Extracting and Using Implicit Supervision to Automatically Improve Meeting-Understanding

Satanjeev Banerjee

CMU-10-016

Language Technologies Institute
School of Computer Science
Carnegie Mellon University
5000 Forbes Ave., Pittsburgh, PA 15213
www.lti.cs.cmu.edu

Thesis Committee:
Alexander I. Rudnicky, Chair
Carolyn P. Rosé
Geoff Gordon
Dan Bohus, Microsoft Research

*Submitted in partial fulfillment of the requirements
for the degree of Doctor of Philosophy
in Language and Information Technologies.*

To my wife, my family, my teachers, and my friends.

Abstract

Automated systems are often trained with expert-generated labeled data; however, such data is typically expensive to procure. For interactive systems, another potential source of labeled data is the system's human users. Under the right conditions, the user's everyday interactions with the system can be interpreted as positive or negative feedback, and used to improve the system's performance. We call this form of supervision *implicit* because the labeled data obtained from the user are *by-products* of his task-oriented interactions with the system. In this thesis, we explore implicit supervision within the domain of *meeting understanding* – the automatic extraction of salient information from speech-based human-human meetings.

Interpreting a user's actions as feedback can be difficult when the inputs and outputs of the user's actions are very different from the desired labeled data. This dissertation examines two tasks within meeting understanding that have such a characteristic. First, we investigate the task of identifying the agenda item being discussed at any given time during a human-human meeting. The system's desired labeled data – meetings segmented into agenda items – are very different from the inputs (speech) and outputs (notes) of the only available human action in meetings – note-taking. Second, we investigate the task of identifing noteworthy utterances in meetings. Here too, the desired labeled data – utterances labeled "noteworthy" or not – are different from the meeting notes.

For agenda item identification, we develop a novel general approach to redesigning the note-taking interface to enable extraction of segmented meetings based on participants' note-taking actions. For noteworthy utterance identification, we develop a simple approach to label an optimal set of utterances as noteworthy, using only the speech and notes of previous meetings. In both cases, we show that automatically extracted data compares well with both manually labeled data as well data extracted through other unsupervised means. In addition, we develop a novel notes-suggestion system based on the automatically extracted noteworthiness labeled data. Through offline evaluation and user studies we show that such an end-to-end automated note-suggestion system can help meeting participants take more and better notes, while reducing their note-taking effort.

Acknowledgements

I am very grateful to my advisor Alex Rudnicky for his support throughout the years of the PhD – from the time I took his Spoken Dialog Lab, through the early days of the CALO project, through the challenging days of the proposal, and through today. Every step of the way, he has been all one could ask for from an advisor – there to help me whenever I needed help, ready to let me do my own thing whenever I wanted to. In particular, I am very grateful for his willingness to let me work with any faculty I wished to, on sometimes unrelated projects even, all in the name of my education. I have learned a tremendous amount from Alex – from thinking about research, to creating a research plan, writing, presenting, etc. I will forever owe Alex a debt of gratitude.

I have also been immensely helped by my committee. Carolyn has practically been a second advisor to me – and not just since the proposal. I took her "HCI Methods in LTI" course many years ago, and since then she has always been willing to help me with my research, my user study designs, my papers, going over and above in taking time out for me. Dan has been a friend to me before he became my external committee member as well, and I have learned a lot from him. To him, I owe much of whatever presentation skills I have now. Dan has always had a patient ear, and been very willing, both before and after becoming a committee member, to discuss my research ideas with me. Geoff has been very kind to me, and generous with his time and advice. My dissertation and this document is better because of all of you.

I am very grateful to all my friends in and around the LTI for their help – both technical and friendly: Antoine Raux, Brian Langner, David Huggins-Daines, Long Qin, Arthur Chan, Jahanzeb Sherwani, Betty Cheng, Thomas Harris, Stefanie Tomko, Moss Chotimongkol, Scott Judy, Mohit Kumar, Dipanjan Das, Ziad Al-Bawab, Matt Marge, Aasish Pappu, Alok Parlikar, Abhimanyu Lad, Shilpa Arora, Mahesh Joshi, Anagha Kulkarni, Abhay Harpale, and Barkha Harpale. At one time or another *every one* of you guys have been participants in one or more of my experiments; for this and many other reasons I thank you. Outside of CMU, I would like to call out Siddharth Patwardhan, University of Utah '09, for his constant help and encouragement over the course of the PhD; the PhD would have been less fun without you <):). And thanks, Karen and Ed, for the support and the laughter.

I would like to acknowledge the various professors I have had all my life because of whom I have been able to make it thus far. Alon Lavie, Alan Black, Jack Mostow, Paul Rybski, Joe Beck, and all other professors at Carnegie Mellon University; Ted Pedersen and all the professors at University of Minnesota Duluth; Professor Nalawade and others at D. Y. Patil, University of Pune; and Abhijit Sircar from St. James' School – thank you for all the knowledge and the guidance that you have given me so freely.

Finally I would like to thank my immediate and extended family. To my in-laws: Amma, Daddy, Bhaiya, Bhabhi, Lavanya, and Vir – thanks for welcoming me into your family as one of your own even though I was a starving student, and thanks, too, for never putting pressure on me to finish. To Mummy, Babuji, and Mamoni, wherever you are, thanks isn't enough so I won't say it, but I know you are proud that I got here eventually.

And finally to Nams, thanks for putting up with a PhD's life, thanks for putting your own life on hold while you waited for me to finish, and thanks for everything you do that lets me do what I need to do.

Contents

Chapter 1

Introduction

In recent years, many automated systems have been developed that exhibit what is typically called “intelligent behavior”. Some examples of such intelligent systems include web search engines that rank documents on the World Wide Web according to their relatedness to a short string of words; automated phone systems that use speech recognition and dialog management techniques to understand a caller’s reasons for calling and attempt to assist him; recommendation systems that use customers’ past purchases to suggest new movies, books, and other products that they might be interested in; email filtering systems that automatically separate legitimately important email from unsolicited “junk” email, etc. Developing such systems requires not only engineering expertise, but also expertise about the domain in which the system will function. Such domain expertise is sometimes provided in the form of rules that specify the system’s behavior when faced with various inputs. More commonly, domain expertise is provided in the form of “labeled data” that pair input data with their expected system output. For a speech recognition-based system, for example, labeled data is typically made available in the form of recorded audio and their text transcripts. For an email filtering system, labeled data can be presented in the form of example emails paired with either the label “Junk” or “Not junk”. For a web search system, labeled data could include example rankings of documents for a given query and a given universe of documents. Using such labeled data, statistical models can be trained to learn the mapping from inputs to outputs, and used as the basis for the intelligent systems.

A persistent problem of such an approach to developing intelligent systems is that of ensuring that the system “generalizes to new data”, that is, it is able to perform adequately when faced with all possible variations in inputs once the system has been deployed. The usual approach to improving a system’s generalizability is to collect as much human-generated labeled data as possible, with the hope that most of the possible variations in the inputs are represented in the collected data. While theoretically sound, such an approach is often expensive because of the human labor involved in collecting labeled data.

For a system that is likely to be used primarily and repeatedly by a single user (or a small set of users), another approach to improving its performance is to *adapt* to the typical inputs and expected outputs of that particular user. In this approach, instead of performing well on *any* input, the system's goal is to perform well on the inputs of a single user, and tailor its performance to suit his idiosyncratic needs. By not having to generalize beyond one user or a small set of users, limited labeled data may suffice. Such an approach is likely not viable for a system that is typically used by many different people – such as an automated phone-call answering and routing system – but is potentially viable for a system that is meant for repeated use by a single or a small number of users, such as desktop dictation software.

In such an approach, the user of the system is the best source of idiosyncratically labeled data. Acquiring labeled data from him is a challenge, however. Unlike the system engineer, end users of systems must be incentivized to provide labeled data to improve the system. Typically, this incentive is provided in the form of a promise of system improvement. For desktop dictation systems, for example, users are requested to train the system before using it for the first time; users are told that such training will lead to vastly improved speech recognition. While such an explicit request for labeled data is appropriate when the amount of data needed is small or when data acquisition is a one-time task, it is less feasible if a large amount of labeled data is needed or if the need is ongoing in nature. In such cases, automatically extracting labeled data from the user's interactions with the system itself may be more likely to consistently generate data that can be used to improve the system's performance. We call such labeled data acquisition "implicit supervision".

We define supervision extracted from a particular human action as being "implicit" if the human action is directly related to the task he wishes to perform. For example, extracting training data from a human's utterances in order to improve a dictation system's speech recognizer will be considered "implicit supervision" if the utterances were spoken during a dictation session, and "explicit supervision" if they were spoken during a training session with the explicit goal of improving the speech recognizer. The advantage of implicit supervision is that the human does not need to be separately incentivized to provide the labeled data. Instead, the labeled data are a *by product* of his everyday task-directed interactions with the system. The technical goal of this thesis is to explore different ways of extracting implicit supervision from humans' interactions with a system.

We perform our experiments within the domain of *automatic meeting understanding*. While the Oxford English Dictionary defines a meeting broadly as "the act or an instance of assembling or coming together for social, business, or other purposes", in this thesis we focus on meetings that are speech-based, involve two or more humans, and are conducted in real-time. At many such meetings, information is presented, progress is discussed, negotiations are done, and decisions are made. Consequently meetings form a very important part of the work place. In 1995, it was estimated that Fortune 500 companies alone held between three and four billion meetings annually [41]. In the same study, managers self-estimated that their productivity at meetings was typically very low. We show in this thesis that part of this inefficiency rises from the difficulty of organizing and accessing information discussed at meetings. The human-impact goal of this thesis is to develop systems that help humans organize information discussed at meetings for better future retrieval.

Specifically, we explore two different automated meeting-understanding tasks in this thesis: identifying the agenda item being discussed at any time during the meeting, and helping meeting participants take notes. The first task can help humans navigate through a recorded meeting and quickly find portions of interest, while the second can help meeting participants take better and more detailed notes so that their future information needs are more likely to be met than if they had no assistance at all.

Through these two tasks we explore two different ways of extracting implicit supervision from human-system interactions. For both tasks we extract labeled data from meeting participants' speech and notes recorded during previous meetings. For agenda item detection, we extract meeting segments labeled with the agenda being discussed during that segment, while for assisted note-taking, we extract utterances labeled as "noteworthy" or "not noteworthy" in order to provide note-taking assistance in future related meetings. We evaluate the quality of the extracted labeled data, and show that it meets or exceeds the quality of labeled data created by third-party human annotators. For assisted note-taking, we develop a note-suggestion system based on the automatically extracted labeled data and show, through offline experiments and online user studies, that such a system can help meeting participants create better and more detailed notes, while reducing their note-taking effort.

Chapter 2

Previous Work

In this chapter we present an overview of previous approaches to both meeting understanding and supervision extraction.

2.1 Meeting Understanding

The goal of *meeting understanding* is to develop tools that help humans extract more value out of meetings than they would have on their own. Such tools fall into one of two main categories:

- Techniques that help meeting participants easily record meetings, and view the recordings afterwards.
- Techniques that automatically extract information from meetings to make navigation less important, or potentially unnecessary.

The technical focus of this thesis is on approaches to automatically extract information from meetings; however, we briefly review meeting recording and browsing systems below as a point of comparison to two meeting recording and assistance systems that we develop to enable the research in this thesis – the MockBrow meeting browsing system (Section 4.4.2) and the SmartNotes meeting assistance system (Section 5.4).

2.1.1 Meeting Recording and Browsing

Several approaches have been presented and evaluated that allow meeting participants to create a rich record of the meeting. The Distributed Meetings (DM) project [12], for example, involves a system that can be used to broadcast and record meetings, and also view pre-recorded meetings, using, among other things, a panoramic camera situated in the middle of the table. The author of this project evaluated the DM by conducting a user study involving real meetings between real participants at Microsoft Research. At each meeting, a person was asked to remain absent, and to later come in and view the meeting recording using the DM viewing software, and then fill out a questionnaire. The questionnaire data provided evidence that users were satisfied with the information they received from it.

In the WorkspaceNavigator [23], the authors attempted to capture many different sources of digital information as a meeting proceeds inside a "smart room". Recording involves taking regular snapshots of different computer displays, the meeting room itself, filenames of open files and URLs of visited webpages from participant laptops, etc. Users are allowed to label the snapshots, or just mark snapshots as being important as they are being recorded. Two qualitative user studies were conducted to provide a detailed view of patterns of actual use of the technology. The authors provide convincing evidence that users were able to index and retrieve portions of meetings when needed. In both the above papers, the authors provide web-based browsing and retrieval tools to view and listen to parts of the recorded meetings.

A Meeting Browser is also presented in [51]. In addition to recording the audio and video in the meetings, this system also transcribed the audio, summarized the speech and identified both speech acts and various non-verbal cues. The recorded audio and video, and all these automatically extracted pieces of information were displayed to the user through dedicated meeting browsing software.

A lot of research has also been focused on facilitating the conduct of meetings when all the participants are not collocated (e.g.: [1], [2], [10], etc).

Such meeting recording and playback technology has several potential uses. First they can be of use to parties that were absent during the actual meeting. While such *non–contemporaneous* meeting participants may not be able to contribute to the meeting itself, they can at least benefit from the information produced at the meetings using such technology, and then be included as a more informed participant in resulting ongoing discussions. A second use of this technology is as an aid to participants' memories of past meetings. Participants forget details of meetings, or worse, have erroneous recollections of past meeting. Meeting capture and play back technology, if appropriately designed, can be used to efficiently retrieve details of past meetings. Thus this technology can be viewed as improving Organizational Memory [46], which has been shown to improve users' productivity [24].

Note that meeting recording raises several privacy and confidentiality issues. [12] reported that participants were "generally comfortable having their meetings recorded", although they caution that this could be the result of participant self selection in their pilot study. Intuitively, to make such recording technology widely acceptable, users will have to be given the opportunity to pause and/or erase portions of the recordings, and also the ability to restrict access to some recorded meetings to a small set of authorized viewers.

2.1.2 Meeting Analysis

Much work has been done in transcribing speech in meetings. Indeed, meeting speech transcription is currently one of the major bi-yearly speech recognition research challenges organized by the National Institute of Standards and Technology [35]. Examples of research done in meeting speech recognition include [55, 48, 31, 20, 49, 47].

Beyond transcribing the meeting speech, work has also been done on automatically understanding some aspect of the recorded meeting, with an eventual goal of helping humans access meeting information more easily. For example, one active line of work is in meeting summarization, an example of which is [33]. In this work the authors apply the text-based extractive summarization work – selecting utterances that should be included in the summary, without doing any further abstraction – to the domain of meetings. Using the ICSI meeting corpus, the authors use both prosodic and word-based features to perform the utterance classification. (Other examples of such work are reviewed in Chapter 6).

Instead of summarizing the entire meeting, work has also been done on extracting key pieces of information from the meeting. An example of such work is that of [37]; here the authors attempt to use shallow-features-based techniques to detect the argument structure of the meeting, and from such detection, find the action items being discussed in the meeting. Another approach to understanding the structure of the meeting is presented in [43]. In this work, the authors use *visual* information (as available through a set of panoramic cameras) to recognize the activities of meetings participants – both low-level, such as whether they are sitting or standing, and based on these activities higher-level activities such as whether a presentation is being made, whether the meeting has just started or is wrapping up, etc.

Another popular stream of work is in the field of detecting the topic of discussion at any point of time in the meeting, e.g. [7, 15, 6, 38]. (A more detailed description of this area of work is presented in Chapter 5).

2.1.3 Relationship to This Thesis

These previously investigated mechanisms for understanding meeting structure or extracting information from meetings are all examples of classical supervised or unsupervised machine learning, in which labeled data is either created by the researcher, or is not used at all. In this thesis, we explore approaches that make aggressive use of the smartest resource available to meeting processing algorithms: the meeting participants themselves. In these approaches, we extract labeled data from the participants, and then evaluate the degree to which such data improve the performance of meeting understanding algorithms.

2.2 Extracting Supervision

Much work has been done in the past to develop systems that adapt to the specific "conditions" in which they are deployed. Such conditions typically include the characteristics of the human user(s) of the system, and also the environment in which the system is used. For example, speech recognition and handwriting recognition systems can improve their accuracy by adapting to the actual end-user's speech and handwriting patterns, and a speech recognizer can also improve by learning the noise characteristics of the surroundings in which it is often used. Previous work on extracting supervision from users in order to adapt can be organized in terms of the *strategy* used to extract the information.

2.2.1 Directly Asking for Supervision

One popular strategy is to directly request the user to provide feedback. Such a training period, often called the "enrollment phase", is often used in commercial dictation systems to improve the quality of the speech recognition. E.g., both the Nuance Dragon NaturallySpeaking system and the dictation system built into the Microsoft Windows operating systems encourage users to go through enrollment before they use the system for the first time. During this phase, users are asked to read system-selected sentences into the microphone, so that the recognizer can adapt to the user's speaking style, pronunciation, etc. Such an approach is reasonable when the user's characteristics can be expected to be relatively stable over a long period of time – it is likely that the user's speaking style and pronunciation will not change drastically over time – which enables a one-time training phase.

Another class of systems that explicitly ask for user supervision is recommendation-generating websites. Web-based retailers (like Amazon.com), movie rentals (like Netflix.com), and streaming music portals (like Pandora.com) allow and indeed actively solicit users to rate the products they buy, movies they rent and music they listen to. Using this rating information, these systems can develop a model of the user's preferences, and suggest other products, movies and music that he might be interested in. Unlike the one-time speech recognizer training paradigm, these websites expect continual and substantial feedback. As a result it is likely that for some (or perhaps most) users, acquiring such direct supervision may not be feasible, and other means of acquiring the same information are needed (e.g., tracking what products customers view on their websites or buy as a proxy of their interest in those products). An overview of such web-based recommendation systems is provided in [44].

2.2.2 Corrections-Based Supervision Extraction

The strategy of implicit supervision – acquiring supervision from a user's actions – has also been explored in the past. One class of such systems provides the user with the system output, and observes his response. If he *corrects* the output, this correction data can then be used as feedback to improve the system. Such an approach is used in [8] where the user's corrections to the output of the speech recognizer is used as training data to improve the speech recognizer.

Another example of a corrections-based supervision extractor is the meeting scheduling assistant described by [52]. The authors describe an automated meeting scheduling system that presents the user with one or more possible schedules when the user receives a meeting request by email. Once the user selects one of the suggestions, the user's response is used to retrain the scheduling system. (Note that Netflix.com also gathers feedback implicitly by simply performing the task – showing the user its predicted rating for movies the user has not rated yet – and then observing the user's response – whether the user corrects or accepts the predicted rating).

A corrections-based mechanism for acquiring feedback from users to improve automatic detection of action items in recorded meetings is presented in [13]. Users are presented with automatically extracted action items after the completion of the meeting. The user's actions are then used to retrain the action item detection engine.

2.2.3 Passively Interpreting User Behavior

Work has also been done in automatically but passively interpreting a user's interactions with the system. One example of this strategy is email systems that attempt to learn how a user assigns his incoming email to folders, and then helps the user by automatically performing the assignment whenever the system is confident of the assignment. This approach is taken in [9] to learn how to assign incoming email to folders.

Another example of passive user behavior interpretation is in the field of information retrieval, where "implicit relevance feedback" is obtained when the user views a subset of the documents presented to him based on his initial search query. The documents viewed by him are interpreted as being relevant to his initial query, while the documents he ignored are viewed as being irrelevant. This feedback can then be used to modify the search algorithm. Such an approach is taken in [53], where the document titles and summaries viewed by the user are interpreted as providing feedback.

Another example of automatically and passively interpreting user behavior is showcased in [32] where the authors present an interactive computer based drawing tool that intelligently modifies the drawing being constructed based on interpretations of the human's strokes on the screen. The angle and direction of repeated strokes made by the human are interpreted as either intended to extend the current line or change its slope, and this change is then automatically reflected in the drawing.

2.2.4 Unsupervised Learning

Finally much research has been done in performing pure unsupervised learning, where minimal interpretation is done of user-generated data before it is used for system improvement. Note that sometimes there is no clear distinction between "passive interpretation" and "unsupervised learning". In both cases, the user is not explicitly asked to provide any supervision. Thus both these techniques can be considered as belonging to different points on the same spectrum.

An example of such unsupervised learning is described in [25]. In this paper the authors improve speech recognition accuracy by first transcribing the speech with the existing system, then using a confidence metric on the resulting transcription to identify the moderate-to-highly confident words, and then using those words to retrain the recognizer. A purely unsupervised approach is also taken by [38] who describe a topic detection system that clusters words into bags without any supervision.

2.2.5 Relationship to This Thesis

In this thesis, we explore two different approaches to extracting information from system users. First, we introduce a new approach to extracting supervision – redesigning the interface through which the human interacts with the system in order to make it easier to assign meaning, or labels, to the human's actions that are visible to the system. We use this approach to extract segments of meetings labeled with the agenda item discussed during those segments, described in more detail in Chapter 5. Second, we explore an approach to learning what utterances are important in a meeting by aligning participant notes and recorded speech in previous meetings. This approach is described in Chapter 6.

2.3 Another Dimension: The *Kind* of Labeled Data Requested

Yet another dimension with which past approaches to supervision extraction can be viewed is the kind of labeled data that is acquired from the user. The labeled data "types" can be grouped into the following three main groups:

2.3.1 Asking Users to Provide Categorical Labels

The goal of many learning systems is to learn how to categorize data points, that is, assign data points to one of a typically small group of categories. A natural label query mechanism for such systems is to encourage the user to perform the same task – that is, associate data points with categorical labels.

For example, systems that learn to label emails as spam or not often extract supervision from users by observing how they label their emails. This supervision can be acquired through passive observation – waiting for the user to classify whatever emails he wishes to classify as spam or not – or it can be acquired through direct requests – providing the users with particular emails and asking them to label them. Both of these strategies were modeled in the 2006 TREC Spam Track evaluation [11].

A system that learns to associate task labels with desktop resources such as files, emails, directories on disc, etc. is presented in [45]. The goal of this system is to help users quickly find related resources for a particular task after an interruption, suggest appropriate destination directories for new files, etc. Users provide supervision by labeling resources through their own initiative, or when labels are actively sought from them by the system.

“Interactive Machine Learning”, an approach to building machine learning-based systems that extract feedback from the system’s users, is presented in [14]. Here, the authors create a vision-based robot navigation system that learns to distinguish parts of the road that are safe to navigate on from parts that are not. Whenever the robot is unable to make the distinction with enough confidence, the user is asked to manually label the safe parts of the road through a pen based interface. The manually labeled pixels are then used as feedback to retrain the image classification engine. These pixels can be though of as being labeled as “safe” and “unsafe” by the system’s users.

2.3.2 Asking Users to Score or Rank Data Points

Some systems attempt to learn how to rank data points according to some unknown scoring function, rather than learning how to categorize the data points. As in the case of associating data points with categorical labels, a useful supervision extraction strategy for such systems is to encourage the user to either provide a score for the data points, or to provide either a complete or a partial rank ordering of the specified data points. The score or ranking that the human produces is then used to retrain the system’s scoring or ranking algorithm.

Feedback obtained for recommendation systems described above falls in this category. Users are sometimes requested to provide a “Like”/”Dislike” binary labeling, such as in Pandora.com for example, but sometimes, as in the case of Netflix.com, they are requested to provide a finer grain 5-level score to express their like or dislike for a particular movie. In [17], one of the implemented methods of obtaining user supervision is by asking the user to provide numerical ratings for movies on the scale of 1 to 10. (The authors implement a few other innovative feedback mechanisms that we review below).

Implicit relevance feedback in information retrieval described above can also be considered a form of ranking where users of web-search are effectively ranking documents that they click on higher than documents they do not click on. A review of a large number of such relevance feedback techniques in the literature is presented in [42].

2.3.3 Other Label Query Mechanisms

Besides the two main feedback mechanisms – labeling and ordering/scoring data points – some authors have experimented with more innovative approaches of obtaining feedback from users. Most of these approaches involve extracting feedback from the users not directly on the function that the system is attempting to learn, but on intermediate knowledge or "features" that can indirectly help the function that the system is trying to learn.

