITS* and *TECHNICAL DIFFICULTIES* segments is necessary for training of topic detection algorithms, these segment types were not removed prior to the analysis presented in this section.

Segment and Break Classification

While most text segmentation methods constrain the number of possible segmentations by specifying a finite set of discrete locations where segment boundaries may occur (most often at sentence boundaries), our annotators were free to assign boundaries at any time during the discourse. Unfortunately, this complicates our use of standard evaluation metrics, and it doesn't suit iterative automatic discourse segmentation algorithms which operate at discrete intervals of time.

To overcome these obstacles we transform our annotations into a set of classifications in two ways, arriving at what we call a *segment classification* and a *break classification*. For each of the two, the first step is to divide the discourse into temporal units based on a set of possible

break locations, e.g. a set of evenly-spaced temporal values, utterance start times, or speaker changes. We use evenly-spaced intervals of 20 seconds in our analysis.

In the case of evenly-spaced windows, a discourse d is evenly divided into $i = |d|/n$ non-overlapping contiguous temporal intervals of length n, with the last window realizing any remainder and possibly being cut short. For the *segment classification*, each temporal unit is classified as to which topic segment it belongs. Temporal units which contain segment boundaries are classified simply by determining in which half of the unit the annotated boundary lies. If it lies in the later half, the unit is classified as belonging to the previous topic segment. For the earlier half, it is classified with the following topic segment. This produces segment boundaries which are between windows.

For *break classification*, each unit is classified as to whether or not it contains a topic boundary. This latter interpretation is essential for making use of the Kappa agreement statistic when the number of topic segments is unconstrained, as it is here. This may be transformed back into a set of segment boundaries by placing boundaries at the center of windows which have been classified as containing a topic break.

Reference Segmentation

Another essential processing step is to produce a reference segmentation from our individual annotations. This is important to providing a comparison to other annotations such as those used in (Galley et al., 2003), and for training automatic segmentation algorithms. Galley, et al. create a reference segmentation by establishing sets of topic boundaries based on co-occurrence between annotations within 20 seconds. They then choose those sets which have been annotated by a majority and establish a boundary at each set's median time value.

In our current method, we employ the same strategy of discarding the minor segments. However, we believe benefit can be derived using our second tier of segmentations as there are many cases where topic boundaries are annotated as a major shift by one annotator and as a minor shift by the other, suggesting some level of agreement that should be used. Also a second tier of segmentation in an automatic segmentation application would likely be useful for more localized "drill-down". Therefore, we do not believe this strategy should be a hard and fast rule: we provide our segmentations as individual annotations without establishing a defined reference. We will likely employ different strategies in the future for establishing a reference segmentation which incorporates minor boundaries.

Evaluating inter-annotator agreement

In this section we present the results of evaluating agreement between our two annotators and compare multiple agreement metrics. The results show variance among meetings, suggesting that the topic segmentation task may be ill-formed for certain classes of meetings.

The current standard metric for measuring inter-annotator agreement in classification tasks is the kappa statistic (K) (Carletta, 1996). While K is a good measure of how well annotators can agree on pinpointing topic breaks at time points, it does not accommodate near-miss break assignments in which annotators label different nearby time points as topic breaks. For the evaluation of segmentation algorithms specifically, two metrics are most commonly used: P_k (Beeferman et al., 1999) and *WindowDiff* (WD) (Pevzner and Hearst, 2002). These were designed principally to evaluate text segmentation algorithms that operate at sentence boundaries, but can be applied to continuous-time segmentations through the use of windowing. P_k accommodates near-miss labelings by considering how likely two time points are to be assigned to the same topic, while WD further refines this notion by measuring the difference in number of topic breaks between two time points. Each metric provides a reasonable, though different, evaluation of inter-annotator agreement. Results given in table 1.1 and figure 1.8 show a high degree of correlation among them.

Our measurement of K follows that suggested in (Carletta, 1996) and described fully in (Siegel and N. J. Castellan, 1988):

$$K = \frac{P(A) - P(E)}{1 - P(E)} \tag{1.1}$$

This measures pairwise agreement on classification tasks, correcting for chance, where $P(A)$ is the probability of agreement and $P(E)$ is the probability of chance agreement between two annotators. Increasing values of K indicate better agreement. We use the break classification form of our annotations when calculating this metric.