For example, in [17] paper described above, the system learns users' movie preferences not only by asking the user to provide a numeric rating for movies, but also through two other feedback mechanisms:

1. Users are asked to provide a numeric rating for the applicability of one or more "features" for the movie. Users can either choose from a set of pre-existing features for the movie, or create new features. E.g., the user may specify that "robots" is a feature of the movie "Star Wars", and that it has an applicability of 9 on a scale of 1 to 10, where higher numbers imply increased applicability.

2. Users are also required to provide a numeric opinion of that feature in that movie. For instance, the user may specify that his opinion of the feature "robots" in the movie "Star Wars" is 4 on a scale of -5 to +5, where higher number imply more "positive" opinion of that feature in that movie.

Observe that these two feedback mechanisms do not directly result in a rating of the movie. Instead these feedback mechanisms result in creating information about the movie (features that are applicable to the movie), and about the user (the user's opinion of different features of this movie). Thus this form of feedback can be thought of as creating/influencing intermediate knowledge that can indirectly help the system improve its movie recommendations.

Similarly, in [18], the authors present an innovative approach to acquiring feature-feedback from users for document classification. Users can not only label (clusters of) documents with the category the documents belong to, but also label individual terms as being indicative of a document category or not. Labeling keywords is an example of acquiring feedback through labeling, but the feedback is being sought on features instead of the actual data points whose labeling the system needs to learn – the documents. Keyword feature based feedback to help text categorization is also explored in [39], while in [26], the authors explore keyword based feedback to learn to identify sentences in reports that are important to include in a summary.

2.3.4 Relationship to This Thesis

In this thesis, we acquire two different kinds of labeled data. In Chapter 5, we describe a process of acquiring labeled data in the form of meeting segments (start and stop times within a given recorded meeting) labeled with the agenda item discussed during that segment. Observe that this is neither a labeling nor a scoring problem – we need to acquire from the user the start and end times of the meeting segment. In Chapter 6 we acquire labels on utterances: either "noteworthy" or "not noteworthy". Since it is infeasible to actually ask meeting participants to label utterances, we infer these labels from other available information.

Chapter 3

Overview of Thesis Research

3.1 Introduction

The aim of the research conducted in this thesis is to develop approaches that extract the supervision implicit in humans' interactions with systems. We conduct this investigation within the domain of meetings, exploring simple yet effective ways of extracting high-quality labeled data from meeting participants' notes to train and improve two different automatic meeting-understanding tasks: detecting the agenda item being discussed at any time during the meeting, and automatically identifying noteworthy utterances that should be included in the meeting notes.

As mentioned earlier, we define supervision extracted from a particular human action as being "implicit" if the human action is directly related to the task he wishes to perform with the system. Extracting implicit supervision, as opposed to asking humans to directly provide the system with feedback, is a more reliable way of acquiring labeled data because the human does not need to be separately incentivized to provide feedback – the labeled data is a *byproduct* of the user's task-based interactions with the system.

In the previous chapter we reviewed different past approaches to extracting supervision by interpreting user actions as feedback. For most of these approaches, interpreting a user's actions is relatively straightforward because the user's actions are similar to the system's intended actions. We will define humans' actions and system's actions a "match" if they both take the same inputs and produce the same outputs. When there is such a match, using the data generated by the humans' actions does not require any sophisticated interpretations.

For example, if the system's expected action is to assign the label "junk" or "not junk" to emails, and the user can also perform the same task manually, then the two actions are a match, and the user's actions directly produce labeled data without further interpretation. Note that not *all* of a user's interactions with the system need to be a match – only the action that is being targeted for supervision extraction. In a dictation system for example, the user's typical visible action is to speak, while the system's action is to transcribe, and these two actions do not match. However, if the human is allowed to *correct* the system's output, and if this correction action is being used for supervision extraction, then there is indeed a match between the actions. Note that by design, system and human actions are usually a match when the human is explicitly asked to provide labeled data. When users provide movie ratings to a recommendation system, for example, his inputs (a movie) and outputs (a rating score) are exactly the same as the recommendation system's inputs and outputs.

In some cases there is not a complete match between the system's and the human's actions, but interpretation is still relatively straightforward. For example, in web-based information retrieval systems, the user's clicks on a documents are relatively easily interpreted as signs of his interest in those documents, even though the human's actions (clicking on documents) does not directly match the system's actions (ranking documents).

In this thesis, we are interested in exploring cases where the human's actions from which labeled data must be extracted do not match the system's actions. This is particularly true in the case of meeting understanding where typically the only visible human action is note-taking. From such note-taking, we wish to extract labeled data for two separate meeting-understanding tasks, as follows.

3.2 Agenda Item Detection

3.2.1 Motivation and Labeled Data Needed

The goal of this system task is as follows: given a meeting and an agenda of that meeting, identify the portions of the meeting during which each agenda item was being discussed. Such a segmentation and labeling of the meeting can help meeting participants navigate recorded meetings easily, and quickly find the regions of the meeting that are of interest to them.

The ideal labeled data for training such a segmentation algorithm is a set of meetings that are fully segmented into the agenda items discussed during each meeting. Getting such data explicitly from meeting participants is likely infeasible. At the same time, observe that there is no match between the system action (segmenting meetings) and the only available human action – note-taking. Thus, interpreting meeting participants' note-taking to extract meetings labeled with agenda items is a challenge. Our approach to doing so is to redesign the system interface through which meeting participants take notes in order to enable such interpretation. We describe this interface design in general terms in the next section, and describe it in more detail in Chapter 5.

3.2.2 Designing the Interface to Extract Implicit Supervision

In order to design a system's interface to extract implicit supervision in the general case, we propose the following three step recipe:

1. Identify the kind of labeled data that is needed to improve the system.
2. Identify a relationship between a user action and the function that needs to be learned.
3. Build an interface that takes advantage of this relationship.

To explain these steps through an example other than the agenda item detection task, let us see how these steps can be used to explain the design of the Peekaboom game [50]. This is a 2 person collaborative game where one person is given an image (say a picture of a dog and a cat) and a word (say "dog"), the other person is initially shown a blank screen, and the goal of the game is for the second person to guess the word "dog". The goal of the first person is to help the second person guess the word as quickly as possible, but his only means of communication with the second person is to incrementally reveal small portions of the image to him. It is intuitively obvious that to play this game well, typically the first person will reveal parts of the image that contain the dog. The authors show that a bounding box around the image of the dog (that is, the smallest box that contains the entire image of the dog) within the full image can be obtained by aggregating the portions of the image revealed by different pairs of gamers playing the game with the same image/word pair. The design of this game can be explained through the three steps identified above, as follows:

1. Identify the kind of labeled data that is needed to improve the system: The "labeled data" that this game collects from human actions is bounding boxes around specific objects within images. Although the authors do not construct an automatic image segmenter, the data that they collect can be used to train such an image segmentation algorithm.
2. Identify a relationship between a user action and the function that needs to be learned: The relationship between user actions and the bounding box creation is:
   (a) When asked to show parts of the image that will best help their partners guess what object is being referred to in the image, game players will click on the part of the image that contains the specified object.
   (b) Different players playing with the same object/image pair will click on slightly different points of the object, such that all the clicks together will cover the entire object.

Observe that both of these two facts need to be true for the labeled data extraction to be viable. If for example, gamers can reveal the word to their partners through some other means (say through a text communication) then a bounding box cannot be extracted. Similarly, if all game players click on the exact same point in the image, again a bounding box cannot be constructed.

3. Build an interface that takes advantage of this relationship: The authors create a two-person game as described above. Additionally, to ensure that the two facts above hold, they ensure that random partners are paired, and that the partners have no other means of communication with each other.

Note of course that for a given interactive system and a given target function, these three steps do not guarantee that labeled data can be acquired – it may not always be possible to identify a suitable user action to harness for labeled data acquisition.

3.2.3 Approach Applied to Meeting Agenda Identification

We now turn our attention to applying the above three-step recipe to developing a meeting note-taking interface that enables the extraction of labeled data in the form of meetings segmented into agenda items.

1. Identify the kind of labeled data that is needed to improve the system: As mentioned above, the ideal form of labeled data for the task of automatic meeting segmentation is examples of meetings segmented into the agenda items being discussed during the meeting. Specifically, the labeled data we require is in the form of pairs of relative start- and stop-times that each signify the start and end of discussions on a particular agenda item.

2. Identify a relationship between a user action and the function that needs to be learned: We target meeting participants' note-taking during meetings as the user action from which the system will passively extract labeled meeting segments. The relationship between the lines of notes taken by the user, and agenda-labeled meeting segments rests on the following two observations:

   (a) Most lines of notes taken in a meeting refer to a particular discussion that occurred during a particular segment of the meeting, and the notes can be said to belong to the same agenda item as the discussion.

   (b) There are strong temporal and textual relationships between a meeting segment containing a spoken discussion and the line of note referring to that discussion: Often, the line of note occurs a short time after the discussion, and contains text that overlaps significantly with the discussion.

Using these two observations we posit a chain of interpretation as follows: individual lines of notes belong to particular agenda items, and (typically) occur a short time after that piece of information was discussed. Thus, we can identify the agenda item of a particular short meeting segment, by identifying the agenda item of the *note* taken shortly after that meeting segment. We identify the agenda item of the note by suitably designing the interface.

3. Build an interface that takes advantage of this relationship: We develop SmartNotes (described in more detail in Chapter 5) – a note taking interface that provides the meeting participant with the ability to associate individual lines of notes with the agenda item that that line of note belongs to. By automatically time-stamping each line of note, we can thus propogate the agenda label to segements of the meeting and thereby acquire the target labeled data.

In Chapter 5 we describe this interface in more detail, and extract and evaluate labeled data in real meetings.

3.3 Note-Taking Assistance

3.3.1 Motivation and Labeled Data Needed

The second task that we explore in this thesis is that of helping meeting participants improve their notes in meetings. The motivation for this task is that while most meeting participants rate note-taking as a hard task, the existence and the quality of notes in a meeting are strong predictors of the likelihood that meeting participants' future information needs will be fulfilled (we show this in Chapter 4).

We break down the task of making note-suggestions to meeting participants into the following sub-tasks: identify utterances containing noteworthy information (which we will call "noteworthy utterances") and then suggest the contents of those utterances as notes to the meeting participants. Note that this formulation is similar to that of *extractive* meeting summarization; this connection is described in more detail in Chapter 6.

In order to train a noteworthiness detector, the ideal form of labeled data needed is in the form of utterances labeled as either "noteworthy" or "not noteworthy". While this is a simple binary labeling, obviously meeting participants cannot be asked to explicitly provide such labeled data (e.g. it is infeasible to ask meeting participants to manually label each individual utterance in the meeting once the meeting is complete). Like with the agenda item detection task, we propose to extract labeled data for this task from meeting participants' notes.

While it would appear that the ultimate actions of both the human and system are a match – in some sense they both take meeting speech as inputs and attempt to output notes – the immediate task of the data extractor is to label utterances as “noteworthy” or “not noteworthy”, and thus the human and the system tasks are not a match.

3.3.2 Labeled Data Extraction Approach

The noteworthiness of an individual utterance is a very subjective matter, and is likely to depend on the meeting participants involved, the topics of discussion, the form of the meeting (brainstorm versus progress-report based), etc. Thus this task lends itself to adaptation. Specifically, we aim to adapt the noteworthiness detection algorithm to a particular *series* of longitudinal related meetings, for example the regular meetings held by the members of a particular project. The goal is to learn the idiosyncratic topics and note-taking habits of a particular group of participants. In progress-report based meetings, for example, meeting participants may be particularly inclined to include action items and reports on previous meetings’ work in the notes, whereas for brain-storming type meetings, participants may be inclined to note down new ideas or examples of suggested reading etc. Thus our goal is to automatically extract labeled data from the notes that meeting participants have taken in previous meetings, in order to train a noteworthiness detection model and use it to make note-suggestions to meeting participants in future meetings in the same meeting sequence.

In order to extract utterances labeled “noteworthy” or “not noteworthy” from meeting participants’ notes, we construct the following chain of interpretation. We show in Chapter 6 that for a substantial fraction of the notes taken in meetings, there is a strong overlap between the text of the notes and the text of utterances in the meetings. That is, in many cases, participants write individual lines of notes that are close approximations of certain utterances that someone just spoke. Intuitively, these utterances, if identified, can be considered as “noteworthy” (since their contents were included in the notes) and other utterances “not noteworthy”. More generally, we develop an algorithm in Chapter 6 to identify a subset of the utterances of each meeting that together have optimal overlap with the notes written in the meeting, and label these utterances as “noteworthy”; we label all other utterances in the meeting as “not noteworthy”.

(Note that not all lines of notes that participants write necessarily are verbatim copies of individual utterances, and nor does the supervision extraction algorithm expect this to be the case. Participants sometimes write notes in their own words, and sometimes even notes that are unrelated to the discussion. The supervision extraction algorithm is robust to such phenomena, as long as there is *some* portion of the notes that do indeed have high overlaps with noteworthy utterances).

In Chapter 6, we describe this supervision extraction algorithm, and evaluate the extracted data directly by comparing to manually labeled data and indirectly by evaluating noteworthiness detectors trained on the data. In addition, in Chapter 6, we present a note-taking user study undertaken to evaluate a notes suggestion system developed on the basis of the automatically extracted labeled data.

3.4 Thesis Statement

The aim of the research program conducted in this thesis is to **develop approaches to automatically extract the supervision implicit in a human's interactions with a system, when the human's and the system's actions do not match.**

We conduct this research within the domain of automatic meeting-understanding, and particularly address two different tasks – extracting labeled data for agenda item-based segmentation of meetings and extracting labeled data for identifying noteworthy utterances in meetings.

3.5 Roadmap for the Rest of the Thesis

In the rest of this thesis, we present the research conducted.

- In **Chapter 4**, we present results of user surveys that establish the kinds of information meeting participants need to retrieve from meetings. This knowledge forms the motivation for both the agenda item detection and the assisted note-taking tasks.

- In **Chapter 5**, we describe the approach to extracting labeled data for the agenda detection task from meeting participants' notes. We describe the design of the interface, and evaluate the data collected from real participants in real meetings against manually labeled data.

- In **Chapter 6**, we describe the process of extracting implicit supervision from meeting participants' notes in order to create a notes-suggestion system. We explore different manual labeling schemes, and weigh their advantages and disadvantages. We then present evaluation of the data extracted.

- Finally in **Chapter 8**, we summarize the thesis, present our contributions, and offer some thoughts on the most promising areas of future work.

Chapter 4

What Information do Meeting Participants Want?

4.1 Motivation

The goals of this chapter[1] are two-fold: find out through user surveys the kinds of information that meeting participants need from previous meetings, and explore the impact of topic segmentation and labeling on navigation through recorded meetings. These tasks are both undertaken with the goal of establishing the motivation for focussing on meeting-understanding as a fruitful area of research.

The first goal in this chapter is to explore the problem of reconstructing information discussed at a meeting. How often do busy professionals need to reconstruct details of past meetings? What kinds of documents do they typically have access to? Are those documents sufficient? What kinds of information are they typically seeking from past meetings? How much time does it take to do the reconstruction? To gain an understanding of these issues, we have run an interview–based survey with 12 faculty members at Carnegie Mellon University. We have chosen this user population since university faculty typify professionals who's lives are dominated by meetings. Interviewees were asked to narrate specific instances of situations when they were trying to catch up on a meeting that they had missed, or were trying to reconstruct forgotten details of a meeting they had attended in the past.

Our second aim is to gauge how helpful topic–level annotations are to meeting browsing. For the purposes of this study, we use the MockBrow, a multi-channel audio and aligned-annotations viewer developed in-house at Carnegie Mellon University, to annotate

[1] The contents of this chapter have been published as [5].

meeting records by labeling different portions of the meeting with their general “topic of discussion”. We also mark different parts of the meeting with what the “state” of the meeting was (discussion / presentation / briefing). The meeting is not annotated with any other information, such as ontology based structural information as in [4]. We hypothesize that a human can retrieve information from a meeting faster if he is armed with topic annotations versus if he is not. In section 4.4 we present a user study aimed at quantifying the extent to which this hypothesis is true.

4.2 Related Work

Closely related to our goal is the survey [28] of potential users of a meeting browser, conducted as a part of the IM2.MDM (Interactive Multimodal Information Management, Multimodal Dialogue Management) project. The goal of this survey was to elicit a set of questions that users may ask of an intelligent meeting browser. Participants were asked to imagine themselves in one (or more) of four roles – a manager tracking employee performance, a manager tracking project progress, an employee who has missed a meeting, and a new employee – and to then think of all the questions they expect to ask about the meeting or set of meetings that the meeting browser may have access to. While this survey provides some broad insights, it differs from ours in both its goals and its methodologies, especially in that it does not adhere to strong HCI methodology for survey research. One of the goals of our survey is to assess how *useful* a meeting browser would be, how urgently its need is currently felt by busy professionals, and in what range of their actual situations they could potentially benefit from the use of a meeting browser. In contrast, the survey reported in [28] makes the implicit assumption that if busy professionals had access to an intelligent meeting browser, they would indeed use it. In our survey interviewees were asked to recall recent instances of actual situations. The resulting analysis of the interviews is therefore grounded in real experiences as opposed to potentially erroneous generalizations. The survey questionnaire in [28] asked participants to imagine themselves in a situation they have never been in before, namely, in possession of a system using which they could “ask questions about the actual content of the meeting”. Thus, it is not clear how many of these questions would indeed be asked by users of a future meeting browser in actual use.

4.3 User Survey

4.3.1 Goals

Efforts towards creating a meeting recording and play–back system can clearly drive research on a large number of fronts including speech recognition, spoken language understanding, vision–based gesture recognition, multi–modal information integration, etcetera. Here, however, we are interested in exploring the *need* for such a meeting browsing application. Intuitively, such an application would be useful to busy professionals who need to catch up on missed meetings or recall forgotten details of meetings they have attended in the past. To understand how professionals currently perform these tasks we have conducted an interview based survey. Specifically, the survey was conducted to find answers to the following questions:

- How often do busy professionals miss important meetings that they need to catch up on?
- How often do users need to reconstruct forgotten details of meetings they did attend?
- What kind of information/documents do they typically have access to in each of the above two cases?
- What kind of information do they typically seek?
- What processes do users currently employ in obtaining this information?
- How effective are these processes in terms of accurately retrieving the desired information?
- How costly are these processes in terms of time/energy spent on retrieving the information?

4.3.2 Survey Methodology

Our survey was based on face–to–face interviews conducted with 12 faculty members at Carnegie Mellon University. We chose faculty members since they attend many meetings as a part of their daily routine, and would be the likely targets of a meeting recording and playback application. Since not all missed meetings are important enough to bother catching up on, we defined a meeting as *important* if the interviewer felt he would indeed make an attempt to find out about it if he missed it. Interviewees were asked to describe instances of two kinds of situations: situations when they had missed important meetings, and those in which they were trying to recollect details of a meeting they had attended in the past. For each instance, interviewees were asked to name and describe the meeting artifacts they had access to, what specific pieces of information they were seeking about the meeting, whether and how they found the information, whether they were satisfied with the information they did find, etc. To avoid bias, interviewees were not informed about the reasons for this interview until the very end of the interview. To ground the interview in real experiences, interviewees were strongly and repeatedly encouraged to avoid replying in potentially erroneous generalities, and instead were asked to recall specific situations from their experiences in answering questions.

4.3.3 Analysis of Non-Missed Meetings

The 12 interviewees reported a total of 19 instances of situations when they were attempting to recall details of a meeting they had attended in the past (1 interviewee reported no such instances, 3 interviewees reported 1 each, and the rest reported 2 each). Interviewees were asked to report both when they were attempting to recall details of a past meeting, as well as when the meeting took place (which was normally within the past few months).

Information Sought from Meeting:

Interviewees were asked to specify the information they were attempting to reconstruct about the meeting; table 4.1 lists the frequencies of the various categories of information sought across all the instances of non–missed meetings reported by the interviewees. Note that interviewees were not shown the categories listed in the table, but were simply asked to recall all the pieces of information they were seeking about the meeting. These answers were later manually clustered into the groups in table 4.1. For example, the category *Specifics of the discussion on a topic* include questions like “What was the name of the algorithm we discussed?”. These categories are not directly comparable to the questions generated by the study in [28] which asked respondents to visualize scenarios where they had *missed meetings*.

While interviewees were not specifically asked to explain why they needed the information they were seeking, for several of the 7 instances of the category *What the decision was regarding a particular topic* interviewees spontaneously mentioned that the reason they were attempting to recall the decision was not because they thought they had forgotten the detail, but because their recollection of the detail differed from that of another co-participant of that meeting. We believe that this phenomenon is an important motivation for meeting recording and play–back technology.

Reconstructing from Available Documents:

The interviewee was asked to list the documents he had access to while he was attempting to make the reconstruction; table 4.2 lists the documents. In 14 of the 19 meetings, the users had access to notes taken at the meeting, typically the notes they had taken during the meeting. In the remaining instances, interviewees had not taken notes at the meeting, and further did not have access to notes taken by any other meeting participants. Interviewees were also asked to rate on a scale of 0 to 5 whether the piece of information they sought

Information sought	# meetings
Specifics of the discussion on a topic	11
What the decision was regarding a particular topic	7
What task someone else was assigned	4
Who made a particular decision	2
What the participants' reactions were to a particular topic	1
What the future plan is	1

Table 4.1: Information Sought from Meeting.

Documents interview had access to	# meetings
Notes	14
Nothing	2
Minutes	1
PowerPoint Slides	1
Excel Sheet	1
Project proposal document	1
Whiteboard content	1
Email	1

Table 4.2: Documents the Interviewee Had Access To.

about the meeting was satisfactorily answered by the meeting documents they had at their disposal, where 0 implied their question remained unanswered, and 5 implied they were completely satisfied with the answer they got. The average rating was 3.0 (std. dev.: 1.7).

Additional Steps Taken to Find Information:

Interviewees were asked what additional steps (besides perusing the meeting documents) they took to find the information they needed from the meetings; Table 4.3. In 8 cases the interviewees asked someone in a face–to–face conversation. This was particularly the case when the question was about a specific detail about the meeting. In 5 cases interviewees reported that they reconstructed from memory, in consultation with a meeting co–participant (note that this is not the same as simple *asking* someone else about a detail). Finally interviewees were asked to rate on a scale of 0 to 5 their quality of reconstruction of the information they were seeking, after they took the additional steps, where 0 implied they could not do any reconstruction at all. The average rating was 4.0 (std. dev.: 0.6) –

Additional step	# meetings
Asked someone face-to-face	8
Reconstructed from memory	5
Emailed someone	1
Found out in follow-up meeting	1

Table 4.3: Additional Steps Taken.

this was significantly higher than the satisfaction before perusing the meeting documents ($p < 0.0005$). 5 interviewees stated that the additional steps took less than 15 minutes, 7 said between 15 minutes and an hour, while for 2 interviewees, the additional steps took more than an hour.

Summary and Conclusions:

- Interviewees mostly sought very minutely detailed pieces of information from the meetings they had attended in the past.
- Very often the interviewees had access to notes that they could consult.
- Interviewees sometimes felt satisfied with the information they were able to retrieve from available documents.
- When the documents did not suffice, interviewees spoke to co–participants, or took other additional steps, which took up to an hour of time. At the end of these steps, interviewees largely felt that their information needs had been satisfied. Nevertheless, this does not guarantee that the information that they received was accurate since we have already established that meeting co-participants may have different recollections of what was discussed at a meeting.

4.3.4 Analysis of Missed Meetings

The 12 interviewees reported a total of 22 instances of meetings they had missed in the past that they needed to catch up on. 9 interviewees reported 2 instances each, while 1 interviewee reported 3, 1 2, and 1 none (that is, one interviewee could not recall any specific instance of an important meeting that he had missed and later attempted to catch up on). 2 of these 22 missed meetings had occurred in the week prior to the interview, while 10 had occurred within the preceding month. Thus, on average interviewees missed one important meeting a month. Of the remaining instances, 9 had occurred within six months prior to the interviews, and 1 between six months to a year before the interview. Note that based on the frequency of missed important meetings reported within the month prior to the survey, it would not be unrealistic to estimate that the population used in the survey missed on average almost 1 important meeting per month.

Understanding of Expected Meeting Content Prior to Meeting:

A person's overall understanding of the information discussed at a meeting is likely affected by his prior knowledge and expectations about the meeting before it takes place. Of the 22 reported instances of missed meetings, in 2 cases the interviewee did not receive any notification about the meeting (such as an email announcing the meeting or inviting him to it). In one of these cases the meeting had already taken place by the time the interviewee received the notification, while in the other case the meeting was a regularly scheduled one and notifications weren't usually sent out. In 12 of the remaining cases the interviewee received an agenda and/or a description of what would be discussed, while in the remaining cases he received a notification email announcing the meeting. Each interviewee was asked to rate on a scale of 0 to 5 how well he felt he knew the contents of the meeting would be, where 0 implied he had no idea what the contents would be, and 5 meant he knew exactly what would be discussed. The average rating was 3.5 (std. dev.: 1.3).

Information Sought from Missed Meeting:

For each instance of reported missed meeting, interviewees were asked why they wanted to catch up on the meeting. In particular, they were asked to list all the pieces of information they needed from each missed meetings. Table 4.4 presents the frequency of each category of information sought by the interviewees across all the instances of missed meetings. Thus in 10 of the 22 missed meetings, the interviewee wished to find out what was discussed about a specific topic. As with non–missed meetings, participants were not provided with the categories in table 4.4; the categories were constructed based on their responses. (Note that in the last 2 interviews the interviewer *did* provide these categories to the interviewees and asked them to categorize the information they sought from their missed meetings according to these categories. However, these two interviewees were also encouraged to add a new category if the given categories were insufficient. Neither interviewee felt the need to add to these categories).

Observe that the first 3 categories in table 4.4 together make up the majority of categories of information sought about missed meetings; note also that these three categories are related in that they are all concerned about seeking information about a particular topic of interest. This suggests that when a person misses a meeting, he is often more interested in catching up with the discussions regarding a specific topic of interest rather than the entire

Information sought	# meetings
What was discussed about a particular topic	10
What decisions were made about a particular topic	7
Whether a particular topic was discussed	5
Whether I was assigned a new task	4
Whether a particular decision was made	3
What decisions were made	2
If there were any new issues/announcements	2
Reasons for a decision	1
What the participants' reactions were to a particular topic	1
The backgrounds of the other participants	1

Table 4.4: Information Sought from Missed Meetings.

meeting. Thus perhaps an automated topic detection and segmentation mechanism that lets the viewer of a recorded meeting focus only on the topic he is interested in will be well received. Surprisingly, there were only a few cases in which interviewees wanted to know about progress related information like action items, announcements, etc. We conjecture that this is because most of the meetings reported were research discussion–oriented where the discussion details were perhaps more important to the interviewee than action items.

The most salient difference between the questions asked in "real world" situations by our interviewees and the questions proposed by [28] is that none of our interviewees reported asking any "hard" questions that require deep understanding of the meeting context, such as "3-Y-7-2 Why X changed his mind on issue Y in the current meeting?", or "3-N-5-9 Why was topic #5 not resolved?" ([28]). We believe this is the case because when people miss meetings, they are unlikely to be aware of enough context to ask these questions. That is, the user will not ask why X changed his mind if he is unaware that X changed his mind. Also, while it is possible to imagine that each of the large variety of questions in [28] may be asked under some circumstance, perhaps these conditions are rare enough that they did not arise in our limited set of interviews.

Understanding of Meeting Content After the Meeting:

To understand what kind of information about the meeting the interviewees could access without having to make an effort at locating the information, we asked the interviewee to list the documents he received from the meeting after the meeting took place, *without him prompting for them*. Note that by "document" we included *any* piece of information that the interviewee may have received without prompting, including oral intimation from other

Post–meeting document received	# meetings
Nothing	12
Notes	7
Minutes	3
Email from meeting participant (not official notes)	2
Document containing draft of a proposal	1

Table 4.5: Documents Received after the Meeting.

participants of the meeting, emailed documents, etc. The aim in asking this question was to understand what kind of documents are routinely sent around – one can presume that (at least a large subset) of these documents would be available even when participants have access to a meeting browsing system. Table 4.5 lists the documents reportedly received by the interviewees. Observe that in more than half the meetings, no document was received at all. Since these meetings are important enough to the interviewee that he wants to catch up on them, not receiving any information from the meeting implies that the interviewee is forced to either actively search for information about the meeting, or give up and not learn anything about the meeting at all.

Interviewees were further asked to rate on a scale of 0 to 5 how well the documents they received (if any) answered their question(s) about the meeting, where 0 meant they either did not receive any documents, or that the documents they received did not answer their questions at all. The average rating was a very low 1.7 (std. dev.: 1.9). This low number is partly explained by the fact that to a large extent interviewees received nothing from the meetings. In cases where the interviewee did receive notes etc from the meeting, they were often not sufficiently detailed to answer their questions regarding the meeting.

Additional Steps Taken to Find Information from Missed Meeting

Interviewees were asked what additional steps, if any, they took to find answers to their questions regarding the meetings. Table 4.6 shows the steps taken. Consistent with our findings about non-missed meetings, in connection with 15 meetings interviewees either asked someone face to face about the meeting, or emailed someone.

When asked how long these steps took, in 11 instances the interviewee said it took less then 15 minutes, in 5 cases between 15 minutes and an hour, and in 1 case more than an hour (this information was not collected for 5 instances). These times were self reported by the interviewees and are rough estimates only: the interviewees often reported having discussed other issues with their interlocutors while catching up on the missed meeting.

Additional step	# meetings
Asked someone face-to-face	9
Emailed someone	6
No additional steps	5
Caught up at next meeting	3
Looked up information on the Internet	1

Table 4.6: Additional Steps Taken.

Finally the interviewees were asked to rate on a scale of 0 to 5 how much they believed their information need was met after taking the additional step, where 0 meant their information need was not met at all. The average rating in this case was 3.4 (std. dev.: 1.3).

Summary and Conclusions:

- Interviewees were more interested in catching up on discussion regarding specific topics rather than the entire meeting.
- Very often no documents were received, even though the meetings were important.
- Typically interviewees had a low level of understanding of the meeting from the documents received.
- Most interviewees attempted to answer their questions regarding the meeting by asking or emailing a co-participant. This extra effort took around 15 minutes.
- Even after taking additional steps to find information about the meeting, the interviewees' levels of understanding about the meeting were felt to be far from perfect.
- Based on the fact that information is often sorely lacking about a missed meeting, we conclude that a meeting recording and playback system would be useful for busy professionals to catch up on missed meetings. Further, if the meeting recording is segmented into discussion topics, users can focus on only their particular topics of interest, thus increasing their efficiency of extracting information from the meeting.

4.4 Meeting Browsing using Topic Annotations

In this section we report on our investigation into the effect of topic- and other annotation on the time it takes for a user to retrieve information from a meeting.

4.4.1 Meeting Annotation

For the purposes of this chapter, we define meeting structure on two levels: A coarse level consisting of meeting states and participant roles, and a finer level consisting of discussion topics.

Based on observing recordings of real meetings, we developed a simple taxonomy of meeting states and participant roles. In this taxonomy, there are three kinds of meetings states, as follows: *Discussion state* which is described as being a state in which a group of two or more meeting participants are involved in quick back and forth of discussion on a topic; *Presentation state* which is described as being a state in which one particular meeting participant is presenting information to the others in a formal setting; and *Briefing state* which is described as being a state where one participant is giving information to one or more meeting participants, but without involving either the formality of the presentation state, or the quick back and forth of the discussion state. Within each meeting state, the possible roles of the meeting participants are defined as follows: within the discussion state participants may take the role of discussion participants or of observer; within the presentation state presenter or observer and within the briefing state information-provider, information consumer and observer.

Discussion topic regions are defined as all the times of the meeting that are devoted to discussing a certain topic. Although “topic” itself can be defined on various levels of granularity, in general we are interested in broad high level topics such as those that typically form different agenda items. For example “buying a printer” may be considered a topic. In well-structured meetings it is possible to split a meeting into a series of time segments, each segment containing all the discussion regarding a particular topic. Research has been done on automatically finding topics both in written texts [21] and in broadcast news [3].

4.4.2 Brief Description of the MockBrow

Meetings can be manually annotated using the meeting annotation and playback tool *MockBrow* implemented at Carnegie Mellon University. This tool allows human annotators to select a time interval within a recorded meeting and associate it with one or more labels. For example, an annotator may mark an interval of the meeting as being a “presentation”, or as belonging to the discussion on “buying a printer”, etc.

MockBrow is also intended as a platform to play back all the time–stamped media streams recorded at a meeting (such as close–talking microphone audio, video from multiple cameras, captured whiteboard markings, etc), along with all annotations generated either automatically or manually as described above. Viewers of the meeting can choose to play or “mute” different media streams during playback; for example they can choose to mute all microphones except the one closest to the speaker delivering a presentation. The interface allows viewers to quickly jump backwards and forwards within the meeting, as well as play only small portions of interest within the meeting. For example, a viewer may choose to only play back the portion of the meeting labeled as “buying a printer”. Currently MockBrow does not support any automatic searching mechanism; however it does allow users to zoom out and view the entire meeting time line (to visually note all the regions of interest within the meeting) and zoom in to specific regions for more detailed viewing.

4.4.3 User Study Goals and Methodology

In order to assess the value of developing technology to automatically annotate meeting recordings with discourse structure based annotations in order to facilitate extracting information from recorded meetings, we designed a within subjects user study.

Materials: We created an audio-video record of two ten-minute-long meetings, each involving three participants. Next we manually annotated each meeting with meeting states, participant roles and discussion topics using MockBrow. The same meeting could then be viewed either with the annotations or without. Finally we prepared for each meeting a set of five questions. (To avoid biasing the questions, the annotations were not consulted while constructing the question set). The questions were objective in nature, each with a single correct answer – this made the users’ answers easier to grade as correct or not. The questions were based on the types of questions collected as part of our survey research.

Participants: 16 Carnegie Mellon graduate students participated in the within subjects experiment. Note that while the survey was conducted on faculty members who would be the most likely target population for a meeting recording and playback application, the career status of the participants is not likely to affect the speed with which they are able to retrieve answers to questions that are provided for them from recorded meetings. Thus, for the purpose of this small study, it was sufficient to use students as participants.

Experimental Manipulation: Each participant was asked to answer the questions for each of the two meetings by viewing the meeting’s video using MockBrow while searching for the answers. In order to control for ordering effects, subjects were randomly assigned to 4 configurations in which half of the subjects viewed the first meeting and then the second meeting whereas the other half of the subjects viewed the meetings in the opposite order. Furthermore, half of the subjects viewed the annotated version of meeting one and the unannotated version of meeting two, whereas the other half of the subjects viewed the annotated version of meeting two and the unannotated version of meeting one.

In all cases, participants were encouraged to answer the questions as fast as possible, and their time to completion of each meeting viewing and question answering was recorded.

4.4.4 User Study Results and Analysis

The timing data collected in the experimental manipulation was sufficient for comparing average speed of answering questions with and without discourse structure based annotations. Results are presented in Table 4.7. The control group (the group that did not have access to the annotations and segmentations) took an average of 10.0 minutes (std. dev. = 2.6) to answer the given questions, while the experimental group took 7.5 minutes (std. dev. = 1.4). A two-tailed Student's T-Test assuming unequal variance shows that this difference in the means is significant with $p < 0.01$. Further the effect size (using the standard deviation of the control group as the denominator) is 0.94. This establishes that the difference in time taken to answer the questions when the participants could view the annotations versus when they could not is a reliable difference. Specifically, the topic segmentations and labels allowed participants to take on average 2.5 minutes less time to retrieve the answers to 5 questions in meetings that were 10 minutes long, when compared to those who had no access to such labels and segmentations.

Note that the absolute amount of time taken to retrieve information from a recorded meeting will depend on many factors, including the length of the meeting in question. For longer meetings, topic segmentations and labels are likely to have an even more significant impact on the time it takes to find answers. Without any such segmentation or labeling, participants in our experiment took as much time on average (10 minutes) as it would to simply listen to the entire meeting. At the same time, when the meeting is segmented into short topics, and correctly labeled, it is likely that the average time to answer each question will be drastically reduced from the full length of the meeting. In other words, the absolute time difference found in the above experiment may be much larger for longer meetings, and should be taken only as indicative of the potential significant gains in human information retrieval efficiency that are possible from accurate topic segmentation and labeling of recorded meetings.

In this experiment we did not record and analyze participants' browsing behavior. However it was clear that different participants used different strategies to retrieve information from the meeting record. When not provided with any annotations, some participants were content to listen to the entire meeting, while some others tried to randomly jump back and forth within the meeting. Several participants who did so complained that they had difficulty keeping track of which parts of the meeting they had already seen and which parts they had not. When provided with annotations, no participant viewed the entire meeting. Instead all participants viewed those annotated portions of the meeting they believed they would find the answer in.

Participant #	Without markup (control)	With markup (experimental)
1	11	8
2	11	8
3	11	9
4	8	8
5	17	7
6	9	8
7	9	7
8	8	4
9	8	7
10	8	8
11	9	8
12	15	10
13	8	6
14	9	7
15	11	9
16	8	6
Mean	10.0	7.5
Stdev	2.7	1.4

Table 4.7: User Study Results: Time to Completion in Minutes.

4.5 Summary

In the first part of this chapter, we reported on a user survey aimed at understanding how busy professionals such as faculty members deal with situations when they are attempting to catch up on missed meetings, or attempting to recall details of meetings they have attended in the past. One important finding was that the busy professions participating in our survey research missed on average 1 important meeting per month. Furthermore, they frequently discovered that their recollection of a discussion at a meeting was not consistent with another group member's recollection. The most frequent recourse when faced with a perceived need to recover meeting information was to talk to a group member who was at the meeting. However, even in the case where people felt satisfied with the information received from a group member, it is not clear to what extent the information they receive based on another's recollection is accurate. Thus, the survey research provides some support for the usefulness of automated methods to help users organize and retrieve information discussed at meetings. It also provides an ontology of question types that represent the types of information typically sought by our target user population.

In the second part of this chapter we have reported on a within-subjects user study performed to quantify the impact that discourse structure based annotations have on the time it takes users to retrieve the answers to focused questions from recorded meetings. We have shown that in our experiment, participants on average took 2.5 minutes less to find answers when given the annotations than when not, and that this is a highly significant difference ($p < 0.01$ using Students' two tailed T-Test, assuming unequal variance). This encouraging result provides sufficient motivation to focus work on automated topic-detection of meetings; we turn our attention to the same in the next chapter.

Chapter 5

Agenda Item Detection using Implicit Supervision

5.1 System Goal

In many task oriented meetings, an agenda is decided on either prior to or just after the meeting starts, and the conversation then proceeds by discussing one agenda item at a time. The order in which agenda items are discussed may be different from the order in which they are listed in the agenda, some agenda items may not be discussed at all, and some agenda items may be visited multiple times during the meeting. Our goal is to create a system that can detect which agenda item is being discussed at any given point of time in the meeting.

Of course not all meetings follow a clear agenda. Many meetings are held without an explicitly stated agenda, or with only one major "agenda item". Even for meetings that do have a formal agenda, sometimes the conversation goes off the agenda. When meetings are not structured with an agenda, it is difficult to decide on what the "topics" of discussion were, even for humans. For example, in [19] it was reported that when human annotators were asked to identify the topics of discussion in a meeting (and were not provided with the agenda used in the meeting, if any) and then asked to segment the meeting into topics, they achieved low inter-annotator agreement for topic segments. We focus on automatically detecting agenda items in well-structured meetings for which human annotators are in strong agreement in terms of which agenda item was being discussed at different times during the meeting. The task of detecting which agenda item is being discussed is very similar to the task of topic identification, segmentation and labeling, with topics defined as agenda items. However we continue to use the term "agenda item detection" to emphasize our definition of the sometimes abstract concept of "topic".

5.2 Motivation for System Goal

Detecting which agenda item was being discussed at any time during a meeting can help meeting participants retrieve information from recorded meetings faster. This was shown in the previous section, where the time taken by participants of a user study to retrieve answers to questions by navigating through the recorded audio/video of a meeting was statistically significantly less ($p < 0.01$) when different segments of the meeting recording were labeled (albeit manually) with the agenda items discussed during that segment. Thus, automatically performing such a labeling fits our overarching goal of helping meeting participants retrieve information from meetings faster.

5.3 Research Goal

Our research goal is to use this system task of automatically identifying the agenda item being discussed during the meeting to demonstrate the proposed approach to implicit supervision acquisition through a specially designed interface. Specifically, we aim to first use the proposed approach to develop an interface to acquire the implicit supervision, and then evaluate the labeled data so obtained from real participants in real meetings. We show that accurate labeled data for the topic-detection task can be obtained through implicit supervision provided by meeting participants.

5.4 Designing an Interface to Acquire Labeled Data: SmartNotes

5.4.1 Applying the Proposed Approach

The system task is the automatic detection of the agenda item being discussed at all times during the meeting. The input data is all the speech recorded at the meeting, time-stamped relative to the start of the meeting, and the expected output is time segments (e.g. “from 5 minutes into the meeting to 10 minutes into the meeting”) labeled with the agenda item being discussed during each segment. There are two possible sources for the actual agenda item labels that will be applied to the meeting segments. They can be provided as input to the algorithm for the test meeting (as provided by the meeting participants of that particular meeting). They can also come from the labeled data acquired from meeting participants in previous meetings in the same meeting sequence.

The visible human actions from which supervision must be extracted are the speech uttered and the notes written during the meetings. We will harness the meeting participants’ note-taking actions in order to extract feedback for improving agenda item detection. Recall that according to our definition of “matching” actions, the system and the human actions do not match, and that in order to extract supervision from the human's notes, we will redesign the interface according to the recipe shown in Chapter 3.

Below we describe how we have used these three steps to design the SmartNotes interface to extract agenda item-labeled meeting segments from meeting participants' during-meeting note taking:

1. Identify the kind of labeled data that is needed to improve the system: In order to improve the system, we aim to collect meeting segments labeled with the agenda item being discussed during that segment. The agenda item label should be a text string that serves as a caption for the topics being discussed in that segment.

2. Identify a relationship between a user action and the function that needs to be learned: Recall from Chapter 3, the relationship between the notes taken by meeting participants and the agenda items discussed during the meeting rests on two facts:

   (a) Most lines of notes refer to discussions that occur during a particular segment of the meeting, and the notes can be said to belong to the same agenda item as the discussions.

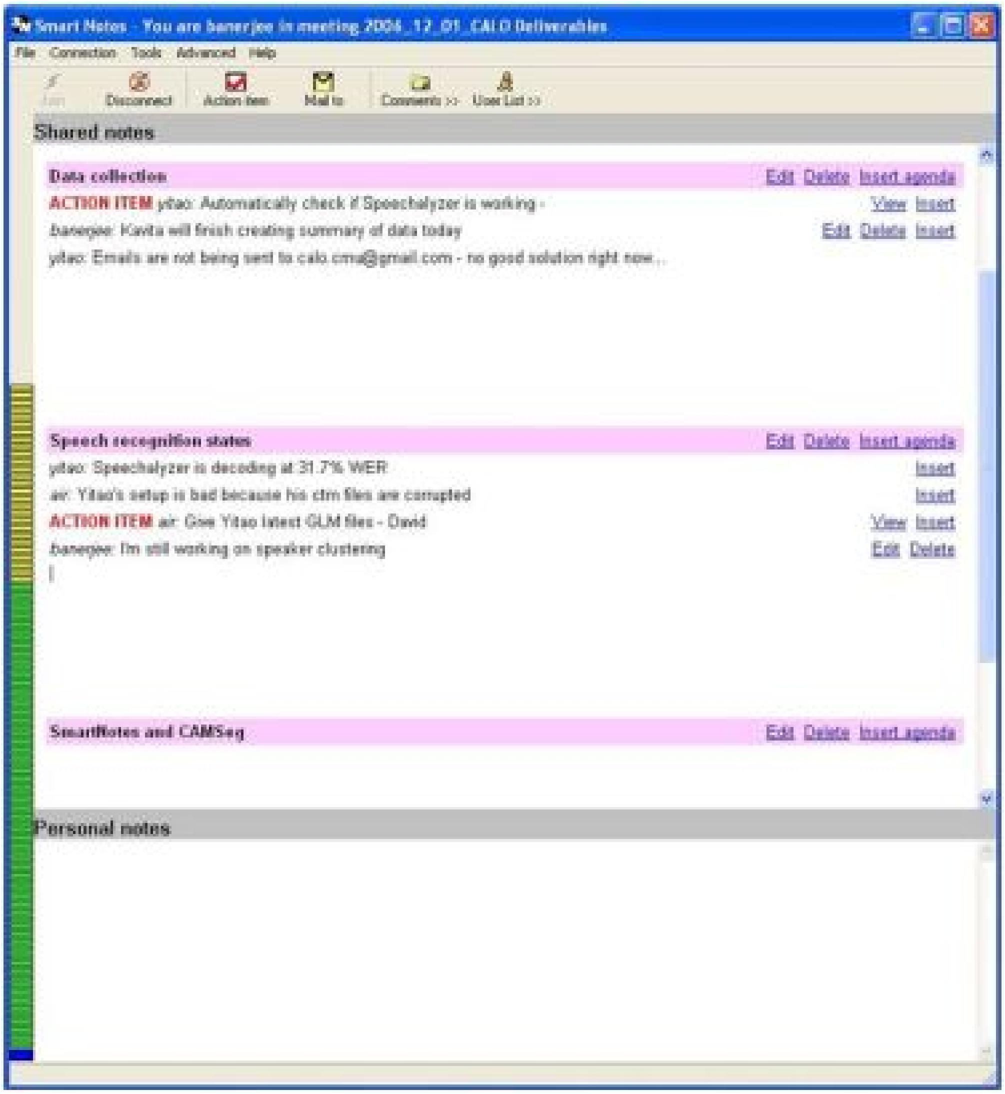
   (b) A line of note usually occurs shortly after the meeting segment that contains the discussion it refers to, and there are large textual overlaps between the notes and the words spoken in during the discussion they refer to.

   Using these two facts, we can construct a labeled data acquisition mechanism whose goal is to encourage meeting participants to label individual lines of notes with the agenda item they belong to. By automatically identifying the meeting segment that line of note refers to, we can then propagate the agenda item label from the note to the meeting segment, thus resulting in labeled data with which the agenda item labeling algorithm can be (re)trained.

3. Interface to take advantage of this relationship: We have developed the SmartNotes interface that takes advantage of the above relationship between notes and agenda item-labeled meeting segments. We describe this interface in the next section.

5.4.2 The SmartNotes System

The SmartNotes system consists of two parts: A desktop application, called the "SmartNotes Client" through which participants can record their audio and type notes, and a web based application, called the "SmartNotes Website" through which participants can access the audio and notes, and write summaries of previous meetings. We describe these two components in this section.



Data collection

ACTION ITEM ytiao: Automatically check if Speechalyzer is working -

banerjee: Kavita will finish creating summary of data today

ytiao: Emails are not being sent to calo.cmu@gmail.com - no good solution right now...

Speech recognition status

ytiao: Speechalyzer is decoding at 31.7% WER

air: Yitao's setup is bad because his ctm files are corrupted

ACTION ITEM air: Give Yitao latest GLM files - David

banerjee: I'm still working on speaker clustering

SmartNotes and CAMSeg

Figure 5.1: The SmartNotes interface. Screen shot taken during a (mock) meeting.

The SmartNotes Client is a desktop application that meeting participants can use to take notes as well as record their speech. Each meeting participant is expected to come to the meeting with a laptop running the client on it, and with a close-talking microphone connected to the laptop. Figure 5.1 shows a screenshot of this application being used in a meeting.

Synchronization, Authentication and Joining a Meeting:

Before the meeting commences, each participant first starts the SmartNotes client. Each client synchronizes itself to a single pre-specified NTP machine. This ensures synchronicity between event timestamps created by different clients. Next, the user logs into the client by authenticating himself. The client authenticates the user by sending his username and password to a central “Meeting Server”. The advantage of authenticating each user is that this allows us to trivially identify the speaker for each utterance, and the note taker for each line of notes written, instead of having to deduce these facts through sophisticated methods. Once authenticated, the server sends the client the names of the currently running meetings; the user has the option of either joining one of these meetings, or “creating” a new meeting. Typically the meeting leader logs in first and creates a new meeting, and then the remaining meeting participants join the meeting. By joining the same meeting, users can share their notes, as described below.