Our second measurement is a variation on P_k, which is computed as follows:

$$P_k(\mathbf{a}, \mathbf{b}) = \frac{\sum_{i=1}^{N-k} (\delta_{\mathbf{a}}(i, i+k) \mathbin{\overline{\oplus}} \delta_{\mathbf{b}}(i, i+k))}{N-k} \tag{1.2}$$

P_k estimates the probability that two randomly drawn temporal values occurring during the discourse are classified as being in *different* segments by the two segmentations $\mathbf{a}$ and $\mathbf{b}$ – thus, decreasing P_k indicates better agreement. Here, $\delta_{\mathbf{x}}(t_1, t_2)$ is an indicator function which evaluates to 1 if the segmentation $\mathbf{x}$ places the times t_1 and t_2 in the same

Table 1.1. Mean/median agreement on topic segmentations

	Major topics	*Major and minor topics*
WD	28.9% / 29.2%	32.7% / 33.8%
P_k	22.6% / 22.5%	26.5% / 26.2%
K	52.1% / 53.2%	47.0% / 46.9%

segment. The $\overline{\oplus}$ operator represents the `XNOR` function. As mentioned in (Beeferman et al., 1999), if the value k is set to half the mean topic segment length, the metric provides appropriate results for all degraded forms of segmentation, including random segmentation. We impose a slight variation on the calculation of k by not treating one annotation as a reference and the other as a hypothesis, but rather by incorporating both annotations when calculating the average segment length.

The third and final metric, WD, is the most recently proposed and is a variation on P_k intended to improve its tolerance of near-misses and varying segment size distributions:

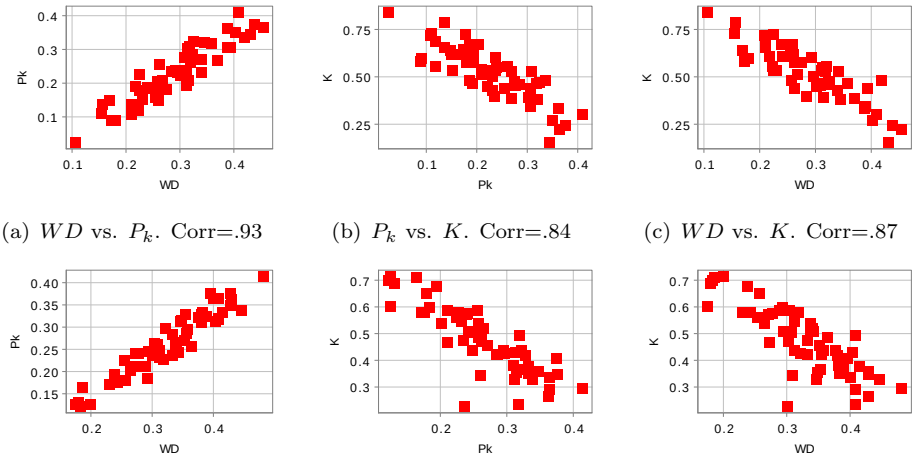
$$WD(\mathbf{a}, \mathbf{b}) = \frac{\sum_{i=1}^{N-k} \left(|b_{\mathbf{a}}(i, i+k) - b_{\mathbf{b}}(i, i+k)| > 0\right)}{N-k} \tag{1.3}$$

Here, $b_{\mathbf{x}}(t_1, t_2)$ replaces $\delta_{\mathbf{x}}(t_1, t_2)$ from equation 1.2 and is the number of segment boundaries occurring between times t_1 and t_2 in the segmentation $\mathbf{x}$. This metric is different from P_k in that a penalty is assessed at each evaluation point if the number of segment breaks in the interval is not equal between the annotations. In P_k, the number of breaks is not counted and a penalty is only assessed if one totals 0 and the other does not. For WD, we impose the same change to the calculation of k as we do in our calculation of P_k.

Because our annotations have continuous-time boundaries, we must establish a stepping method for i. Following (Galley et al., 2003), we use 20-second stepping intervals. An investigation of inter-annotator agreement for varying step sizes from 5 to 60 seconds showed no significant change in P_k or WD. An evaluation of K with varying break classification window widths showed a maximum at near 20 seconds. For the purposes of transparency and descriptiveness, we include measurements of all three of the above metrics in our evaluation, using a 20-second window width and/or step size.

Results

Multiple graphs showing results for inter-annotator agreement may be found in Figure 1.8. The top three plots show agreement based



(a) WD vs. P_k. Corr=.93 (b) P_k vs. K. Corr=.84 (c) WD vs. K. Corr=.87

(d) WD vs. P_k. Corr=.95 (e) P_k vs. K. Corr=.84 (f) WD vs. K. Corr=.82

Figure 1.8. Segmentation inter-annotator agreement: each point represents a single meeting. (a)-(c) include major topics only; (d)-(f) include major and minor topics

only on major topic boundaries. The bottom three include minor topic boundaries in the evaluation. Each of the columns rows shows a pairwise comparison of two of the three metrics. Means and medians are provided in Table 1.1.

As expected, the metrics show a high level of correlation (correlation coefficients are given in the figure captions). It is difficult to say what values for our metrics signify a "good" level of reliability in the annotations. In computational linguistics, a value of $K = .67$ is generally used as a cutoff for reliable analysis, though it has been suggested on multiple occasions that this is not appropriate for all tasks (see (Eugenio and Glass, 2004) for a discussion). Undeniably low scores do occur in our annotations. This is often found for meetings which involved presentations of visual information, which made the audio-only annotation task difficult. Some of this information may be gleaned from the available annotator notes. Poor agreement and self-evaluation by the annotators on some meetings suggest that some of the annotations should not be used. It should be noted that there are more numerous outliers in the evaluation of major segments only, which is a result of there being some meetings which were only annotated as having as few as two major boundaries (after pre-processing).

In addition, the two annotators marked 921 and 1267 utterances respectively as belonging to discussion about action items. We have yet to do significant analysis of these annotations and wish to produce further

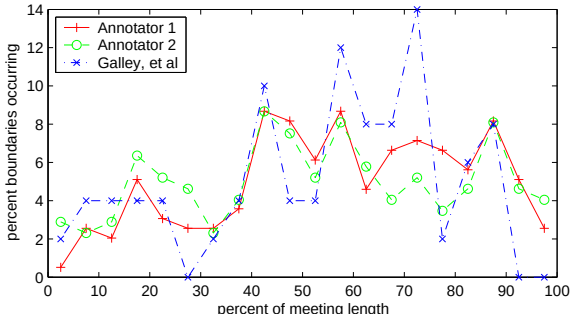


Figure 1.9. Distribution of boundaries over meeting duration.

annotations of decision-making processes before using the data. Current analysis shows inter-annotator agreement of utterance classification at $K = .36$.

Comparison with similar annotation sets

In (Galley et al., 2003), 25 of the meetings in the ICSI Meeting corpus were hand annotated for topic breaks. A minimum of three annotators per meeting were given the task of deciding if each *speaker change* in a linearly represented meeting constituted a topic break.

Due to their process of establishing a reference segmentation, topic boundary frequency is significantly different between their annotations and our individual annotations. Our annotators produced major segments with an average length of 180 seconds, while Galley, et al.'s average 684 seconds. Their annotations total 12.6 hours, while ours total 45.9.

Another noteworthy statistic is the distribution of topic boundaries over meeting duration, depicted in figure 1.9. The distribution is shown for each of our annotators and from Galley, et al. While the total number of meetings is different between the two sets, there are significantly more topic changes in the latter half of the meetings for each. It will be interesting to take note of this statistic in other corpora to see if the trend is universal. It is unclear if this is a by-product of the annotation process or of the meeting itself.