Recording Speech:

As soon as a participant joins a meeting, his audio starts getting recorded. The application has a VU-meter that continuously shows the user his volume level; meeting participants can use this VU-meter to adjust their microphone volume to an appropriate level. (This VU-meter is visible on the left margin of Figure 5.1. Audio is recorded at 16 kHz and uploaded to the Meeting Server opportunistically, based on network bandwidth availability. The transfer can continue beyond the end of the meeting, if sufficient network bandwidth is not available during the meeting. Additionally, the audio transfer is robust to network loss and power shutdowns.

Collaborative Note Taking:

The main function of the SmartNotes client is to allow meeting participants to take notes during the meeting. Once a user joins a meeting, he is shown the note taking interface, as shown in Figure 5.1. This interface consists of two main note taking areas: The “shared notes” area and the “private notes” area. Notes typed into the shared notes area are automatically prepended with the author’s name, and shared with all other meeting participants. This sharing is facilitated through the Meeting Server, and occurs in real time so that at all times, every meeting participant’s shared notes look exactly the same. To avoid the problem of multiple simultaneous edits, participants can only edit/delete their own notes. Notes typed into the private notes area are not shared with any other meeting participant.

Agenda Based Note-Taking:

The goal of this note taking interface is not only to acquire labeled data to improve agenda item detection, and to also help meeting participants organize their notes as they are taken. (By providing the users with some value, they are likely to use the interface as suggested). To serve both these goals, the interface provides a mechanism for participants to enter the agenda for the meeting. (Entering the agenda can be done at any time and by anyone, although typically it is done at the beginning of the meeting by the meeting leader). The interface splits the shared notes area (in each meeting participant's client) into as many text boxes as agenda items in the entered agenda, and labels each text box with the name of the corresponding agenda item. In Figure 5.1, the agenda items are "Data collection", "Speech recognition status" and "SmartNotes and CAMSeg". Meeting participants are expected to type notes on a particular agenda item in the box labeled with that agenda item's name.

The immediate benefit to the meeting participants of grouping their notes by agenda item is that since the notes are shared live with all participants in the meeting, it helps participants ground their understanding of what the agenda items for that meeting are and which agenda item is being discussed at any given point of time. (The notes themselves help participants ground their understanding of what is being said in the meeting). Additionally, grouping notes by agenda item helps meeting participants quickly retrieve all notes on a single agenda item (particularly ones that recur across multiple meetings) from the SmartNotes website.

As meeting participants take notes in an agenda item box, they are indicating that the note belongs to that agenda item. Additionally, the SmartNotes system records the time at which the note was written. These two pieces of information together allows the system to extract meeting segments labeled with the agenda item that was being discussed during those segments as described above.

Action Item Identifier

In addition to taking notes, the meeting participants can also annotate certain notes as being "action items" – commitments by one or more participants to perform certain tasks within a specific time frame. When a user clicks on the "Action Item" button, he is provided with a form in which he can enter the details of the action item, such as what the action is, who is responsible for it, when it is due by, etc. The advantage to the group is that all the action items are separately available for future access, and can also be automatically emailed to participants as reminders before the next week's meeting.

The SmartNotes Website

We showed in the previous chapter that meeting participants sometimes need to retrieve information from past meetings. To address these needs, the SmartNotes website allows meeting participants to access the speech and the notes taken in past meetings using the SmartNotes client. The main goal of the website is to help users quickly access information from previous meetings. It is hoped that by making information access easier, users may be enticed into using the SmartNotes client's interface in a way that gives the system better data. For example, once the users find the feature by which they can access notes from a single agenda item over multiple meetings useful, they may feel more encouraged to take notes within agenda item boxes in the SmartNotes client. Additionally the website provides more opportunities for acquiring data with which to improve other system goals besides topic segmentation; discussion of these goals is beyond the scope of this paper. However, for completeness, we shall briefly describe the SmartNotes website's interface.

Meeting participants log in to the website using the same username/password combination they use to log in to the SmartNotes client. Once logged in, the system shows the user a list of all his meetings. The user can click on any meeting to bring up all the shared notes taken during that meeting. Additionally, he can access the speech from each participant in that meeting, using the notes as an index. That is, the user can access n minutes of speech before and m minutes of speech after each note, from each speaker in the meeting (or he can listen to the combined audio channels). Finally, the user can create or access previously created summaries of the meetings.

5.4.3 Obtaining Agenda Item-Labeled Meeting Segments from SmartNotes

Given the time-stamped and agenda item-labeled notes in the SmartNotes system, we extract agenda item-labeled meeting segments through the following algorithm. For each line of note typed by the user, we have the following pieces of information:

1. The agenda item box it was typed into.
2. The time at which it was typed.
3. The meeting participant who typed it.
4. The text in the note.

Our goal is to use these pieces of information to automatically detect which meeting segment this note refers to, and associate the agenda item label of this note with that meeting segment. For simplification however, we only use the time stamp of the note. Table

Note #	Time stamp (seconds from start of meeting)	Agenda item label on the note
1	100	Agenda item 1
2	150	Agenda item 1
3	170	Agenda item 2
4	230	Agenda item 2
5	250	Agenda item 1
6	290	Agenda item 3
7	350	Agenda item 3

Table 5.1: Hypothetical Time Stamps and Agenda Labels on Notes.

Segment #	Agenda item label	Relative start time	Relative end time
1	Agenda item 1	0	160
2	Agenda item 2	160	240
3	Agenda item 1	240	270
4	Agenda item 3	270	370

Table 5.2: Extracted Labeled Segments from Hypothetical Meeting.

5.1 shows these two pieces of information for each line of note in a hypothetical meeting that is 370 seconds long.

To obtain agenda item-labeled meeting segments, we first order the notes according to their time stamps. Next, for every pair of chronologically consecutive notes that were typed into different agenda item boxes, we hypothesize a boundary midway between the time stamps of those two notes. Thus, in the hypothetical example in Table 5.1, the algorithm does not hypothesize a boundary between notes 1 and 2, but does hypothesize one halfway between notes 2 and 3, that is, at time point 160 seconds from the start of the meeting. Similarly, there would be no boundary hypothesized between notes 3 and 4, but there would be one halfway between notes 4 and 5 at time point 240 seconds, and again one halfway between notes 5 and 6, and so on. Thus, for this hypothetical example, the labeled segments would be as shown in Table 5.2. We call this segmentation the *notes based segmentation*, since it is completely based on the notes taken by the meeting participants, and does not use the speech at all.

The method described above is only one way of determining which segments to transfer the agenda item labels to. Another reasonable method may be to split the distance between two consecutive notes from different agenda items closer towards the earlier note, instead of halfway as is done above. More effectively, the location of the split could be determined based on the text in the notes and the speech.

Because of the simplicity of the algorithm of determining which meeting segments to transfer labels to, the algorithm's accuracy completely depends on the note taking behavior of the participants in the meeting. For example, if the participants discuss an agenda item, but do not type even a single note on that item, then this algorithm will not output a segment corresponding to that agenda item at all. Similarly, if a participant types notes into one agenda item box while the discussion is focusing on a different agenda item, then again this segmentation would be incorrect. Finally, the quality of these labeled segments will depend on the lengths of the gaps between consecutive notes that are in different agenda item boxes. The larger this gap is on average, the harder it is to tell where exactly the "real" boundary is. The advantage of this mechanism of course is that since the text in the speech is not required, the algorithm is not affected by the transcription accuracy of the uttered speech in the meeting.

5.5 Informal Usability Evaluation of SmartNotes

We evaluate both the SmartNotes system, as well as the labeled data we acquire through it. We evaluate the SmartNotes interface by asking how often meeting participants use the system in the "desired way" from the point of view of extracting labeled data. That is, how often do meeting participants enter their agenda into the meeting, and how often do they take their notes in the agenda item boxes as prescribed.

We have deployed the SmartNotes system among different teams that meet on a regular weekly basis. The deployment was primarily done from January 2006 through October 2008. During this period 75 meetings were held where participants used SmartNotes to record their audio, and take notes. 16 unique participants have taken part in 4 meeting sequences of at least 3 meetings each. An agenda was specified and at least one line of note has been taken in each of these 75 meetings.

For more detailed analysis we looked at a subset of 10 meetings from April to June 2006. On average, these meetings are 31 minutes long, have 4.1 agenda items, and 3.75 participants (ranging from 2 to 5). Each agenda item has on average 5.9 lines of notes in them, for a total of about 25 lines of notes per meeting. These notes include all the notes taken by all the participants in the meeting in the shared note taking area.

These numbers show that the users the system was deployed with made consistent and frequent use of SmartNotes. More importantly they used the system in the manner that results in labeled data – that is, they entered their agenda item at the beginning of the meeting, and then took notes within the agenda item boxes. This shows that users find enough value in the system for them to use it on a regular basis.

5.6 Evaluation of the Extracted Labeled Data

In order to evaluate the extracted labeled data, we will compare it to the manually labeled data. We will also compare it to two unsupervised segmentation baselines. Finally we will combine the labeled data obtained above with an unsupervised algorithm to create a hybrid algorithm. We first describe evaluation metric, then the data for experimentation, and finally the unsupervised algorithm and the hybrid algorithm.

5.6.1 Evaluation Metric

We evaluate meeting segmentation using the Pk metric [7]. This metric computes the probability that two randomly drawn points a fixed time interval k seconds apart from the meeting are incorrectly segmented by the hypothesized segmentation, as compared to a reference segmentation, where a “correct” segmentation requires that both the reference and the hypothesis put the two points in a single segment, or both put them in different segments. Note that if the hypothesis and the reference put the two points in different segments, the hypothesis is deemed to be correct for those two points, even if the hypothesis and the reference predict different numbers of boundaries between the two points. Following [7], we compute the value of k for a given meeting to be half the average size of the segments in the reference segmentation for that meeting. Observe that Pk is a measure of error, and hence lower values are better. Specifically, Pk ranges between 0 (the reference and the hypothesis segmentations agree on every pair of points k seconds apart) and 1 (they disagree on every point).

5.6.2 Data and Annotations Used for Evaluation

For the evaluation reported here, we used the first 10 meetings from this sequence. On average, each meeting is 31 minutes long, has 4.1 agenda items, and has 3.75 participants (ranging from 2 to 5). Each agenda item has on average 5.9 lines of notes in them, for a total of about 25 lines of notes per meeting. Note of course that these notes include all the notes taken by all the participants in the meeting in the shared note taking area.

Each meeting was manually segmented by two independent annotators. These annotators were provided with the agenda of the meeting (but not with the notes) and were asked to split the meeting into segments such that each segment corresponded to one of the agenda items in that list. To compute the degree of agreement between their annotations, we follow [19] and simply compute the Pk between the two annotations. Since the choice of the reference affects the value of k (the distance between the two probe points in the calculation of Pk), we chose to perform this computation twice, each time using a different person’s annotation as the reference. The resulting average Pk value over the 10 meetings was **0.06** (standard deviation: 0.049), regardless of the choice of reference annotation. These agreement numbers are substantially better than those reported in [19]. Several factors may have contributed to these high levels of agreement. First the concept of an agenda item is better defined than that of a “topic” as defined in that paper. Second, the meeting participants in our corpus typically displayed a somewhat disciplined adherence to the agenda at hand, unlike many of the meetings annotated in [19]. Finally, and perhaps most crucially, our annotators had access to the list of agenda items in the meeting they were annotating, whereas the annotators in that paper were asked to identify the topics in addition to segmenting the meetings into the topics, which can introduce further variability between annotators.

5.6.3 Unsupervised Segmentation Baseline

In order to situate the evaluation of the segmentation obtained from meeting participants’ notes through the SmartNotes system, we establish a simple unsupervised baseline by implementing the popular TextTiling algorithm described in [21]. Here we give a brief overview of our implementation of this algorithm. The algorithm’s input is all the audio recorded during a meeting, along with the meeting’s absolute start and end times. The algorithm’s output is a set of time points within the meeting’s start and end times that the algorithm considers to be times at which the meeting participants finished discussing an agenda item, and started discussing another one. Note that in our implementation of this algorithm, we cannot output labels for the segments.

The algorithm proceeds by considering each time point t seconds from the start of the meeting, where t takes values 0, 1, 2, etc till the end of the meeting. For each such time point t, two windows are considered, one starting at time $t - k$ and ending at t and another starting at t and ending at time $t + k$. For each of these two k-seconds long windows, it constructs a vector containing the frequencies of the words uttered during the window by all the meeting participants, as output by a speech recognizer or as manually transcribed by a human, depending upon the experimental setup. Closed class words such as the articles and prepositions are ignored. Next the cosine distance between these two vectors is computed, according to the formula in Equation 5.1.

$$\cos(v_1, v_2) = \frac{\sum_{i=1}^{n} w_{i,v_1} w_{i,v_2}}{\sqrt{\sum_{i=1}^{n} w_{i,v_1}^2 \sum_{i=1}^{n} w_{i,v_2}^2}} \tag{5.1}$$

Here, v_1 and v_2 represent the two vectors whose similarity is being computed, w_{i,v_1} represents the frequency of the i^{th} word in vector v_1 (and similarly, w_{i,v_2} the frequency of the i^{th} word in vector v_2), and n is the size of the vectors. Care is taken to ensure that each dimension in the two vectors represents the frequency of the same word in the two windows. Words that occur in one window but not in the other are considered to have a frequency of 0 in the other window. For time points near the beginning and the end of the meeting, we use smaller values of k to ensure that the windows to the left and right of any given time point have the same size.

The computed value quantifies how "similar" the word frequencies in the two windows are. Intuitively, the more dissimilar they are, the more likely it is that those two windows contain discussions on different agenda items, and consequently that the time point between those two windows is a topic boundary. After calculating the cosine similarity values, a depth score is computed for all the time points being considered in the meeting, as described in [21]. Roughly speaking, depth scores are non-zero only for the bottoms of valleys and are higher if the valleys have "tall walls". Given these depth scores, a threshold is computed from them (the mean of the depth scores, plus half their standard deviation), and all time points with depth scores more than the threshold are reported as agenda item boundaries.

Observe that this algorithm is almost completely unsupervised; the only parameters that can be tuned are the size of the window (the value of k in the description above), and the threshold above which boundaries are reported. We set the value of k to 350 seconds as this value performed well on separate held out data. As mentioned above, we follow [21] in computing the threshold, with the only difference being that we add half the standard deviation instead of subtracting it; adding performed better on our data.

The algorithm has no notion of the contents of different topics, and hence cannot be used to identify the topics for labeling purposes, for example. This fact is both its strength and weakness: While it does not need to be pre-trained on the specific topics in the meeting it needs to segment, its accuracy is low.

5.6.4 Hybrid Algorithm 1: "Improved Baseline"

Although we derived a segmentation directly from the notes typed by the user above (and called it "notes based segmentation"), its accuracy may vary a lot depending on the note taking behavior of the specific group. However, instead of simply outputting the notes based segmentation, another strategy could be to use the notes to *improve* the baseline algorithm that uses the speech as its input, and is thus more *robust*.

The baseline algorithm assumes that all words are equally important for topic segmentation (except for closed class words as mentioned above). This is a reasonable assumption, if nothing further is known in advance about the meeting. However, if some (perhaps short) segments of the meeting are *labeled* with the agenda item they belong to, we can use these segments to learn which uttered words are indicative of the individual topics, and which are not.

We can derive such labeled data from the notes by performing almost the same process as that for deriving segmentation from the notes, with the following difference. For every pair of consecutive notes that are in two different agenda boxes, instead of hypothesizing one boundary halfway between the two notes, we hypothesize *two* boundaries, one each at the time points of the two notes. Thus, for the hypothetical example in table 5.1, the output labeled segments would be:

- Segment 1: Agenda item 1. From 100 to 150
- Segment 2: Agenda item 2. From 170 to 230
- Segment 3: Agenda item 1. From 250 to 250
- Segment 4: Agenda item 3. From 290 to 350

Thus, this algorithm takes the meeting time *between* consecutive notes that are in the same agenda box, and labels that time with the name of the agenda item of that box. However, when consecutive notes are in different boxes, it is unclear when the discussion changed from one agenda item to the next, so it leaves the entire meeting time between those two notes unlabeled.

Given these labeled segments, for every agenda item, we create a bag containing all the words uttered in all the segments labeled with the name of that agenda item. This gives us one bag of words per agenda item. From these bags, we find those words that occur in only one bag, and that occur at least twice. These are the words that most differentiate agenda items from each other. We then run the baseline algorithm exactly as before, but only using these words in every window pair, and ignoring all other words. We can use other measures of association; however, we started with the simplest algorithm possible.

5.6.5 Hybrid Algorithm 2: "Improved and Constrained Baseline"

In the modified baseline algorithm described above, we first identify the most distinguishing words, and then run the same baseline algorithm as before. Although this improves the results compared to the original baseline (as shown in the Evaluation section), this algorithm still does not directly use any information from the notes, and thus often predicts boundaries that fall between notes that were typed into the same agenda item box. The simplest way to constrain these outputs is to simply only output boundaries that lie between consecutive notes that are in different agenda item boxes. We experiment with this simple constraint and report evaluation results below.

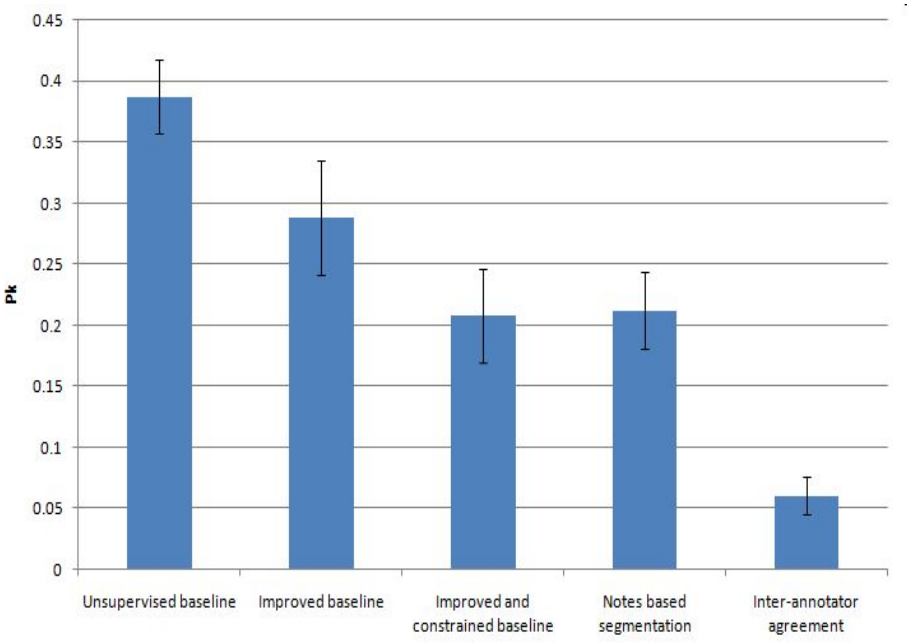


Figure 5.2: Performance for the four approaches (first four bars) and the inter-annotator agreement (fifth bar), averaged over the 10 meetings in the corpus. Error bars are drawn using Standard Error.

5.6.6 Results

Figure 5.2 summarizes the overall results for the four segmentation approaches over the 10 meeting corpus. Note that for the algorithms that take spoken words as input ("Unsupervised Baseline", "Improved Baseline" and "Improved and Constrained Baseline" in the figure), the speech was automatically recognized using the CMU Sphinx–3 speech recognizer; across all the participants over the entire set of 10 meetings, the Word Error Rate was 40%. That is, 40% of the words automatically recognized were incorrect, as compared to human transcription. Although we do have manual transcriptions for our meeting corpus, we focus on segmentation results obtained using automatically transcribed speech since manual transcriptions will in general not be available. (However, see the Discussion section for segmentation results using manual transcriptions).

The baseline algorithm achieves an average P_k of 0.387 (standard deviation: 0.096), ranging from 0.272 to 0.543. This implies that in approximately 38.7% cases, the algorithm mis-segments a randomly drawn pair of points from the meeting. On the other end of the spectrum, the purely notes based segmentation achieves an average P_k of 0.212 (standard deviation: 0.099), ranging from 0.085 to 0.382. This result represents a 45% improvement over the baseline algorithm, and is a significant improvement ($p < 0.01$, using the Wilcoxon matched–pairs signed–ranks test).

Using the notes based segments to learn the utterance words that are most strongly correlated with the segments results in the “improved baseline” - the 2^{nd} bar from the left in figure 5.2. This algorithm achieves a P_k of 0.288 (standard deviation: 0.148), ranging from 0.052 to 0.571. This result represents a 25% improvement over the unsupervised baseline, and the improvement is significant ($p < 0.01$ using the Wilcoxon test). The 3^{rd} bar from the left in figure 5.2 represents the same improved baseline as above, but where the outputs of the algorithm are constrained so that there are no boundaries between chronologically consecutive notes that are in the same agenda box. This algorithm achieves an average P_k of 0.208 (standard deviation: 0.121), ranging from 0.060 to 0.423. This result represents a 46% improvement over the baseline, and is a significant improvement ($p < 0.01$, using the Wilcoxon test). Although this P_k value is slightly lower than that achieved by the notes based segmentation, the difference is not statistically significant.

Observe that all these results are statistically significantly higher than the inter-annotator agreement of 0.06. This shows that this automatic labeled data extraction algorithm is not quite as accurate as the accuracy ceiling defined by the inter-annotator agreement, and there is still room for improvement by perhaps using a more sophisticated extraction algorithm that takes the words into account.

5.6.7 Discussion

Benchmarking Against [38]

To benchmark our meeting segmentation results against an independent state–of–the–art algorithm, we invited the authors of [38] to run their algorithm on our 10–meeting corpus. This algorithm uses an unsupervised generative topic modeling technique to segment multi–party discourses into topic segments. The algorithm was run on our corpus and, for each utterance in each meeting, it produced the probability that there was a change in agenda item at the end of that utterance. For each meeting, a threshold value was manually set, and all utterances with boundary probabilities greater than this threshold were marked as boundaries. A different threshold was set for each meeting to ensure that the algorithm produced exactly as many segments as the human annotator produced for that meeting; in a production system this threshold value can potentially be computed automatically based on the number of notes–based segments. This algorithm achieved an average P_k of 0.257 over the 10 meetings when run using the words output by the speech recognizer, and a P_k of 0.277 when using the manually transcribed speech. Although both the purely notes based segmentation and the improved and constrained baseline outperform this segmentation (by 21% and 24% relative respectively), these differences are not statistically significant. Thus, we conclude that the notes based segmentation performs as well as a state–of–the–art segmentation algorithm.

The Quality of the Notes Based Segmentation

Thse results show that the extracted labeled data that forms the basis of the purely notes based segmentation performs well – significantly better than the unsupervised baseline, and as well as a state-of-the-art algorithm. One obvious drawback of this algorithm is that it is heavily dependent on the users' note taking behavior. For example, one would expect that if users took fewer notes than they did in our corpus, the notes based segmentation would suffer. To test the effect of a reduced number of notes on the performance of the notes based segmentation, we performed the following experiment. For each meeting, we randomly deleted X% of the notes (X varying from 0 to 100 with a step size of 10), performed the purely notes based segmentation using the remaining notes, and computed the P_k value. For each value of X, we repeated this experiment 1000 times, each time deleting a different random set of notes, and computed the average P_k over all meetings over all iterations.

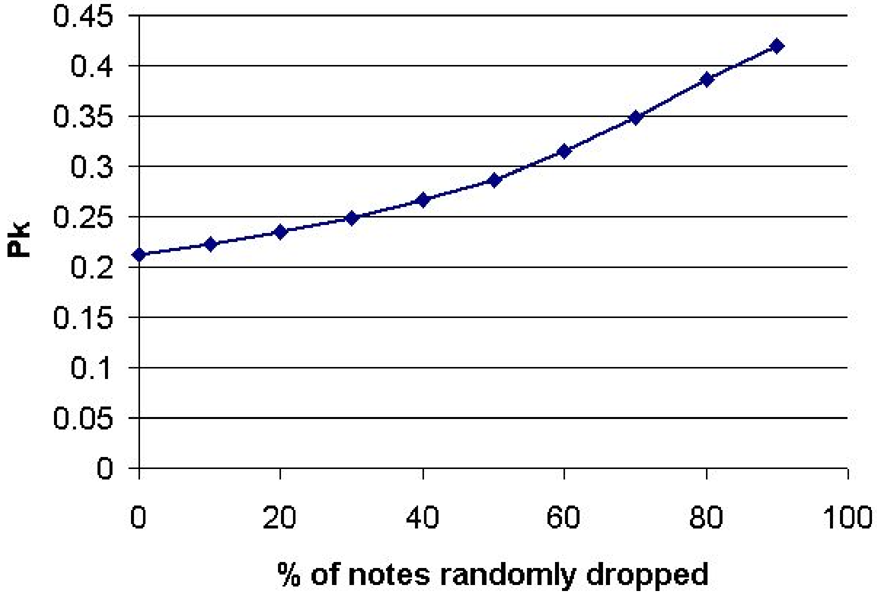


Figure 5.3: Performance of Notes Based Segmentation after Deleting a Random X% of the Notes.

Figure 5.3 shows a plot of the average P_k against the percentage of notes dropped. Observe that at $X = 0$, no notes are dropped, and the P_k is 0.212 as reported in the Evaluation section. At $X = 100$, all the notes are dropped, and no segments are created. The P_k value at this point is meaningless. At $X = 30\%$, the P_k value (0.249) is still better than that obtained by the state–of–the–art algorithm (0.257). Thus even with 30% less notes than that taken by the participants in our corpus, the notes based segmentation performs as well as the state–of–the–art algorithm. Further, even at $X = 70\%$, the P_k value (0.348) is better than that produced by the unsupervised algorithm (0.387). This implies that even if we have only 30% of the notes taken by the participants in our corpus, we can outperform the baseline unsupervised algorithm. These numbers are not surprising since the notes based segmentation algorithm only depends on how close the last note of one agenda item and the first note of the next agenda item are to the actual boundary between the two agenda items.

Other ways in which the quality of the notes based segmentation can be compromised include situations where users type notes on one agenda item while they are discussing another agenda item (either erroneously, or on purpose). We did not experiment with this situation, but in general plan to test the notes based segmentation algorithm on different groups of people to assess its efficacy with varying human note–taking habits.

The Quality of the Other Algorithms

The “Improved baseline” that uses the notes based segmentation to learn the words that are correlated with the segments did not perform as well as we had hoped. This is perhaps because of a data sparcity problem, which can potentially be alleviated by accumulating the learning over multiple meetings which we shall attempt in the future. Using the notes based segmentation to constrain the “learned” segmentation performed as well as the notes based segmentation itself, but not better as we would have hoped. Perhaps a better approach would be to also look for “cue phrases” (topic independent phrases that signal topic boundaries, such as “moving on...”), as has been done by [7, 15].

Using Manual Speech Transcriptions

As mentioned earlier, our focus is on evaluating our various agenda segmentation algorithms using automatically recognized speech. However, to see the effect of the errors in such automatic transcripts on segmentation performance, we also ran all our experiments on manually transcribed speech. For the “Unsupervised Baseline” algorithm, the P_k improved from 0.387 to 0.339 by using the manual transcriptions, and this improvement was moderately significant ($p < 0.05$ using the Wilcoxon test). For the “Improved Baseline” algorithm, the difference was not statistically significant. For the “Improved and Constrained Baseline” algorithm, the P_k value improved from 0.208 to 0.152 by using manual transcripts, and this improvement was statistically significant ($p < 0.01$ using the Wilcoxon test). Finally, the algorithm described in [38] did not experience a statistically significant improvement in segmentation performance by using the manual transcripts. It is unclear why the "Improved and Constrained Baseline" algorithm benefited so much from the manual speech transcripts whereas the "Improved Baseline" algorithm did not.

5.7 Summary

In this chapter we have explored an approach to extracting supervision from human actions (note-taking) when those actions do not match the system's actions (segmenting meetings into agenda items). Our approach has revolved around creating a specially designed note-taking interface called SmartNotes that encourages participants to write their notes within appropriate agenda item boxes and thus generate labeled data for this task.

We have deployed this system to real participants participating in real meetings, and have shown that the generated data is significantly better than a simple unsupervised baseline algorithm, and is of approximately the same quality as a state-of-the-art algorithm. While the data is not as accurate as manually annotated data, it is competitive with data obtained through other non-manual-labor-intensive approaches. This high quality data was obtained by simply giving meeting participants the SmartNotes interface and informing them about the advantages (to *them*) of taking notes in the interface. Observe too that the labeled data was obtained without performing any speech recognition or feature extraction, or by applying any "sophisticated" algorithm in the classic sense.

These results imply that high quality labeled data can be obtained through implicit supervision acquision, even when the system and human tasks are ill-matched.

Chapter 6

Noteworthy Utterance Detection using Implicit Supervision

6.1 System Goal and Motivation

The second system task is to automatically assist meeting participants to take notes during meetings by identifying and highlighting important information from the speech. While the previous task – automatically identifying the agenda item being discussed during a meeting - was aimed at making it easier to navigate through the recorded audio of past meetings, the goal of this second task in contrast is to reduce the need to navigate through the recording at all. Since it is faster to access information from written notes than recorded audio, our goal in this task is to help the user create high-quality information-rich notes during the meeting.

The importance of notes for fulfilling information needs is made clear from two facts shown in Chapter 4:

1. The availability of notes has a big impact on whether an information need gets met.

2. Notes are often not available.

Thus it is likely that a system that enhances the quality and quantity of notes taken at a meeting can help meeting participants more easily find the information they seek from previous meetings. Our approach to improving the information-content of the notes is to automatically identify noteworthy utterances during the course of the meeting, and suggest them to the users for inclusion in their notes. If the noteworthy-utterance detection component is accurate, the user will be able to include more information in his notes with less effort than if he did not have the assistance of the system, thus making it more likely that the notes will contain the information he seeks in the future.

Note that one approach to retrieving information from recorded meetings without resorting to sift through the audio is to first transcribe the speech using a speech recognizer, and then perform Information Retrieval type free-text searches through the transcribed text. The advantage of an accurate noteworthy utterances detector is brevity: only a small percentage of the utterances in a meeting are truly noteworthy and contain information that is likely to be needed in the future. If these utterances can be automatically identified, then the resulting summary will be far more concise than the whole transcript, thus requiring less time for the human to scan through the information.

6.2 Research Goals

Our main research goal is to explore the technique of implicit supervision through the task of noteworthy-utterance detection. This can be broken down into the following specific goals:

1. Evaluate a standard algorithm to perform extractive meeting summarization. This is the component that needs to be improved by acquiring implicit supervision from humans.
2. Develop an algorithm to extract implicit supervision in the form of labeled data from the notes taken in previous meetings.
3. Evaluate the extracted labeled data against manually labeled data.
4. Train a noteworthy utterance detector from the automatically extracted labeled data, and evaluate the detector against manually annotated data.
5. Through a note-taking user study evaluate the usefulness of notes-assistance system that suggests notes based on the noteworthy utterance detector trained on automatically extracted labeled data.

We discuss these tasks in more detail in the following sections:

- In Section 6.3, we address the question of whether it is possible for a human to accurately identify information that others would also find noteworthy.
- In Section 6.4, we evaluate a baseline noteworthiness detection algorithm.
- In Section 6.5, we present a modified version of this algorithm that uses more than 2 levels of noteworthiness.

- In Section 6.6, we present an algorithm to automatically extract labeled data from meeting participants' notes, and train a noteworthiness detector on this data.

- Finally, in Chapter 7, we present a user study to evaluate note-suggestions based on the automatically trained noteworthiness detector.

6.3 WoZ Study: Can a Human Do This Task?

6.3.1 Introduction

Judging which pieces of information in a meeting are noteworthy is a very subjective task. The subjectivity of this task is likely to be more acute than even that of meeting summarization, where low inter-annotator agreement is typical e.g. [16, 29, 36] etc. - whether a piece of information should be included in a participant's notes depends not only on its importance, but also on factors such as the participant's need to remember, his perceived likelihood of forgetting, etc. To investigate whether it is feasible even for a human to predict what someone else might find noteworthy in a meeting, we conducted a Wizard of Oz-based user study where a human suggested notes (with restriction) to meeting participants during the meeting.

Specifically, our goal is to establish whether it is possible for a human to identify noteworthy utterances in a meeting such that

1. For at least some fraction of the suggestions, one or more meeting participants agree that the suggested notes should indeed be included in their notes, and

2. The fraction of suggested notes that meeting participants find noteworthy is high enough that, over a sequence of meetings, the meeting participants do not learn to simply ignore the suggestions.

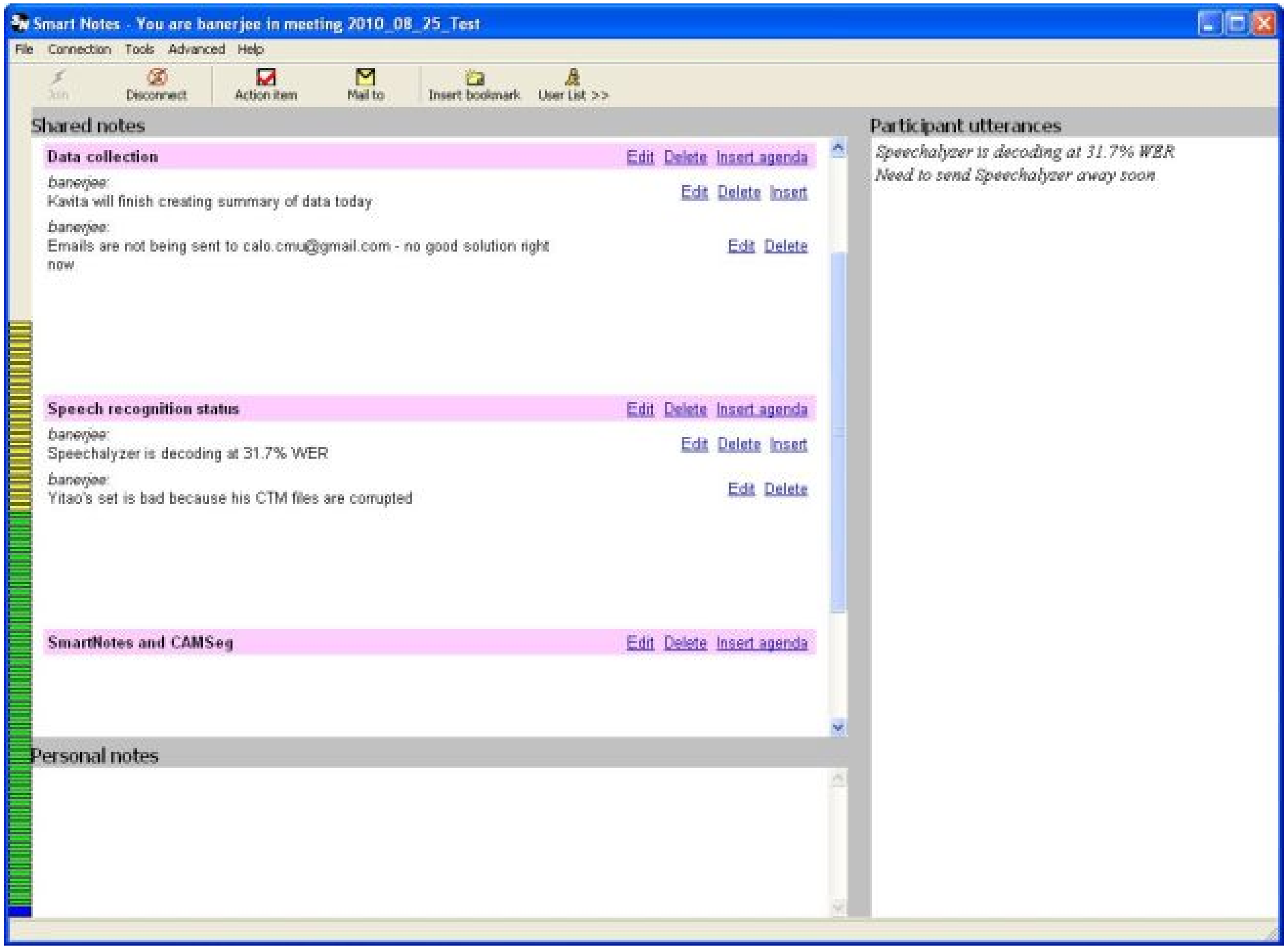
Observe that this task of identifying noteworthy utterances is more restricted than that of generic note-taking. While a human who is allowed to summarize discussions and produce to-the-point notes is likely to be useful, our goal is to establish the usefulness of an *extractive* summarization system that simply identifies noteworthy utterances and suggests their contents to the participants without further abstraction, condensation, paraphrasing, etc. Towards this goal, we conducted a Wizard of Oz-based pilot user study, as described in this section.

6.3.2 Study Design

We designed a user study in which a human Wizard listened to the utterances being spoken during the meeting, identified noteworthy utterances, and suggested their contents to one or more participants for inclusion in their notes. In order to minimize differences between the Wizard and the system (except for the Wizard's human-level ability to judge noteworthiness), we restricted the Wizard in the following ways:

1. The Wizard was allowed to only suggest the contents of individual utterances to the participants, and not summarize the contents of multiple utterances.
2. The Wizard was allowed to listen to the meeting speech, but when suggesting the contents of an utterance to the participants, he was restricted to using a real-time automatic transcription of the utterance. (He was allowed to withhold suggestions if they were too erroneously transcribed.)
3. In order to be closer to a system that has little or no "understanding" of the meetings, we chose a human (to play the role of the Wizard) who had not participated in the meetings before, and thus had little prior knowledge of the meetings' contents.

6.3.3 Notes Suggestion Interface



Shared notes

Data collection

banerjee:
Kavita will finish creating summary of data today

banerjee:
Emails are not being sent to calo.cmu@gmail.com - no good solution right now

Speech recognition status

banerjee:
Speechalyzer is decoding at 31.7% WER

banerjee:
Yitao's set is bad because his CTM files are corrupted

SmartNotes and CAMSeg

Participant utterances

Speechalyzer is decoding at 31.7% WER
Need to send Speechalyzer away soon

Figure 6.1: The new design of the SmartNotes interface. Observe the pane on the right labeled “Participant utterances” through which Wizard suggestions were shown to meeting participants. This pane was not available in the first version of the SmartNotes interface shown in 5.1.

In order to suggest notes to meeting participants during a meeting - either automatically or through a Wizard - we have modified the SmartNotes system, whose meeting recording and note-taking features have been described earlier in Section 5.4.2. Briefly, each meeting participant comes to the meeting with a laptop running the SmartNotes system. At the beginning of the meeting, each participant's SmartNotes client connects to a server, authenticates the participant and starts recording and transmitting his speech to the server. In addition, SmartNotes also provides meeting participants with a note-taking interface that is split into two major panes. In the "Shared notes" pane the participant types his notes that are then recorded for research purposes. In the new "Participant utterances" pane (Figure 6.1), Wizard-suggested notes are displayed. If at any time during the meeting a participant double-clicks on one of the suggested notes in the "Participant utterances" pane, its text gets included in his notes in the "notes" pane. The Wizard uses a different application to select real-time utterance transcriptions, and insert them into each participant's "suggestions" pane. (While we also experimented with having the Wizard target his suggestions at individual participants, we do not report on those experiments here; those results were similar to the ones presented below.)

6.3.4 Study Results

We conducted the Wizard of Oz study on 9 meetings that all belonged to the same sequence. That is, these meetings featured a largely overlapping group of participants who met weekly to discuss progress on a single project. The same person played the role of the Wizard in each of these 9 meetings. The meetings were on average 33 minutes long, and there were 3 to 4 participants in each meeting. Although we have not evaluated the accuracy of the speech recognizer on these particular meetings, the typical average word error rate for these speakers is around 40% – i.e., 4 out of 10 words are incorrectly transcribed.

On average, the Wizard suggested the contents of 7 utterances to the meeting participants, for a total of 63 suggestions across the 9 meetings. Of these 63 suggestions, 22 (34.9%) were accepted by the participants and included in their notes. Thus on average, about 2.5 Wizard-suggested notes were accepted and included in participants' notes in each meeting. On average, meeting participants took a total of 5.9 lines of notes per meeting; thus, 41.5% of the notes in each meeting were Wizard-suggested.

It cannot be ascertained if the meeting participants would have written the suggested notes on their own if they weren't suggested to them. However the fact that some Wizard-suggested notes were accepted implies that the participants saw value in including those suggestions in their notes. Further, there was no drop-off in the fraction of meeting notes that was Wizard-suggested: the per-meeting average percentage of notes that was Wizard-suggested was around 41% for both the first 4 meetings, as well as the last 5. This implies that despite a seemingly low acceptance rate (35%), participants did not "give up" on the suggestions, but continued to make use of them over the course of the 9-meeting meeting sequence.

6.3.5 Conclusion

From the fact that participants accepted nearly 35% of the notes suggested to them, and used these suggestions to construct about 41.5% of the notes, implies that it is feasible for an extraction based system to be useful to meeting participants in taking notes, provided the detection of noteworthy utterances is "accurate enough".

6.4 Noteworthiness Detection Baseline

6.4.1 Introduction

We now turn our attention to the task of developing and evaluating an algorithm for detecting noteworthy utterances. Noteworthiness of utterances is highly subjective in nature, varying widely by domain, meeting, participant and even perhaps time for a single participant. Thus it is difficult to strongly define what a noteworthy utterance is. Nevertheless, for a given meeting, we define an utterances as being *noteworthy* if one or more meeting participants are willing to include the information contained in that utterance in their notes. As mentioned earlier, the subjectivity of the note-taking task makes it a suitable task in which to attempt to automatically adapt to the note-taking habits of a particular group of participants over time.

Note-taking is both similar to and distinct from the problem of meeting summarization. Both aim to identify utterances that are in some ways more important than others. At the same time, whereas summaries aim to list the main points of discussion, notes are often taken to serve as an aid to memory. As such, even though some utterances may be important in some sense, they will not be included in the notes unless participants feel (a) they need to be remembered, and (b) they are likely to forget them. Nevertheless, the techniques developed in meeting summarization research are a good starting point for noteworthy utterance detection. In this section we present results of applying an *extractive summarization* approach to the problem of noteworthy utterance detection.

In typical extractive summarization research, human annotators manually identify individual utterances which, when extracted, would form a summary of the meeting. Such an approach is taken by [56] who extract a large number of features – lexical, structural, prosodic – and then train a binary classifier from the data. They experiment with various classifiers, but report best results with support vector machines and logistic regression. [33] also present a feature based approach, using Gaussian Mixture Models for each of the two classes. They compare this supervised approach to two unsupervised approaches – Maximum Marginal Relevance and an approach based on Latent Semantic Analysis, and find only small differences in accuracy. Finally [30] present a similar feature based extractive approach to broadcast news summarization. They compare Bayesian Network classifiers, Decision Trees and Support Vector Machines, and find Bayesian Network classifiers to be the most accurate.

6.4.2 Overall Approach

Our goal is to learn to identify noteworthy utterances – utterances whose content one or more meeting participants are willing to include in the notes – over a sequence of related meetings. Intuitively we hope to learn the idiosyncratic words and other features in a meeting sequence that are correlated with noteworthiness. Towards this end, we have recorded sequences of weekly project meetings where participants take notes. We then manually identify those utterances in the meeting that are most closely related to these notes, and label them as noteworthy. Thus every utterance in the meeting sequence is labeled as either noteworthy or not. We then extract lexical and structural features and learn a binary noteworthy/not-noteworthy classifier similar to a typical meeting summarization approach. We evaluate this classifier on held out meetings in the same sequence.

6.4.3 Data Annotation

Using the same set of data used for the agenda item detection results, described in section 5.6.2, we manually label each utterance with two different labels:

2-Way "Noteworthy" Labels

We used manual transcription of the speech in these meetings. Each line of note across all the participants in the meetings was manually aligned with the fewest possible utterances such that the text of those utterances together contain all the information in the note. More than 98% of all the lines of notes could be aligned to between 1 and 5 utterances. For example the note "End pointer giving grief due to move to 16 khz models" is aligned to the utterances "Was the end pointer giving grief?" and "Yeah, that was because the model changed from 11,025 to 16khz". The remaining 2% of lines of notes could not be aligned because they were high level summaries of discussions or were not actually spoken at the meetings. Note that because notes are shared live in SmartNotes, there were few lines of notes from different participants that referred to the same utterances. On average there were 22 lines of notes per meeting. Overall only 5% of all utterances were aligned with notes. This is a very low number compared to meeting summarization; in [33] the length of the summaries was 10% of the meeting. The alignment annotation was performed by one annotator only because the task is well defined, with little room for judgment calls.

2-Way "Show-worthy" Labels

Utterances were labeled as "noteworthy" or not using a very strict judgment: they were labeled as "noteworthy" only if the information contained in them was also included in the meeting participants' notes. For exploratory purposes, we also annotated all utterances that contained *any* information worth including in the notes as "show-worthy" utterances. This judgment was made very liberally, and only off-topic utterances (e.g. jokes) and utterances that are only related to the mechanics of the meeting (e.g. "Is this mic working?") were labeled as not show-worthy.

6.4.4 Features Extracted from Utterances

As mentioned previously, we take a supervised binary classification approach to both problems of identifying noteworthy utterances and show-worthy utterances. We experimented with both the Naïve Bayes classifier and with Decision Trees, but found superior performance with the latter; in this section we only report on results using the Decision Tree classifier. We extract the following features from utterances.

Our main sets of features are word n-grams. In past approaches to meeting/speech summarization, the actual words are typically abstracted away to improve generalizability. For example, [33] and [30] use the words to identify named entities, and then use the number of named entities as features. Since our goal is to adapt to the specifics of a particular meeting sequence, and learn correlations between the words and noteworthiness of utterances, we use the words themselves. We use all n-grams with n = 1 to 6 that occur at least 5 times or more across all the meetings in the sequence. We designate all other n-grams that occur less than 5 times as Out of Vocabulary (OOV) words, and use the number of OOV words in each utterance as a feature.

Our second set of features is based on term frequency – inverse document frequency (TFIDF). Following [33], for each word w, we define document frequency $df(w)$ as the number of utterances in which word w occurs, and term frequency $tf(w,i)$ as the number of times word w occurs in utterance i. $TFIDF(w,i)$ is then computed as:

$$TFIDF(w,i) = tf(w,i) * log(N/df(w)) \tag{6.1}$$

where N is the total number of utterances in the meeting sequences. For each utterance we use the following four TFIDF based features: the maximum and minimum TFIDF scores in that utterance, and the average and standard deviation of the TFIDF scores in that utterance.

In addition to these lexical features we used two types of structural features. The first is speaker information – who spoke the current utterance, who spoke how much in the preceding set of utterances and who spoke immediately after. Note that while these features are similar to the anchor/reporter features used for summarization of broadcast news in [30], the difference is that we use the actual identity of the speakers themselves (similar to using the words themselves, without abstraction). Besides speaker information we also use the length of the current utterance as a feature, as has been used in previous work.

6.4.5 Evaluation Metrics

Two sets of evaluation metrics have been used in speech summarization work in the past. In the first set, called ROUGE, the inputs are the system summary and one or more reference summaries. Using these inputs, n-gram-level precision, recall and F-measure are reported as the ROUGE score for the system. In the second set, the inputs are utterance-level "in summary / not in summary" classifications from the system and similar judgements from human judges. Using these annotations, utterance-level precision, recall and F-measure are reported. Utterance level precision is computed as the number of true positive utterances divided by the number of utterances labeled as noteworthy by the system, while recall divides the number of true positives by the total number of utterances manually marked as noteworthy. F-measure is then computed as a harmonic mean of precision and recall, typically with equal weight on both values. Utterance-level F-measure, precision and recall are considered a more stringent evaluation mechanism than ROUGE; we use utterance level F-measure as the metric of evaluation here, but will use ROUGE in later sections.