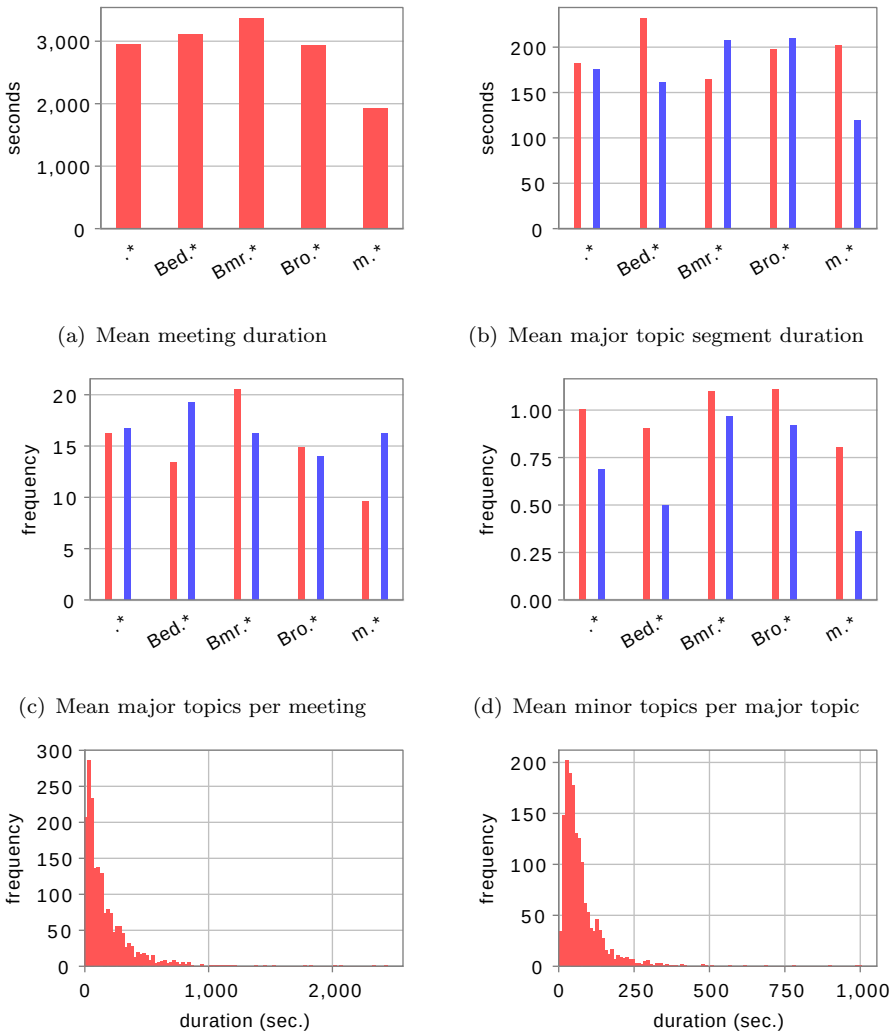
Finally, figure 1.10 gives some further details about the characteristics of the topic segmentation annotations we have collected. The first four graphs highlight characteristics of distinct sets of meetings based on their general type. *Bed*, *Bmr*, and *Bro* are each a particular subset of the ICSI meetings which have a similar theme (see the corpus documentation for details), and *m* indicates meetings from the ISL corpus. The first graph

shows mean meeting length, while the second shows mean major topic segment length. The following two show the number of major topic segments per meeting and the number of minor topics per major topic. Finally, histograms indicating the distribution of the durations of major and minor topic segments are given.

5. Current and Future Work

The work described in this chapter represents our first steps toward automatic meeting understanding for a personal office assistant. While coarse-level meeting segmentation is a useful first step, we are tackling the problem from multiple angles: including robust natural language chunk parsing, dialogue act detection, argumentation structure analysis, and decision detection. Our first steps in these areas will likely be similar to those we have taken in topic segmentation: establishing modular additions to the annotation ontology, supporting this in the NOMOS audiovisual toolkit, coding annotation, research, and application tools for them, and then collecting annotations. Annotation of these richer structures will require greater use of the inference capabilities the ontology provides. For example, a tool designed for the annotation of argumentative structure will need to employ the constraints imposed by the ontology on that structure through the use of reasoning engines to constrain the annotations a human can make.

In parallel, we are currently developing automatic topic segmentation and action item detection tools by training classifiers on the annotations presented above while using the presented software framework for feature extraction and visualization. For topic segmentation, initial investigation following a roughly similar approach to (Galley et al., 2003) (using a decision tree trained on both lexical cohesion values and some discourse-based features – speaker activity, speaker overlap, amount of silence – and cross-validating over 25 ICSI meetings) has given average P_k error levels of around 0.35 for major topics. This is higher than Galley, et al. achieved on their segmentation, but this would be expected with our finer-grained and less restricted notion of topic, and is at least comparable to our human annotator agreement. Future development will add prosodic features and chunk parser output. For action item detection, an initial n-gram-based classifier using a combination of manually and automatically extracted features is currently being developed, and has shown promising performance on a separate small test meeting corpus; future development will include the use of more structured hierarchical action item annotations.