6.4.6 Noteworthy Utterance Detection Results

The following results were obtained by doing a leave-one-out cross-validation at the level of meetings. That is, for each meeting in the sequence, we trained the noteworthy and show-worthy classifiers on the manually labeled data from all the remaining meetings in the sequence, and then evaluated this classifier on the test meeting. We average the test results over all the meetings, and present results here. Note that we do not perform any tuning of the parameters because of the small amount of available data. Also, while a deployed system would only have access to previously held meetings, here we use future meetings as well in order to maximize the yield from our data.

Using all the lexical and structural features mentioned above, the noteworthiness classifier achieves an overall classification accuracy of **91%**. That is, 91% of the utterances are correctly labeled by the system as being noteworthy or not-noteworthy. Considering only the noteworthy utterances, we achieve an utterance level precision of **0.15**, recall of **0.12** and F-measure of **0.14**.

The high accuracy number of 91% is not surprising given the large skew in the data, and in fact is worse than the majority baseline that labels all utterances as non-noteworthy; such a baseline would get an accuracy of 95%, but would result in 0 precision and recall. For the precision/recall/fmeasure metric, another trivial baseline is to simply label all the utterances as noteworthy. Such a method would achieve a recall of 1.0, but a precision of 0.05, and an f-measure of 0.1. Thus, our algorithm triples the precision from this trivial baseline, and improves f-measure by 40% relative.

6.4.7 Showworthy Utterance Detection Results

We used the same set of features to train a show-worthy utterance classifier. We followed the same hold-one-out protocol as for noteworthy utterances, and achieved an overall accuracy of **81%** for both show-worthy and non-show-worthy utterances. This is a **28%** absolute improvement over the majority baseline of **53%**. Considering only show-worthy utterances, this algorithm achieves a precision of **0.81**, recall of **0.78** and F-measure of **0.80**. These results imply that the chosen features have good discriminative power for show-worthy utterances. In inspecting the decision trees learned, the topmost features are typically TFIDF features – tests on the maximum and average TFIDF scores of the utterance being classified.

We investigated whether show-worthy information about an utterance can help noteworthiness detection. There are two ways to use show-worthy information. The simple approach is to simply filter out all utterances in the test meeting labeled not show-worthy and then classify the remaining utterances as noteworthy or not. In order to see the extent to which show-worthy information can help, we used the manually labeled show-worthy information to do this filtering, and found that doing so improves precision of noteworthiness detection from **0.15 to 0.20**, while the recall remains fixed at 0.12 (f-measure increases to 0.15).

Another approach to improving noteworthy utterance detection using show-worthy information is to use the latter information as a *feature* in learning the noteworthiness classifier. This approach (again using manually labeled show-worthy information) also resulted in an f-measure of 0.15, but with no increase in precision and a modest increase in recall – from **0.12 to 0.14**. Thus show-worthy information can result in a slight improvement in the quality of noteworthiness detection.

6.4.8 Analysis of Results

Balancing Positive/Negative Examples

One of the problems of the data set as mentioned earlier is its skew – only 5% of the utterances are noteworthy. In order to see what effect this has on the accuracy of the

Ratio of noteworthy to non-noteworthy utts	**Precision**	**Recall**	**F-measure**
1 : 0.5	0.09	0.70	0.16
1 : 1	0.10	0.51	0.16
1 : 2	0.10	0.39	0.16
1 : 5	0.09	0.20	0.13
1 : 10	0.10	0.14	0.11
$\sim$ 1 : 20 (entire data)	0.15	0.12	0.14

Table 6.1: Balancing Positive/Negative Training Examples.

classifier, we re-sampled the training data as follows. For each training set, we retained all the noteworthy utterances, and randomly picked non-noteworthy utterances (without replacement) until we had a desired ratio of noteworthy to non-noteworthy utterances. We trained the classifier on this training data, and then evaluated the classifier on the test data. We did not change the ratio of utterances in the test data. We did this entire process 10 times for each chosen ratio, each time picking non-noteworthy utterances randomly, and computed the average f-measure across the 10 trials. We performed this experiment using 5 different ratios; the results of which are presented in Table 6.1. The results show that while recall increases greatly, the precision remains stable. This implies that as the fraction of non-noteworthy utterances is reduced in the training data, many more utterances in the test data previously labeled non-noteworthy by the system are labeled noteworthy. Additionally, utterances whose true labels are noteworthy and non-noteworthy are both labeled noteworthy by the system in equal proportions, resulting in the stable value of precision. These results suggest that the imbalance in the training data has a negative effect on noteworthiness detection accuracy, and re-balancing the data may help improving the recall of the classifier, although this hurts the precision values.

Other Analyses

In order to see the usefulness of each group of features, we performed training/testing using each feature group separately (and with the entire skewed training data). Interestingly, none of the TFIDF based features, nor the structural features – speaker and length information – are sufficient to learn a decision tree any better than a one-node stump that simply labels all utterances as "non-noteworthy". This indicates that on their own these features do not have the discriminating power to overcome the skew in the training data. The n-gram features on the other hand do not result in a degenerate tree. Unigrams alone (without any other feature) achieve an f-measure score of 0.06. Unigrams and bigrams together achieve an f-measure of 0.08. Trigrams and higher n-grams do not seem to improve this f-measure.

Thus individually, the features get a highest f-measure of 0.08, whereas when used together, they result in an f-measure of 0.14.

6.4.9 Summary

In this section we evaluated the effectiveness of a simple baseline extractive-summarization techniques when applied to the problem of noteworthy utterance detection. We showed that these algorithms achieve an f-measure of 0.15, and that recall improves greatly when the training data is well-balanced. We also introduced the concept of "show-worthy" utterances, and showed that such utterances can be accurately labeled.

We have also shown that one of the main challenges is the skew of the data: annotating utterances as noteworthy only if they were aligned to the notes taken in the meeting results in very small fractions of positively labeled data. To deal with this challenge, we introduce the concept of *multilevel noteworthiness* in the next section.

6.5 Multilevel Noteworthiness

6.5.1 Introduction

In the previous section, we used a binary "noteworthy" / "not noteworthy" annotation that is typical of work in the field of extractive summarization, e.g. [16, 29, 36]. We observed however that there are often many utterances that are "borderline" at best, and the decision to label them as "in summary" or "out", or "noteworthy" or "not ntoeworthy" is arbitrary. In this section, we explore a three-way annotation scheme to separate out utterances that are "clearly noteworthy" from those that are "clearly not noteworthy", and to label the rest as being between these two classes. Note of course that arbitrary choices must still be made between the edges of these three classes. However, having three levels preserves more information in the labels than having two, and it is always possible to re-create two labels from the three, as we do in later portions of this section.

6.5.2 Annotation

These multilevel noteworthiness annotations were done by two annotators. One of them – denoted as "annotator 1" – had attended each of the meetings, while the other – "annotator 2" – had not attended any of the meetings. Although annotator 2 was given a brief overview of the general contents of the meetings, his understanding of the meeting was expected to be lower than that of the other annotator. By using such an annotator, our aim was to identify utterances that were "obviously noteworthy" even to a human being who lacks a deep understanding of the context of the meetings. (In section 6.5.4 we describe how we merge the two sets of annotations.)

The annotators were asked to make a 3-level judgment about the relative noteworthiness of each utterance. That is, for each utterance, the annotators were asked to decide whether a note-suggestion system should "definitely show" the contents of the utterance to the meeting participants, or definitely not show (labeled as "don't show"). Utterances that did not quite belong to either category were asked to be labeled as "maybe show". Utterances labeled "definitely show" were thus at the highest level of noteworthiness, followed by those labeled "maybe show" and those labeled "don't show". Note that we did not ask the annotators to label utterances directly in terms of noteworthiness. Anecdotally, we have observed that asking people to label utterances with their noteworthiness leaves the task insufficiently defined because the purpose of the labels is unclear. On the other hand, asking users to identify utterances they would have included in their notes leads to annotators taking into account the difficulty of writing particular notes, which is also not desirable for this set of labels. Instead, we asked annotators to directly perform the task that the eventual notes-assistance system will perform.

In order to improve inter-annotator agreement in the annotations, the two annotators discussed their annotation strategies after annotating each of the first two meetings (but not after the later meetings). A few general annotation patterns emerged, as follows: Utterances labeled "definitely show" typically included:

- Progress on action items since the last week.
- Concrete plans of action for the next week.
- Announcements of deadlines.
- Announcements of bugs in software, etc.

In addition, utterances that contained the crux of any seemingly important discussion were labeled as "definitely show". On the other hand, utterances that contained no information worth including in the notes (by the annotators' judgment) were labeled as "don't show". Utterances that did contain some additional elaborations of the main point, but without which the main point could still be understood by future readers of the notes were typically labeled as "maybe show". Thus, this 3-level annotation scheme is a refinement of the "noteworthy" and "show-worthy" annotation schemes of the previous section, and indeed effectively borrow elements of both schemes.

Table 6.2 shows the distribution of the three labels across the full set of utterances in the dataset for both annotators. Both annotators labeled only a small percentage of utterances as "definitely show", a larger fraction as "maybe show" and most utterances as "don't show". Although the annotators were not asked to shoot for a certain distribution, observe that

Annotator #	Definitely show	Maybe show	Don't show
1	13.5%	24.4%	62.1%
2	14.9%	38.8%	46.3%

Table 6.2: Distribution of the three labels for each annotator. Annotator 1 was a participant in these meetings, annotator 2 was not.

they both labeled a similar fraction of utterances as "definitely show". On the other hand, annotator 2, who did not attend the meetings, labeled 50% more utterances as "maybe show" than annotator 1 who did attend the meetings. This difference is likely due to the fact that annotator 1 had a better understanding of the utterances in the meeting, and was more confident in labeling utterances as "don't show" than annotator 2 who, not having attended the meetings, was less sure of some utterances, and thus more inclined to label them as "maybe show".

6.5.3 Inter-Annotator Agreement

Using Kappa

To gauge the level of agreement between the two annotators, we compute the Kappa score. Given labels from different annotators on the same data, this metric quantifies the difference between the observed agreement between the labels and the expected agreement, with larger values denoting stronger agreement.

For the 3-way labeling task, the two annotators achieve a Kappa agreement score of **0.44** (± 0.04). This seemingly low number is typical of agreement scores obtained in meeting summarization. [29] reported Kappa agreement scores between 0.11 and 0.35 across 6 annotators while [36] with 3 annotators achieved Kappa of 0.38 and 0.37 on casual telephone conversations and lecture speech. [16] reported inter-annotator agreement of 0.32 on data similar to ours.

To further understand where the disagreements lie, we converted the 3-way labeled data into 2 different 2-way labeled datasets by merging two labels into one. First we evaluate the degree of agreement the annotators have in separating utterances labeled "definitely show" from the other two levels. We do so by re-labeling all utterances not labeled "definitely show" with the label "others". For the "definitely show" versus "others" labeling task, the annotators achieve an inter-annotator agreement of **0.46**. Similarly we compute the agreement in separating utterances labeled "do not show" from the two other labels – in this case the Kappa value is **0.58**. This implies that it is easier to agree on the separation between "do not show" and the other classes, than between "definitely show"

Metric	Definitely show	Maybe show	Don’t show
Precision	0.57	0.70	0.70
Recall	0.53	0.46	0.93
F-measure	0.53	0.54	0.80
Accuracy	69%		

Table 6.3: Inter-Annotator agreement for each of the three classes using utterance-level precision, recall, F-measure and accuracy.

and the other classes.

Using Accuracy, Precision, Reccall, and Fmeasure

Another way to gauge the agreement between the two sets of annotations is to compute utterance-level accuracy, precision, recall and f-measure between them. That is, we can designate one annotator’s labels as the “gold standard”, and use the other annotator’s labels to find, for each of the 3 labels, the number of utterances that are true positives, false positives, and false negatives. Using these numbers we can compute precision as the ratio of true positives to the sum of true and false posi-tives, recall as the ratio of true positives to the sum of true positives and false negatives, and f-measure as the harmonic mean of precision and recall. (Designating the other annotator’s labels as “gold standard” simply swaps the precision and recall values, and keeps f-measure the same). Accuracy is the number of utterances that have the same label from the two annotators, divided by the total number of utterances.

Table 6.3 shows the evaluation over the meeting dataset using annotator 1’s data as “gold standard”. The standard error for each cell is less than 0.08. Observe in Table 6.3 that while both the “definitely show” and “maybe show” classes have nearly equal f-measure, the precision and recall values for the “maybe show” class are much farther apart from each other than those for the “definitely show” class. This is due to the fact that while both annotators label a similar number of utterances as “definitely show”, they label very different numbers of utterances as “maybe show”. If the same accuracy, precision, recall and f-measure scores are computed for the “definitely show” vs. “others” split, the accuracy jumps to **87%**, possibly because of the small size of the “definitely show” category. The accuracy remains at **78%** for the “don’t show” vs. “others” split.

Using ROUGE Scores

Annotations can also be evaluated by computing the ROUGE metric [27]. ROUGE, a popular metric for summarization tasks, compares two summaries by computing precision, recall and F-measure over n-grams that overlap between them. Following previous work on meeting summarization (e.g. [54, 33]), we report evaluation using ROUGE-1 F-measure, where the value “1” implies that overlapping unigrams are used to compute the metric. Unlike previous research that had one summary from each annotator per meeting, our 3-level annotation allows us to have 2 different summaries: (a) the text of all the utterances labeled “definitely show” and, (b) the text of all the utterances labeled either “definitely show” or “maybe show”. On average (across both annotators over all the meetings) the “definitely show” utterance texts are 18.72% the size of the texts of all the utterances in the meetings, while the “definitely or maybe show” utterance texts are 61.6%. Thus, these two texts can be though of as representing two distinct points on the compression scale. The average R1 F-measure score is **0.62** over the 6 meetings when comparing the “definitely show” texts of the two annotators. This is twice the R1 score – **0.3** – of the trivial baseline of simply labeling every utterance as “definitely show”. The inter-annotator R1 F-measure for the “definitely or maybe show” texts is **0.79**, marginally higher than the trivial “all utterances” baseline of 0.71. In the next section, we compare the scores achieved by the automatic system against these inter-annotator and trivial baseline scores.

6.5.4 Learning to Detect Multilevel Noteworthiness: Approach and Features

So far we have presented the annotation of the meeting data, and various analyses thereof. In this section we present our approach for the automatic prediction of these labels. We apply a classification similar to that used in the previous section, Section 6.4. Instead of using decision trees however, we experiment with a Support Vector Machines-based classifier, with a linear kernel, similar to that used in [54]. Using the 3-level annotation described above, we train a 3-way classifier to label each utterance with one of the multilevel noteworthiness labels. In addition, we use the two 2-way merged-label annotations – “definitely show” vs. others and “don’t show” vs. others – to train two more 2-way classifiers. For each of these classification problems, we use the same set of features described below.

Instead of using manual transcripts, we transcribe utterances using the Sphinx speech recognizer, with speaker-independent acoustic models, and language models trained on publicly available meeting data. The word error rate was around 40%, which is typical of spontaneous meeting speech using speaker-independent models. (More details of the speech recognition process are in [22]).

For training purposes, we merged the annotations from the two annotators by choosing a “middle or lower ground” for all disagreements. Thus, if for an utterance the two labels are “definitely show” and “don’t show”, we set the merged label as the middle ground of “maybe show”. On the other hand if the two labels were on adjacent levels, we chose the lower one – “maybe show” when the labels were “definitely show” and “maybe show”, and “don’t show” when the labels were “maybe show” and “don’t show”. Thus only utterances that both annotators labeled as “definitely show” were also labeled as “definitely show” in the merged annotation.

We use leave-one-meeting-out cross validation: for each meeting m, we train the classifier on manually labeled utterances from the other meetings, and test the classifier on the utterances of meeting m. We then average the results across all the meetings in the database.

N-Gram features:

As shown in Section 6.4, the strongest features for noteworthiness detection are n-gram features, i.e. features that capture the occurrence of n-grams (consecutive occurrences of one or more words) in utterances. Each n-gram feature represents the presence or absence of a single specific n-gram in an utterance. E.g., the n-gram feature “action item” represents the occurrence of the bigram “action item” in a given utterance. In the previous section we explored “numeric” n-gram features, where each n-gram feature captured the frequency of a specific n-gram in an utterance. In this section, we use boolean-valued n-gram features to capture the presence/absence of n-grams in utterances. We do so because in tests on separate data, boolean-valued features out-performed frequency-based features, perhaps due to data sparseness. Before n-gram features are extracted, utterances are normalized: partial words, non-lexicalized filler words (like “umm”, “uh”), punctuations, apostrophes and hyphens are removed, and all remaining words are changed to upper case. Next, the vocabulary of n-grams is defined as the set of n-grams that occur at least 5 times in the entire dataset of meetings, for n-gram sizes of 1 through 6 word tokens. Finally, the occurrences of each of these vocabulary n-grams in an utterance are recorded as the feature vector for that utterance. In our dataset, there are 694 unique unigrams that occur at least 5 times across the 6 meetings, 1,582 bigrams, 1,065 trigrams, 1,048 4-grams, 319 5-grams and 102 6-grams. In addition to these n-gram features, for each utterance we also include the number of Out of Vocabulary n-gram - n-grams that occur less than 5 times across all the meetings.

Overlap-based features:

We assume that we have access to the text of the agenda of the test meeting, and also the text of the notes taken by the participants in previous meetings (but not those taken in the test meeting). Since these artifacts are likely to contain important keywords we compute

Metric	Definitely show	Maybe show	Don't show
Precision	0.21	0.47	0.72
Recall	0.16	0.40	0.79
F-measure	0.16	0.43	0.75
Accuracy	61.4%		

Table 6.4: Results of 3-Way Classification.

two sets of overlaps features. In the first set we compute the number of n-grams that overlap between each utterance and the meeting agenda. That is, for each utterance we count the number of unigrams, bigrams, trigrams, etc that also occur in the agenda of that meeting. Similarly in the second set we compute the number of n-grams in each utterance that also occur in the notes of previous meetings. Finally, we compute the degree of overlap between this utterance and other utterances in the meeting. The motivation for this last feature is to find utterances that are repeats (or near-repeats) of other utterances – repetition may correlate with importance.

Other features:

In addition to the n-gram and n-gram overlap features, we also include term frequency – inverse document frequency (tf-idf) features like we did in the previous section. These features capture the information content of the n-grams in the utterance. Specifically we compute the TF-IDF of each n-gram (of sizes 1 through 5) in the utterance, and include the maximum, minimum, average and standard deviation of these values as features of the utterance. We also include speaker-based features to capture who is speaking when. We include the identity of the speaker of the current utterance and those of the previous and next utterances as features. Lastly we include the length of the utterance (in seconds) as a feature.

6.5.5 Results and Analysis

Table 6.4 presents the accuracy, precision, recall and f-measure results of the 3-way classification task. (We use the Weka implementation of SVM that internally devolves the 3-way classification task into a sequence of pair-wise classifications. We use the final per-utterance classification here.) Observe that the overall accuracy of 61.4% is only 11% lower relative to the accuracy obtained by comparing the two annotators' annotations (69%, Table 2). However, the precision, recall and f-measure values for the "definitely show" class are substantially lower for the predicted labels than the agreement between the two annotators.

Comparing What	R1-Fmeasure
Definitely show	0.43
Definitely or maybe show	0.63

Table 6.5: ROUGE Scores for the 3-Way Classification.

The numbers are closer for the “maybe show” and the “don’t show” classes. This implies that it is more difficult to accurately detect utterances labeled “definitely show” than it is to detect the other classes. One reason for this difference is the size of each utterance class. Utterances labeled “definitely show” are only around 14% of all utterances, thus there is less data for this class than the others. We also ran the algorithm using manually transcribed data, and found improvement in only the “Definitely show” class with an f-measure of **0.21**. This improvement is perhaps because the speech recognizer is particularly prone to getting names and other technical terms wrong, which may be important clues of noteworthiness.

Table 6.5 presents the ROUGE-1 F-measure scores averaged over all the meetings. Similar to the inter-annotator agreement computations, we computed ROUGE between the text of the utterances labeled “definitely show” by the system against that of utterances labeled “definitely show” by the two annotators. (We computed the scores separately against each of the annotators in turn and then averaged the two values.) We did the same thing for the set of utterances labeled either “definitely show” or “maybe show”. Observe that the R1-F score for the “definitely show” comparison is nearly 50% relative higher than the trivial baseline of labeling every utterance as “definitely show”. However the score is 30% lower than the corresponding inter-annotator agreement. The corresponding R1-Fmeasure score using manual transcriptions is only marginally better – 0.47. The set of utterances labeled either definitely or maybe shows (second row of table 6.5) does not outperform the all-utterances baseline when using automatic transcriptions, but does so with manual transcriptions, whose R1-F value is 0.74.

These results show that while the detection of definitely show utterances is better than the trivial baselines even when using automatic transcriptions, there is a lot of room for improvement, as compared to human-human agreement. Although direct comparisons to other results from the meeting summarization literature are difficult because of the difference in the datasets, numerically it appears that our results are similar to those obtained previously. [54] uses Rouge-1 F-measure solely, and achieve scores between 0.6 to 0.7. [33] also achieve Rouge-1 scores in the same range with manual transcripts.

The trend in the results for the two 2-way classifications is similar to the trend for the inter annotator agreements. Just as inter-annotator accuracy increased to 87% for the “definitely show” vs. “others” classification, so does accuracy of the predicted labels increase to **88.3%**. The f-measure for the “definitely show” class falls to **0.13**, much lower than the inter-annotator f-measure of 0.53. For the “don’t show” vs. “others” classification, the automatic system achieves an accuracy of **66.6%**. For the “definitely plus maybe” class, the f-measure is **0.59**, which is 22% relatively lower than the inter-annotator f-measure for that class. (As with the 3-way classification, these results are all slightly worse than those obtained using manual transcriptions.)

Useful Features

In order to understand which features contribute most to these results, we used the Chi-Squared test of association to find features that are most strongly correlated to the 3 output classes. The best features are those that measure word overlaps between the utterances and the text in the agenda labels and the notes in previous meetings. This is not a surprising finding – the occurrence of an n-gram in an agenda label or in a previous note is highly indicative of its importance, and consequently that of the utterances that contain that n-gram. Similar to the results in the previous section, max and average TF-IDF scores are also highly ranked features. These features score highly for utterances with seldom-used words, signifying the importance of those utterances. Domain independent n-grams such as “action item” are strongly correlated with noteworthiness, as are a few domain dependent n-grams such as “time shift problem”. These latter features represent knowledge that is transferred from earlier meetings to latter ones in the same sequence. The identity of the speaker of the utterance does not seem to correlate well with the utterance’s noteworthiness, although this finding could simply be an artifact of this particular dataset.

6.5.6 Summary

In this section we explored a 3-way noteworthiness annotation scheme as a counter-point to the traditional 2-way scheme explored in the previous section. We showed that using a 3-way scheme, annotators achieved stronger inter-annotator agreements than is typical in the field of meeting summarization annotation. We used this annotated data to train various classifiers. With the three-way classifier, the F-measure of the “definitely show” class, which is similar to the “noteworthy” class in the previous section, did not experience a substantial improvement, going from 0.14 in the previous section (Section 6.4.6) to 0.16 in this section. This lack of improvement can be attributed to the lack of improvement in recall; precision improved by 40% from 0.15 to 0.21. At the same time, in the 2-way “don’t show” vs “others” classification, the “definitely plus maybe” class was accurately identified (F-measure of 0.59). This class can be thought of as representing a compromise between the “noteworthy” and “show-worthy” classes of the previous section.

In the next section we turn our attention to automatically extracting labeled data to train noteworthiness detection models from participants’ notes in previous meetings.

6.6 Learning Noteworthiness by Extracting Implicit Supervision from Notes

6.6.1 Introduction

In the two previous sections we have trained noteworthy-utterance classifiers using 2-way (Section 6.4) and 3-way (Section 6.5) annotation schemes. In both cases, we have made use of manually annotated data. In a real usage setting, such manual annotations will not be available. Our research goal is to develop a system that can learn to identify noteworthy utterances for a particular sequence of related meetings by automatically extracting labeled data. In this section we explore the extraction of labeled data from meeting participants' notes in earlier meetings. That is, we treat the notes that participants have taken in the past as an *implicit* form of supervision for the task of identifying noteworthy utterances in future related meetings.

In this section, we first present a simple approach to extracting labeled data in the form of utterances labeled as noteworthy or not from previously recorded meetings. We then train a noteworthiness detector on this labeled data, and compare its performance against that of classifiers trained on manually labeled data.