(a) Mean meeting duration

(b) Mean major topic segment duration

(c) Mean major topics per meeting

(d) Mean minor topics per major topic

(e) Major topic segment durations

(f) Minor topic segment durations

Figure 1.10. Topic Segmentation Annotation Characteristics: (a) gives the mean duration of the meetings annotated of each different type, which is useful in interpreting the other graphs; (b)-(d) show per-annotator statistics broken down by meeting type, while (e)-(f) show histograms of major and minor topic segment durations of the reference annotation.

Lastly, we expect to use the NOMOS audiovisual toolkit as a part of the CALO office assistant itself. This will involve the integration of our architecture with the CALO Desktop environment, allowing for pervasive feedback to our algorithms and online supervised learning.

Acknowledgments

Thanks to our two annotators Michael Deeringer and Claire Gilbert, to our CALO associates Satanjeev Banerjee and Bill Jarrold, three anonymous SIGDIAL reviewers who commented on the version of this paper appearing in the proceedings of SIGDIAL 2005, the participants of the conference who provided invaluable feedback, and two additional reviewers who evaluated this book chapter. This work was supported by DARPA grant NBCH-D-03-0010. The information in this publication does not necessarily reflect the position or the policy of the US Government, and no official endorsement should be inferred.

Notes

1. http://labrosa.ee.columbia.edu/mapmeet/
2. http://www.m4project.org
3. http://www.amiproject.org
4. http://penance.is.cs.cmu.edu/meeting_room/
5. http://diuf.unifr.ch/im2/
6. http://chil.server.de/
7. http://www.ai.sri.com/project/CALO
8. *NOMOS* is an abbreviation for "anNOtation of Media with Ontological Structure."

References

References

Allan, James, Carbonell, Jaime, Doddington, George, Yamron, Jonathan, and Yang, Yiming (1998). Topic detection and tracking pilot study: Final report. In *Proc. of the DARPA Broadcast News Transcription and Understanding Workshop*.

Banerjee, Satanjeev, Rose, Carolyn, and Rudnicky, Alex (2005). The necessity of a meeting recording and playback system, and the benefit of topic-level annotations to meeting browsing. Proceedings of the Tenth International Conference on Human-Computer Interaction.

Barker, Ken, Porter, Bruce, and Clark, Peter (2001). A library of generic concepts for composing knowledge bases. In *Proceedings of the First International Conference on Knowledge Capture*.

Beeferman, Doug, Berger, Adam, and Lafferty, John D. (1999). Statistical models for text segmentation. *Machine Learning*, 34(1-3):177–210.

Berners-Lee, Tim, Hendler, James, and Lassila, Ora (2001). The Semantic Web. *Scientific American*.

Burger, Susanne, MacLaren, Victoria, and Yu, Hua (2002). The ISL meeting corpus: The impact of meeting type on speech style. In *Proceedings of the ICSLP 2002*, Denver.

Carletta, Jean (1996). Assessing agreement on classification tasks: The kappa statistic. *Computational Linguistics*, 22(2):249–255.

Carletta, Jean and Kilgour, Jonathan (2004). The NITE XML toolkit meets the ICSI meeting corpus: Import, annotation, browsing. In *Proceedings of the Joint AMI/PASCAL/IM2/M4 Workshop on Multimodal Interaction and Related Machine Learning Algorithms*, Martigny, Switzerland.

Carletta, Jean, McKelvie, David, Isard, Amy, Mengel, Andreas, Klein, Marion, and Moller, Morten Baun (2004). A generic approach to software support for linguistic annotation using XML. In Sampson, Geoffrey and McCarthy, Diana, editors, *Corpus Linguistics: Readings in a Widening Discipline*. Continuum International.

Choi, Freddy Y. (2000). Advances in domain independent linear text segmentation. In *Proceedings of NAACL-00*.

Eugenio, Barbara Di and Glass, Michael (2004). The kappa statistic: A second look. *Computational Linguistics*, 30(1):95–101.