6.6.2 Extracting Supervision from Notes

Introduction

As mentioned above, our goal is to automatically extract labeled data from the notes taken in previously recorded meetings. We call the component that performs this task the *supervision extractor*. The input to this component is a dataset of recorded meetings consisting of automatically transcribed utterances and notes taken by the participants in the meetings. The output of this component is the set of utterances in those meetings that should be labeled as “noteworthy”. We will later train a classifier on this labeled data, and use it to identify noteworthy utterances in the test meetings.

As before, we define an utterance as “noteworthy” if one or more participants are willing to include its contents in the meeting notes. Thus to find the noteworthy utterances in the previously recorded meetings, we need to find the utterances whose contents were included in the notes of those meetings. In this section we present a simple bags-of-n-grams-based approach to finding these utterances.

Algorithm 1 Supervision Extraction Algorithm

```
 1: UttList $\Leftarrow (u|\forall u \in$ meeting utterances$)$
 2: Use n-gram precision to compute a score for each utterance in UttList
 3: Sort utterances in UttList in descending order according to their n-gram precision
 4: bestN-GramFmeasure $\Leftarrow 0$
 5: indexOfBestN-GramFmeasure $\Leftarrow -1$
 6: for i = 1 to |UttList| do
 7:     currentSetScore $\Leftarrow$ N-Gram Fmeasure of utterance set $\{u_j|u_j \in$ UttList, $1 \leq j \leq i\}$
 8:     if currentSetScore $>$ bestN-GramFmeasure then
 9:         bestN-GramFmeasure $\Leftarrow$ currentSetScore
10:         indexOfBestN-GramFmeasure $\Leftarrow$ i
11:     end if
12: end for
13: return $\{u_j|u_j \in$ UttList, $1 \leq j \leq$ indexOfBestN-GramFmeasure$\}$
```

Supervision Extraction Algorithm

We use algorithm 1 to label utterances as "noteworthy" or "not noteworthy" based on the notes in the meeting. The inputs to this algorithm are the utterances and the notes taken at a particular meeting. The output is the set of utterances that should be labeled "noteworthy"; the remaining utterances not output by this algorithm should be labeled as "not noteworthy". This is a greedy algorithm, and it proceeds in two main steps:

1. **Rank Utterances**: First we rank the meeting utterances in descending order of their degree of overlap with the notes in that meeting (steps 1, 2 and 3 in algorithm 1). There are several ways to measure overlap: the absolute number of n-grams overlapped, the number of overlapping n-grams as a fraction of the number of n-grams in the utterance (n-gram precision), or as a fraction of the number of n-grams in the notes (n-gram recall). Choosing different metrics results in different rankings because utterances vary widely in length. We choose n-gram precision as the utterance ranking metric because it tends to give high rank to utterances that were noted down nearly verbatim, even if the utterances themselves are short. These utterances can most confidently be labeled as "noteworthy".

2. **Greedily Pick Utterances**: In the second phase (remaining steps in algorithm 1), we pick the first n utterances from this ranked list that together have the highest n-gram fmeasure (harmonic mean of n-gram precision and recall). At each index number i of the sorted utterance list, we compute the fmeasure of the union of utterances in the list from index number 1 to i. We do so by considering the text of all those utterances as a single string of text, computing the n-gram precision and recall of

that text against the notes of the meeting, and then computing n-gram fmeasure from these two values. We do this for all index numbers in the list, find the index number with the highest n-gram fmeasure score, and return utterances from index number 1 to that index number as the utterances that should be labeled as “noteworthy”.

Discussion of Algorithm

Observe that although we sort the utterances by n-gram precision, we use n-gram fmeasure to evaluate each successive list of utterances. We do not use n-gram precision for this evaluation because it is likely to pick only the first utterance in the sorted utterance list, since labeling any more utterances as noteworthy would lower the precision of the utterance set. On the other hand, n-gram recall is likely to pick most utterances that have even one n-gram overlapped with the notes since adding utterances with overlaps increases recall. Fmeasure strikes a balance between these two extremes. (Other balance points could be struck, by giving higher weight to precision than recall, for example, but we have only experimented with equal weights.)

In the typical extractive summarization setting, a length cut-off is needed in order to decide how many utterances to return as “in summary”. In our case, since we have the notes to compare against, we can use fmeasure as a natural stopping condition, and thus do not need a length constraint. As a consequence, we avoid having one more parameter to tune.

In the algorithm above, we have not specified the size of the n-gram to be used. In fact, we repeat the above process with n-grams of size 1 through 6. Thus for each n-gram size, the algorithm returns a set of utterances to be labeled as “noteworthy”. In section 6.6.4 we show that these different utterance sets overlap to some degree, and that merging them together to form the final “noteworthy” utterance set creates labeled data of the best quality.

We normalize the texts of the utterances and notes in order to ensure better n-gram matching. Specifically, we remove punctuations, convert all words to lower-case, replace words with their root forms (using the Porter stemmer), and remove low-information-content words such as articles and prepositions.

Note that the n-gram F-measure used in this algorithm is essentially similar to the ROUGE-1 F-measure metric [27] that we have used before. (ROUGE has more built-in features that can also take into account multiple references). ROUGE-F1 has been used to create labeled data in the past, for example in [40].

The complexity of the greedy utterance-picking portion of the above algorithm is $O(N)$ (N is the number of utterances in the meeting), since fmeasure is computed N times. (The algorithm’s overall complexity is $O(Nlog(N))$ since it is dominated by the need to sort the utterances by their n-gram precision scores.)

Our next step is to evaluate the quality of this automatically extracted labeled data. In the next section we present the data used to do so.

6.6.3 Data and Annotations for Experiments

For experiments, we use the same dataset used in the previous chapters. Recall that all meetings were natural and did not follow pre-determined scenarios; they were in fact regular meetings held by a particular research group. Consequently there is continuity in topics and meeting participants from one meeting to the next. The 10 meetings had 6 unique participants, between 2 and 5 of whom showed up at any one meeting (mean of 3.8). The topics of conversation in these meetings revolved around software components of a large system. While meetings all had written agendas, participants followed them loosely.

The meetings lasted a mean of 31 minutes, yielding slightly over 5.5 hours of data. The speech was manually transcribed and checked by transcribers. The audio was segmented into utterances at silences longer than 0.5 seconds, yielding a total of 7,544 utterances; these utterances are the units of classification. The meetings had on average 24 lines of notes each. Our speech-and-notes-recording system provides a shared note-taking interface that is visible live to all meeting participants during the meeting. Multiple participants can simultaneous write and see others' notes. As a result we have one set of notes per meeting, jointly written by the different meeting participants.

In this section, we report results using both manual transcripts and transcripts output by the Sphinx automatic speech recognizer (ASR), using speaker independent acoustic models, and language models trained on both publicly available meeting data and the words in the meeting notes. Recall that the word error rate of the automatic transcripts is 40%.

Although the algorithm above does not need labeled data, we still require labeled data for evaluation purposes. We will use two sets of annotated data that we have used in previous sections, but we will rename them for ease of explanation.

Manual Alignments

The first labeled data, which we will name as "Manual Alignments" in this section, is the same as those used in the noteworthiness baseline experiments in Section 6.4. Recall that for these annotation, annotators were asked to find the smallest set of utterances that contained as much as possible of the information in the notes of that meeting. Thus, in this set of annotations, human annotators were doing the same task that the supervision extractor above performs – given the notes, find the utterances that contributed to those notes. A diagrammatic representation of this alignment is shown in Figure 6.2. A total of 532 utterances across the 10 meetings had contents judged to have been included in the notes. That is, only 7% of the utterances of the meetings were deemed to have been

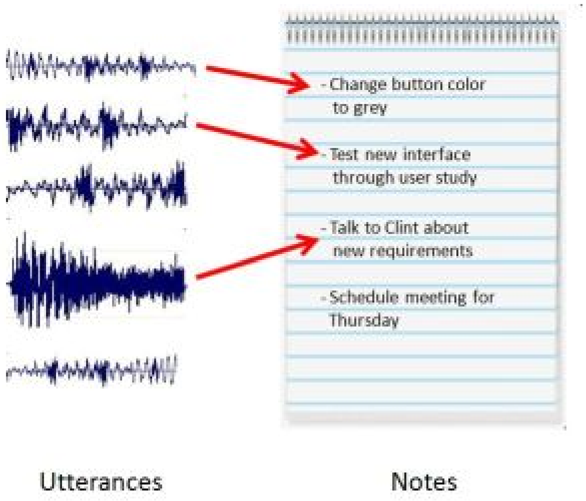

- Change button color to grey
- Test new interface through user study
- Talk to Clint about new requirements
- Schedule meeting for Thursday
Utterances
Notes

Figure 6.2: Diagrammatic representation of Manual Alignments, in which utterances, depicted on the left, are manually aligned to individual lines of notes, on the right, that contain the same information.

included in the notes. This "compression level" is similar to those reported in prior meeting summarization work [33].

Manual Classifications

As mentioned before, although the utterances whose contents were included in the notes can be considered as noteworthy, there may be other utterances whose contents were not included in the notes but that are also noteworthy. This is likely because note-taking is a distracting task and meeting participants routinely choose to omit some information from their notes that they would have included if note-taking was easier. In order to identify *all* utterances that contain noteworthy information, we will use the multilevel noteworthiness annotation done in the previous section, Section 6.5. Recall that in this annotation, utterances were labeled with three different classes – "Definitely Show": utterances whose contents are noteworthy enough that they should definitely be suggested to meeting participants, "Don't show": utterances whose contents do not have any information worth showing to the meeting participants, and "Maybe show": utterances with borderline noteworthy content that a system may choose to show to the participants or not. In this section, however, we do not use all three classes of annotation, and instead merge the "Maybe show" and "Don't show" classes into a single class of utterances. This is similar to the "Definitely vs Others" merged annotations that we experimented with in the previous section. We will call

Utterances

Figure 6.3: Diagrammatic representation of Manual Classifications, in which utterances are labeled as noteworthy (check marks) or not (cross marks), without reference to a set of notes.

Data Labeler	ROUGE
Supervision Extractor	0.51
Manual Alignment	0.31
Manual Classification	0.28

Table 6.6: Mean ROUGE scores computed for different utterance sets against meeting notes.

this annotation "Manual Classifications". A diagrammatic representation of this annotation is shown in Figure 6.3. Across the 10 meetings, 561 utterances (7.4% of all utterances) were identified as "Definitely Show". To study inter-annotator agreement, we used two annotators over 5 meetings and found an inter-annotator Kappa value of 0.76, which is relatively high for meeting summarization annotation, as compared to prior work such as [29, 36].

6.6.4 Analysis of the Extracted Supervision

Before we test the labeled data output by the Supervision Extractor by training and evaluating a noteworthiness detector from that data, we first present an analysis of the data to gain insights into the automatic alignment process.

Evaluating Against	Prec	Rec	F
Manual Alignment	0.28	0.34	0.30
Manual Classification	0.29	0.33	0.29

Table 6.7: Average utterance-level precision, recall and f-measure of utterances labeled as "noteworthy" by the Supervision Extractor, computed against different manual annotations.

Comparison against Participant Notes

We start by measuring the overlap between the text of the utterances labeled "noteworthy" by the automatic supervision extractor, and the text of the notes in the meetings. Note that this measurement does not form an evaluation of the supervision extractor, since the extractor had access to the notes *and* was designed to maximize the same metric (the degree of overlap between noteworthy utterances and the notes). We present this measurement merely to form a benchmark against which to compare later evaluations (such as those presented in Table 6.11).

Similar to evaluation of extractive summarization, we use ROUGE-F1 to measure the degree of overlap between the text of the utterances labeled as noteworthy by the supervision extractor and the meeting participants' notes. This result – 0.51 – is presented in the first row of Table 6.6. We compare this value against two benchmarks. Recall that in Manual Alignment, human annotators labeled those utterances as "noteworthy" whose contents, in their judgment, were included in the notes. The second row of Table 6.6 shows that "noteworthy"-labeled utterances in this Manual Alignment annotation achieved a ROUGE score of only 0.31. Thus the Supervision Extractor outperforms the Manual Alignment by 64%. Although this is seemingly a case of a system outperforming humans at the same task, it is not a surprising result because the human annotator takes meaning into account, not just word overlaps, whereas the system is designed to pick utterances that maximize the value of the ROUGE-F1/n-gram fmeasure metric.

As a point of interest, we also report the ROUGE-F1 score obtained by the noteworthy utterances in the Manual Classification annotation. Recall that this annotation was performed without access to the notes – consequently the ROUGE-F1 score for it is even lower on average (third row of Table 6.6).

Comparison against Manual Alignments and Manual Classifications

In addition to measuring the overlap between the text of noteworthy utterances and the notes, we can directly count the number and fraction of utterances that overlap between the utterances labeled "noteworthy" by the Supervision Extractor and the human annotator (in

Evaluating Against	Using Manual Transcripts	Using ASR Transcripts
Manual Alignment	0.30	0.20
Manual Classification	0.29	0.19

Table 6.8: Utterance-level F-measure scores of noteworthy utterances extracted by the Supervision Extractor, when working from manual transcriptions versus transcriptions created by an automatic speech recognizer (ASR).

the Manual Alignment annotations). That is, instead of counting n-gram overlaps, we can report stricter evaluation in terms of true and false positives and negatives at the utterance level. Such an evaluation gives us a sense of how closely the Supervision Extractor's output resembles the human annotations.

The first row of Table 6.7 shows the precision, recall and f-measure values obtained by the automatically labeled "noteworthy" utterances, compared to Manual Alignment anotations. (For these results, the input to the Supervision Extractor was manually transcribed utterances.) This row shows that 28% of the utterances automatically labeled as "noteworthy" were also designated as "noteworthy" by the human annotators, while 34% of utterances so designated by the human were found by the Supervision Extractor. This implies that nearly two-thirds of the utterances labeled "noteworthy" by the human were not found by the Supervision Extractor; these are utterances whose semantic contents were included in the notes, but did not have enough word-wise overlaps to be found by the Supervision Extractor. Row 2 of Table 6.7 shows that the Supervision Extractor achieves similar accuracy when compared against utterances labeled "noteworthy" in the Manual Classifications annotation (which were done by the human without access to the notes).

When ASR-based transcripts are used as input to the Supervision Extractor, the resulting F-measure scores are predictably lower, as shown in table 6.8. Specifically the score falls by 33% when evaluated against Manual Alignment annotation, and by 34% when evaluated against Manual Classifications. This is not surprising – automatic speech recognition inserts word errors that lead to poorer n-gram matches between utterances and notes, resulting in poorer supervision extraction.

Disaggregating the ROUGE Scores

To further compare the utterance sets that are aligned to the notes by the Supervision Extractor and the human annotators, we partition the utterances into four sets according to whether they were aligned to the notes or not by the Supervision Extractor and in the Manual Alignments. We then compute the ROUGE value of each of these sets; these scores are presented in table 6.9. Observe that the ROUGE scores of utterances aligned to the notes

Manual	**Supervision Extractor**	
Alignments	Aligned	Not Aligned
Aligned	0.44	0.34
Not Aligned	0.46	0.07

Table 6.9: ROUGE scores of utterances disaggregated by whether they were aligned or not by the Supervision Extractor and by the Manual Alignments.

N-Gram Size	**Prec**	**Rec**	**F**
1	0.26	0.28	0.26
2	0.47	0.20	0.27
3	0.55	0.11	0.18
4	0.56	0.05	0.09
Combined	0.28	0.34	0.30

Table 6.10: Utterance-level precision, recall and f-measure of utterances labeled as "noteworthy" by the Supervision Extractor when using different n-gram-sizes. Values in this table were computed against Manual Alignments annotations.

by the Supervision Extractor (first column of the table), are higher than the other two cells, as a consequence of the design of the Supervision Extractor. Also observe that utterances not aligned by the supervision extractor but aligned in the Manual Alignments have low ROUGE score (top right cell of the table); these are utterances that share the meaning but not the words of the notes they are aligned to. Utterances that were aligned by the Supervision Extractor but not by the human annotator receive a high ROUGE score (bottom left cell of the table) implying that they share many words with the notes. There are several reasons why they were not aligned to the notes – some are simply repeated information (recall that the human annotator was asked to pick the *smallest* set of utterances that contain all the information in the notes), some referred to topics without contributing any new information worth noting down, etc. Distinguishing between these cases is beyond the scope of a simple n-gram-based approach. Finally note that utterances that are not aligned to the notes by either aligner have very poor ROUGE score (bottom right cell of the table), implying that there are many utterances whose contents are never referred to in the notes.

Using Different N-Gram Sizes during Supervision Extraction

As mentioned in section 6.6.2, we ran the Supervision Extractor multiple times, each time using a different n-gram size to compute n-gram-precision and n-gram-fmeasure. We evaluated the utterance set labeled as “noteworthy” for each n-gram size by computing precision, recall and fmeasure against the Manual Alignments; the results are presented in Table 6.10. We observe that the precision of the output utterance set steadily rises from 0.26 for unigram-based n-gram-precision/fmeasure to 0.56 for 4-gram based metrics. Recall on the other hand falls from 0.28 to 0.05. At the same time, beyond 2-grams, the Supervision Extractor returns fewer utterances as “noteworthy”. These various effects – higher precision, lower recall, and lower number of utterances labeled as noteworthy as higher and higher n-gram sizes are used – are graphically shown in Figure 6.4.

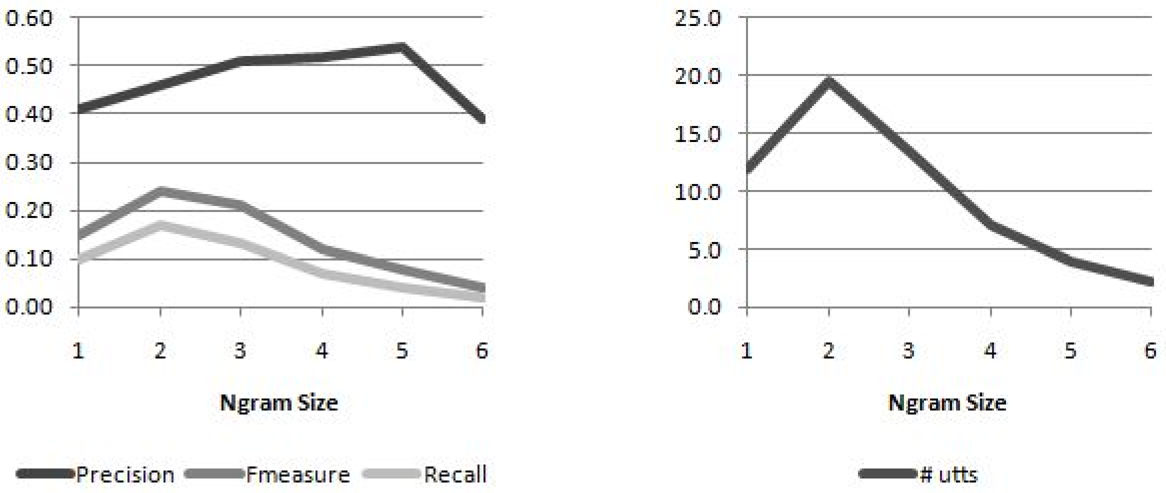


Figure 6.4: The figure on the left shows the increase precision values and decreasing recall and F-measure values as larger and larger n-gram sizes are used. (These scores were computed against the Manual Alignment annotations.) The figure on the right shows variation in the number of utterances in the highest-scoring set output by the Supervision Extractor for different sizes of n-grams.

The sets of utterances labeled as “noteworthy” by the Supervision Extractor using different n-gram sizes are not subsets of each other. For example, utterances of equal size that have the same number of words overlapping with the notes get the same score when unigram fmeasure is computed, but if one of them has higher order n-grams overlapping with the notes, they will outscore the other utterances if fmeasure with higher order n-grams is computed during supervision extraction. These different utterance sets can be combined in different ways; for now we simply merge these utterance sets together, and note that the combined utterance set has the highest fmeasure score (last row of Table 6.10).

6.6.5 Learning from Extracted Supervision

Classification Approach

We now consider training a noteworthiness detection model from the automatically extracted supervision data. The goal is to train a model on the utterances automatically labeled by the Supervision Extractor as “noteworthy” and “not noteworthy” based on the notes in previous meetings, and to apply this model to identify noteworthy utterances in future related meetings. As mentioned earlier, the motivation is to help meeting participants take notes in future meetings, by learning from their notes in previous meetings.

Similar to the previous sections, we take a classification approach to the problem. That is, the input to the model is an utterance (and features derived from other relevant information in the meeting), and the output is a binary label: “noteworthy” or “not noteworthy”. As done in Section 6.5, we train a **Support Vector Machine** based classification model using a linear kernel.

Features

The features we use for this classification task are similar the ones used in the previous section (Section 6.5. These are:

N-Gram Features: those n-grams that occur at least 5 times across the entire dataset, with $1 \leq n \leq 6$. For each n-gram, we define a boolean valued n-gram feature that is set to 1 if the n-gram occurs at least once in the input utterance, and to 0 otherwise. We use boolean valued n-gram features because these were shown to outperform frequency-based features.

Overlap-based Features: the number and fraction of n-grams in the current utterance that overlap with the text of the agenda of the meeting, the notes taken in the previous meetings, and other utterances in the same meeting. We represent the overlaps with each of these three sources as three sets of features. We use these features since texts such as previous meetings’ notes and agenda items are likely to contain topic-specific phrases that could signal utterance noteworthiness.

Other Features: features that were shown to correlate with noteworthiness in previous sections. These include the minimum, maximum and average term frequency – inverse document frequency values of the n-grams in the utterance. We include speaker identity-based features such as the speaker of the input meeting, the speakers of the previous and next utterances, etc. The length of the utterance is also used.

Classifier	Evaluated Against			
	Notes	Manual Classifications		
	ROUGE-F1	P	R	F
Implicitly Supervised	0.37	0.31	0.18	0.21
Manual-Alignments-Based	0.29	0.21	0.13	0.15
Manual-Classifications-Based	0.28	0.31	0.22	0.23
Random Classifier	0.24	0.17	0.04	0.06

Table 6.11: Evaluation of the outputs of the 4 classifiers against notes (using ROUGE-F1), and against utterances labeled "noteworthy" in the Manual Classifications (using utterance-based precision, recall and f-measure). Manual transcripts were used for these results.

Experimental Methodology

We use the same data for experimentation as described above. We perform leave-one-meeting-out cross validation to maximize the data available for training. We treat each meeting as the test meeting in turn. We use the Supervision Extractor to extract labeled data from the utterances and notes of the remaining 9 meetings, and train the SVM-based noteworthiness classifier on this automatically extracted labeled data. We call this classifier the "Implicitly Supervised Classifier". We then use this "Implicitly Supervised Classifier" to classify the utterances of the test meeting as noteworthy or not, and evaluate this classification against the notes as well as the manual annotations in this test meeting. We repeat the whole process for each of the 10 meetings, and report average evaluation scores.

In addition to the Implicitly Supervised Classifier, we train and evaluate two more "benchmark" classifiers – the "Manual-Alignments-Based Classifier" and the "Manual-Classifications-Based Classifier". For both these classifiers, we follow the same cross-validation procedure. The "Manual-Alignments-Based Classifier" is trained on utterances manually labeled as "noteworthy" or not based on the notes taken by the original meeting participants. Evaluation of this classifier thus indicates how well the "Implicitly Supervised Classifier" would perform, if the Supervision Extractor were replaced by a human annotator. The "Manual-Classifications-Based Classifier" is trained on utterances manually labeled as "noteworthy" or not, regardless of the actual notes taken at the meeting. Evaluation of this classifier shows how well a noteworthiness classifier would perform, if it had the correct noteworthiness labels of *all* utterances rather than only those utterances that the meeting participants had the time to include in their notes. This full training and evaluation paradigm is shown in Figure 6.5.