Farrar, Scott (fort). Using 'ontolinguistics' for language description. In *Ontolinguistics*. Mouton de Gruyter, Berlin.

Galley, Michel, McKeown, Kathleen, Fosler-Lussier, Eric, and Jing, Hongyan (2003). Discourse segmentation of multi-party conversation. In *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics (ACL-03)*, Sapporo, Japan.

Geurts, Joost, Bocconi, Stefano, van Ossenbruggen, Jacco, and Hardman, Lynda (2003). Towards ontology-driven discourse: From semantic graphs to multimedia presentations. In *Second International Semantic Web Conference (ISWC2003)*, pages 597–612, Sanibel Island, Florida.

Gruenstein, Alexander, Niekrasz, John, Deeringer, Michael, and Gilbert, Claire (2004). *Discourse annotation using Annotate!* godel.stanford.edu/twiki/bin/view/Public/TopicActionItemDataset.

Hearst, Marti (1997). TextTiling: Segmenting text into multi-paragraph subtopic passages. *Computational Linguistics*, 23(1):33–64.

Hearst, Marti A. (1994). Multi-paragraph segmentation of expository text. In *Proceedings of the 32nd Meeting of the Association for Computational Linguistics*, Los Cruces, NM.

Hirschberg, Julia and Nakatani, Christine (1998). Acoustic indicators of topic segmentation. In *Proc. of ICSLP*.

Janin, Adam, Baron, Don, Edwards, Jane, Ellis, Dan, Gelbart, David, Morgan, Nelson, Peskin, Barbara, Pfau, Thilo, Shriberg, Elizabeth, Stolcke, Andreas, and Wooters, Chuck (2003). The ICSI meeting corpus. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP-03)*, pages 364–367.

Kaiser, Ed, Demirdjian, David, Gruenstein, Alexander, Li, Xiaoguang, Niekrasz, John, Wesson, Matt, and Kumar, Sanjeev (2004). A multimodal learning interface for sketch, speak and point creation of a schedule chart. In *Proceedings of the 6th international conference on Multimodal interfaces*, pages 329–330. ACM Press.

Niekrasz, John and Purver, Matthew (2006). A multimodal discourse ontology for meeting understanding. In Renals, Steve and Bengio, Samy, editors, *Machine Learning for Multimodal Interaction: 2nd International Workshop, MLMI 2005, Edinburgh, UK, July 11-13, 2005, Revised Selected Papers*, volume 3869 of *Lecture Notes in Computer Science*, pages 162–173. Springer-Verlag.

Niekrasz, John, Purver, Matthew, Dowding, John, and Peters, Stanley (2005). Ontology-based discourse understanding for a persistent meet-

ing assistant. In *Proceedings of the 2005 AAAI spring symposium on persistent assistants*, Stanford.

Passonneau, Rebecca J. and Litman, Diane J. (1997). Discourse segmentation by human and automated means. *Computational Linguistics*, 23(1):103–139.

Pevzner, Lev and Hearst, Marti (2002). A critique and improvement of an evaluation metric for text segmentation. *Computational Linguistics*, 28(1):19–36.

Ries, Klaus (2001). Segmenting conversations by topic, initiative and style. In *Proc. of ACM SIGIR Workshop on Information Retrieval Techniques for Speech Applications*.

Shriberg, Elizabeth, Dhillon, Raj, Bhagat, Sonali, Ang, Jeremy, and Carvey, Hannah (2004). The ICSI meeting recorder dialog act (MRDA) corpus. In *Proceedings of HLT-NAACL SIGDIAL Workshop*.

Siegel, Sidney and N. J. Castellan, Jr (1988). *Nonparametric Statistics for the Behavioral Sciences*. McGraw-Hill, 2nd edition.

Tür, Gökhan, Hakkani-Tür, Dilek, Stolcke, Andreas, and Shriberg, Elizabeth (2001). Integrating prosodic and lexical cues for automatic topic segmentation. *Computational Linguistics*, 27(1):31–57.

Wrede, Britta and Shriberg, Elizabeth (2003). Spotting "hot spots" in meetings: Human judgements and prosodic cues. In *EUROSPEECH 2003*, Geneva.