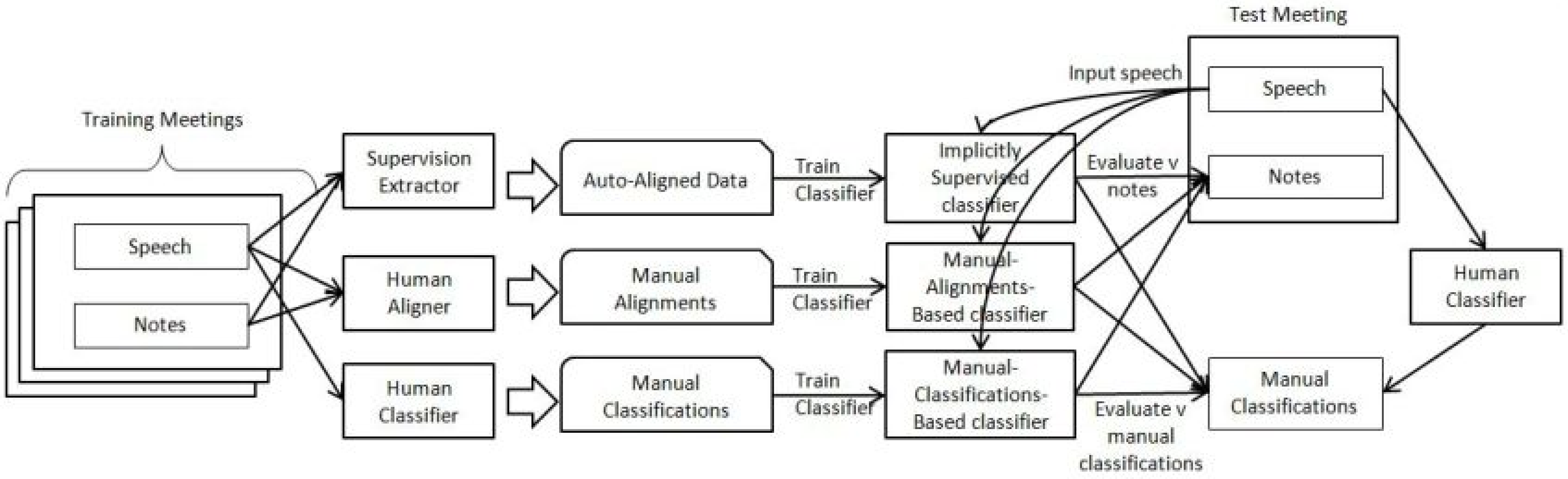


Figure 6.5: The training/testing paradigm used in this section. This figure shows the training and testing for a single fold of the evaluation. In each such fold, one meeting is designated as the test meeting, and the remaining meetings as training meetings. From those training meetings, labeled data is automatically extracted to train the Implicitly Supervised classifier. Separately humans create Alignments and Classifications, based on which two other classifiers are trained. These three classifiers are then used to classify the utterances of the test meeting and are evaluated against the notes and the manual classifications of that meeting.

As a sanity check, we include results of a random baseline. We replace the supervision extraction module with a random extractor that labels a randomly chosen sub-set of utterances as “noteworthy”. The number of utterances to label as “noteworthy” is set at the number of utterances originally labeled “noteworthy” by the human in the Manual Alignments. The “Random Classifier” is then trained on this randomly labeled data.

We evaluate these classifiers, by comparing their output against two versions of ground truth. First, we compare them to the notes taken in the meetings by the original meeting participants. This evaluation shows to what degree we can predict the information that the meeting participants themselves noted down. Second, we compare the classifier outputs to utterances labeled “noteworthy” in the Manual-Classifications annotation. As we have mentioned before, it is expected that meeting participants will be willing to take more notes if they have a notes-assistance system. The Manual-Classifications annotations captures the noteworthiness of *all* the utterances in the meetings, regardless of whether they were noted down by the meeting participants. Comparing against those utterances shows to what degree these classifiers approach the ultimate goal of finding all noteworthy information in the meetings.

Results: Comparison against Notes

The first results column of Table 6.11 shows the ROUGE-F1 metric evaluation of the classifiers against the notes originally taken at the meetings. This value quantifies the degree to which the contents of the utterances classified as noteworthy overlap with those of the notes actually taken by participants.

Observe that the Implicitly Supervised classifier outperforms all other classifiers on this metric. For all results in this table we use two-sample bootstrap to measure significance. Using this method, the Implicitly Supervised Classifier is marginally significantly better than the Manual-Alignments-Based and Manual-Classifications-Based classifiers ($p < 0.1$) and significantly better than the random classifier. This is a significant result: it implies that a classifier trained on automatically extracted labeled data *outperforms* one trained on manually annotated data, if the task is to create notes similar to the ones taken by the meeting participants in the test meeting. The classifiers trained on manual annotations have lower performance because manual annotations do not overlap as well with the notes to begin with, as shown in Table 6.6. Instead, as discussed earlier, human annotators take meaning into account when labeling utterances as “noteworthy”. It appears from this result that if the goal is to create notes similar to the ones that a participant would take in the meeting *without any assistance*, one is better off using automatically extracted labeled data than using manual annotations. These results decay somewhat when using ASR-transcribed utterances, with the Implictly Supervised Classifier scoring a ROUGE-F1 score of 0.31 when compared against the notes.

Results: Comparison against Manual Classifications

Recall that utterances were labeled as noteworthy in the Manual Classifications annotation based on each utterance's inherent noteworthiness, and not on what notes were taken in the meeting. This evaluation tells us the degree to which a classifier trained on automatically extracted supervision from notes in previous meetings can accurately judge noteworthiness of utterances in future meetings, even when they talk about topics not noted down in previous meetings. These results are presented in the second group of columns labeled "Manual Classifications" in Table 6.11.

There is no significant difference between the precision, recall and fmeasure of the Implicitly-Supervised Classifier and the Manual-Classifications-Based Classifier, while the Manual-Alignments-Based Classifier is marginally poorer. This is a somewhat surprising result, since the Manual-Classifications-Based classifier has access to *complete* utterance by utterance labels in the previous meetings, but yet is unable to outperform the implicitly supervised classifier. One possible reason for this result is that in our dataset meeting participants talked about closely related topics from meeting to meeting. In other datasets, where topics change significantly from meeting to meeting, the Manual-Classifications-Based Classifier is likely to outperform the Implicitly Supervised Classifier. Equally surprising is the weak result of the Manual-Alignments-Based classifier. Recall that there are far fewer utterances labeld as "noteworthy" in Manual Alignments, as compared to Manual Classifications. This difference in the amount of positively labeled data may perhaps explain this disparity in results.

These results also decay to some degree when using ASR-based transcripts, going down from an F-measure of 0.21 to only 0.09, which is moderately significantly more than the random classifier. This shows that speech recognition errors compound from the label data extraction to the training of the classifier on that error-filled extracted data.

Feature Analysis

An inspection of the normalized features to which the SVM algorithm assigned the largest weights reveals differences between the Implicitly Supervised Classifier and the Manual-Classifications-Based Classifier. The Implicitly Supervised Classifier gave high weights to features that measured overlaps between utterances in the current meeting and notes in previous meetings, and to n-grams that occurred multple times in both notes and utterances across multiple previous meetings. On the other hand, the Manual-Classifications-Based Classifier assigned high weights to n-gram features such as "restarting problem", "feature selection algorithm", etc. These n-grams represent recurring topics across meetings that human annotators denoted as being important to include in the notes. Thus, the two classifiers learn different aspects of the meetings - while the implicitly supervised classifier focusses on n-grams that co-occur often between notes and utterances, the Manual-Classifications-based Classifier learns pragmatics-related features – such as n-grams denoting particular problems that occur over multiple meetings. This is reflected in the fact that of the union of utterances labeled noteworthy by the two classifiers, they agree on only 12%, for a Kappa value of 0.19. A future extension of this work is to look for ways to combine the strengths of these two classifiers, without requiring the user to provide labels. One way to do so may be to make use of active learning, and observe which automatically suggested utterance the user includes in his notes.

6.7 Example Classification Output

We present here the utterances labeled noteworthy by the classifier trained on auto-aligned data for a short meeting segment. Although not chosen randomly, this output is typical of the output of the classifier. (Note again that we are using manual speech transcriptions). Names have been redacted to ensure anonymity. Each indented line is a continuation of the utterance on the previous line.

```
on the model training stuff
I'll have some results for adaptation of switchboard and fisher models
   using different parts of our data
with new acoustic models
and I'll give you the data from this meeting as well
but you know that's significant because it shows that your algorithm
   is actually working in both decoder
so I'm gonna send the audio from this meeting to ***
```

The following are utterances identified as noteworthy by the human annotator, on the same meeting segment as the one above.

```
I'll have some results for adaptation of switchboard and fisher models
   using different parts of our data
I don't know, ***, you've
you've already decoded the last two meetings
and I'll give you the data from this meeting as well
```

Observe the overlaps between utterances labeled noteworthy by the automatically trained noteworthiness detector and the human annotators. Observe too that the false positives can also conceivably be considered as "noteworthy", even though they weren't labeled as such by the human annotators.

6.7.1 Summary

In this section, we have presented an approach to automatically acquiring labeled data from notes taken by the same participants in earlier related meetings. We have shown that the automatically extracted labeled data has higher n-gram overlaps with the notes in the meetings than utterances manually annotated as "noteworthy" by human annotators. We have then trained a noteworthiness detector on this labeled data, and shown that it *outperforms* classifiers trained on manually labeled data on the ROUGE-F1 metric compared to the notes. This result implies that if the goal is to create notes in future meetings that are as close as possible to the notes the same humans would take, using automatically extracted data is preferable to using manual annotations. We conclude that automatically extracting labeled data from meeting notes is a viable approach to learning to detect the noteworthiness of utterances.

The raw fmeasure numbers are low, but as we have shown in the Wizard-of-Oz studies in Section 6.3, a precision level of around 0.3 may be sufficient for the system to be useful as a notes-suggestion system. In the next chapter we evaluate this system within the context of a user study to ascertain whether humans find suggestions based on automatically trained noteworthiness detector useful in creating their notes.

Chapter 7

A Controlled User Study to Evaluate Automatic Note Suggestions

7.1 Motivation

In section 6.3 we had shown through a Wizard-of-Oz study that when a human Wizard identifies “noteworthy” utterances and suggests their contents verbatim to meeting participants, without doing any further editing or gisting, meeting participants are willing to utilize the suggestions to construct their notes. In the previous chapter we have presented a system that learns to detect noteworthy utterances by training on labeled data automatically extracted from participants’ notes in previous meetings. We have presented an offline evaluation of this detector: we have shown that there is significant overlap between utterances classified as noteworthy by the detector and utterances labeled as “definitely” important by human annotators. There is also significant overlap between the text of these utterances and the text of the notes taken by the original participants of the recorded meetings.

Since the automatically trained noteworthiness detection system appears to identify many utterances that humans also think are important, one may expect that a notes-assistance system developed on the basis of such a system could potentially be helpful to meeting participants. However, since the accuracy of the detector is significantly less that that of the human Wizard, it is unclear precisely how helpful such a notes-assistance system would be to meeting participants. In addition, other aspects unrelated to noteworthiness detection – such as interface issues, speech recognition accuracy, etc. – might contrive to lower or completely negate the usefulness of a such system. In this chapter we present a user study undertaken to test the usefulness of note suggestions based on the detector’s predictions, within an actual note-taking environment.

7.2 Goals

Our goal is to measure the impact that in-meeting note-suggestions based on the noteworthy-utterance detector's predictions have on meeting participants' note-taking. Specifically, we aim to answer the following two major questions:

- Is the implicitly trained noteworthy-utterance detector accurate enough such that meeting participants see value in and are willing to utilize suggestions based on its predictions?

- If so, what impact do the automatically generated suggestions have on the *quality* of the notes and on meeting participants' *note-taking effort*?

Orthogonal to these two questions, we also wish to answer the following two questions:

- How do the automatically generated suggestions compare against suggestions made by a human Wizard?

- What impact do speech recognition errors have on the usefulness of the suggestions?

In the following section we describe the design of the user study.

7.3 Design of the User Study

7.3.1 Design Choices

The ultimate goal of a notes-assistance system is to help meeting participants create notes that improve human productivity, e.g. by reducing note-taking effort, improving the likelihood that future questions whose answers were discussed at these meetings are more likely to be answered, and answered faster, etc. Given a suitable concrete definition of productivity, therefore, one choice for the design of a study to evaluate a notes-assistance system is to conduct a long-term longitudinal study: groups of humans would be recruited to use the system in their regular meetings, their note-taking would be monitored, and their productivity would be measured and compared against that of control groups that did not have access to the notes-assistance system. Such a study is unfortunately expensive to run, hard to control for, and is best suited to a more mature software product. For the purposes of this thesis, we conduct a more controlled short-term study.

7.3.2 Chosen Study Design

Specifically, we designed a within-subjects study where participants were asked to use the SmartNotes notes-assistance system to take notes while listening to 4 separate *recorded meeting segments*. During 3 of these meeting segments, note-suggestions were shown to the study participants within the SmartNotes application. (We modified the SmartNotes interface in order to show these suggestions; these modifications are described in more detail in Section 7.6). For each participant, the suggestions for each of these 3 meetings were generated from a different source. The source of the note-suggestions thus formed the experimental manipulation, and "no suggestions" was the control condition. The 3 suggestion sources used for all participants were:

- Implict (manual): Study participants were shown manual transcripts of utterances predicted as "noteworthy" by the implicitly trained noteworthy-utterance detector. (Details on how the detector was trained are presented in the next section). This is the main test condition, and tests the usefulness of the automatically trained system when speech transcriptions are accurate.

- Implicit (ASR): Study participants were shown automatic transcriptions of utterances suggested in the "Implict (manual)" condition above. The goal of this condition is to show the degree to which the usefulness of the implicitly trained system reduces when the utterances are transcribed by an automatic speech recognizer, even though the actual suggestions being shown are exactly the same as those shown in the "Implicit (manual)" condition.

- Wizard (manual): Study participants were shown manual transcripts of all utterances in the meeting segments that were annotated as "Definitely Show" by all human annotators. Since the noteworthiness judgment is provided by humans and the transcripts are accurate, this condition establishes an upper bound on the usefulness of suggestions based on an extractive summarization approach. This condition is similar to the Wizard-of-Study conducted previously.

We balanced the condition order across the participants. Thus each of the 24 participants of the study was exposed to a different order of these 4 conditions. The order of the meetings themselves was kept unchanged across all the participants.

Participants were instructed to take notes such that someone who had not attended the meeting could use the notes to get an overview of the important points discussed at the meeting. (More details on the instructions to participants are presented in Section 7.5). Since participants are likely to be unfamiliar with the topics discussed at the meetings, they were provided with "Scaffolding Text" before each meeting. This text included the names of the meeting participants, and brief descriptions of the names and technical terms discussed during the meetings. To simulate a real meeting, participants were not allowed to pause or rewind the meeting recording.

After each meeting segment, participants were asked to complete a questionnaire to express their opinions on 5-point Likert-like scales regarding various aspects of the suggestions in the preceding meeting. The aspects of the suggestions tested included the frequency, intelligibility, timeliness, and usefulness of the suggestions, and the degree of distraction caused by them. In addition, participants were also asked to complete two other survey questionnaires, one at the very beginning of the experiment, and one at the very end. The goal of the questionnaire presented at the beginning of the experiment was to gather demographic information about the participants, and information about their note-taking habits in meetings – how often they attended meetings, how often they took notes in them, how they rated their own note-taking skills, etc. The questionnaire presented at the end of the experiment asked the participants to express their opinions about various aspects of the SmartNotes interface, including suggestion acceptance mechanisms, the note-taking interface, etc. More details about these questionnaires, and aggregate results obtained from them are presented in the results section below.

The full experiment took about 1.5 hours. At the end of the experiment, participants were paid $15. We did not employ performance-based compensation incentives because note-taking is subjective in nature and difficult to evaluate quickly and on absolute terms. In addition, because participants could not pause or rewind the meetings, there was a natural limit to the amount of time they could take to create notes, and there was no need to incentivize them to complete the tasks in a timely fashion.

7.3.3 Advantages and Disadvantages of the Study Design

The main advantage of this study design is that it allowed us to control for several variations. While different meetings are likely to have different quantities of important information, we chose 4 meeting segments that all had roughly similar amounts of noteworthy information. In addition, although the performance of the implicit detector varies from meeting to meeting, the meeting segments were chosen to ensure that the average performance over the 4 meeting segments was close to the average performance on the full dataset. (More details on the selection of meeting segments are presented below). Yet another advantage of using pre-recorded meetings is that it allowed us to process the meetings and create the suggestions offline instead of doing so as the meeting progresses. We used this opportunity to use manual speech transcriptions in one of the conditions, in order to test the impact of automatic speech recognition errors on the usefulness of the suggestions.

One disadvantage of this design is that the task that study participants perform is an approximation of, but not exactly the same as the task that meeting participants perform. While meeting participants have to typically engage in the conversation by speaking and

Meeting	Segment Length		#	# Suggestions		Detector Perf.		
Segment	# Secs	# Utts	Spkrs	Wizard	Implicit	Prec	Rec	F1
1	301	101	3	19	8	0.75	0.32	0.45
2	304	118	2	29	7	0.29	0.07	0.11
3	300	141	4	20	7	0.43	0.15	0.22
4	315	140	2	17	7	0.29	0.12	0.17
Mean	305.0	125.0	2.8	21.3	7.3	0.44	0.17	0.24
Stdev	6.5	19.2	1.0	5.3	0.5	0.22	0.11	0.15

Table 7.1: Information on meeting segments selected for the user study.

responding to other participants, user study participants were free to focus entirely on the note-taking task. Indeed, since they do not have a stake in the meeting discussions, they can even perform their listening and comprehending activities entirely in the service of the note-taking task. As such, one may conclude that participants of the user study are *less* in need of note-taking assistance because they can better focus their attention on the note-taking task than typical meeting participants. On the other hand, study participants have several disadvantages as compared to meeting participants. Despite the availability of scaffolding text, it is likely that study participants are less familiar with the contents of the recorded meetings than typical meeting participants. Further, meeting participants have access to many non-verbal cues in face to face meetings that the participants of our user study do not have access to. Despite these differences, we will show in the results section that, for the chosen meeting segments, the user study participants' notes compare well with the notes that the original participants of these recorded meetings took.

7.4 Data for User Study

As mentioned above, we selected 4 recorded meeting segments for participants of the user study to listen to and take notes on. A numerical description of the chosen segments is provided in Table 7.1. These segments were selected from the meeting dataset presented earlier. Each segment was chosen to be of roughly equal length – about 5 minutes long. Thus, none of the chosen segments spanned the entire length of any of the meetings in the dataset (the shortest of which is 20 minutes long). Care was taken, however, to ensure that each segment contained a coherent set of topics.

Observe from the table that, although the meetings were of similar lengths of time, the number of utterances in these meetings varied. This variance is merely a reflection of the number of pauses speakers took, and how often the speaking turn changed between participants. The 4^{th} column shows the number of speakers in each meeting segment. Note that some segments had more participants who simply did not speak during the chosen 5-minute segment.

The 5^{th} column in Table 7.1 presents the number of suggestions shown to study participants in the Wizard condition in each of the 4 meetings. (Of course every participant saw suggestions from the Wizard in exactly *one* of the 4 meetings, and saw suggestions from other sources or no suggestions at all in the other 3 meetings). As mentioned above, these suggestions consist of manual transcripts of utterances annotated as "Definitely Show" by both human annotators. Note that the meeting segments were deliberately chosen to have a high number of such utterances. 5-minute contiguous meeting segments in the full meeting dataset have 9.6 "Definitely Show" utterances on average (standard deviation 6.8, range 0 to 32), whereas the meeting segments used in the study have an average of 21.3 such utterances. A notes-assistance system is unlikely to be useful when meetings have little or no noteworthy information. Using the number of utterances annotated as "Definitely Show" as a metric for the density of noteworthy information in a segment, we chose segments with relatively high density to avoid the possibility that the notes-assistance system fails due to an insufficient amount of important information in the segment.

The 6^{th} column in the table presents the number of suggestions in the two Implict conditions for the 4 meetings. As was done for the offline evaluations in previous chapters, the noteworthy-utterance detector was trained on labeled data automatically extracted from all meetings in the dataset *except* the meeting containing each chosen meeting segment. Labeled data was extracted from the notes written by the original participants of the meetings in the dataset, and not from the notes taken by the user study participants. A Support Vector Machines-based noteworthy utterance detector was trained on this data, as described in the previous chapter. This trained detector was then used to classify (manually transcribed) utterances in the chosen meeting segments as either "noteworthy" or "not noteworthy". Utterances identified as "noteworthy" by the classifier were then used as suggestions. For the "Implicit (manual)" condition, the manual transcripts of these utterances were shown to the study participants as suggestions, while for the "Implicit (ASR)" condition the automatically generated transcripts of these utterances were shown. Recall that the speech recognizer had an overall word error rate of 40% for this dataset.

Thus, for all 3 conditions in all 4 meetings, the utterances to be shown as suggestions were identified before the start of the experiments. During the experiment, as the study participant listened to a particular meeting segment, each suggestion from a particular chosen condition was shown to the participant 3 seconds after the audio of the corresponding utterance was played. This time-lag was designed to mimic the typical time needed by a live notes-assistance system to process an utterance.

The last 3 columns of Table 7.1 represent the offline performance evaluation of the implicitly trained noteworthy-utterance detector for each meeting segment, as evaluated against manually annotated "Definitely Show" utterances. Since utterances predicted as "noteworthy" by the detector were used as suggestions in the two Implicit conditions, and utterances labeled "Definitely Show" were used as suggestions for the Wizard condition, these numbers are effectively the same as computing Precision, Recall and F-measure between suggestions in the Implicit and Wizard conditions for each meeting. Although the performance of the detector varied on all 3 metrics across the 4 meetings, the meetings were chosen so that the average F-measure value across the meetings – 0.24 – is close to the average performance value of the detector on the full meeting dataset. Our goal in doing so was to ensure that, in aggregate, we evaluate the usefulness of a detector whose accuracy we have already achieved, rather than one with an artificially high accuracy that may or may not be achievable in a real system.

7.5 How the Experiment was Conducted

In this section we present a detailed description of how the user study was conducted for each participant. In order to ensure that all participants received identical instructions, we created a "script" that was adhered to as instructions were given to the participants. This script is presented in Appendix A.

7.5.1 Inviting Participants and Physical Set-up

The experiment was conducted in two phases. During the first "pilot" phase, a total of 10 friends and colleagues of the author were invited to participate in the experiment, and the various aspects of the study were fine-tuned based on their feedback. The data from these participants are not presented in this thesis.

During the second phase of the experiment, 24 new participans were recruited to take part in the experiment, and the study was conducted unchanged for all these participants. To recruit these participants, an announcement briefly describing the experiment and inviting readers to participate in it was posted on an experiment scheduling website maintained by the Center for Behavioral Decision Research at Carnegie Mellon University. People of varying demographics (described in more detail in Section 7.5.2) signed up and were scheduled to participate in the experiment. This phase of the experiment was conducted over 11 days, and on each day 1 to 4 participants performed the experiment in individual, non-overlapping time slots. These slots were all between 9 am and 7 pm, and were 2 hours long each, although no participant took more than 1.5 hours to complete the entire experiment (including instructions, demonstrations, etc.)

All experiment sessions were conducted in the same laboratory at the Carnegie Mellon University campus. Directions to this laboratory were given to the participants during the sign-up process, and appropriate signage was placed near the room to help participants find it quickly and easily. During the sign-up process, prospective participants were encouraged to bring their own headphones or earphones in order to eliminate a source of unfamiliarity. A small number of participants did not bring their own headphones; they were supplied a pair of circumaural headphones. All participants performed the entire experiment on the same desktop computer with a 21-inch Liquid Crystal Display monitor, and a standard keyboard and mouse. To minimize distractions, the participant sat facing a wall, the experimenter sat 10 feet away, the laboratory had no other person besides these two, and the laboratory door remained shut for the entirety of the session.

7.5.2 Explaining the Consent Form and the Demographic Survey

Once a participant arrived for her scheduled time slot, she was given a very brief overview of the experiment. She was told that she would have to take notes while listening to 4 recorded meetings, fill out a questionnaire after each meeting, and complete two additional survey questionnaires. The consent form was then briefly explained to the participant. She was advised that her notes and her answers to the questionnaires would be recorded and used for research, but that all the data would be collected anonymously and stored with an ID number. Once the participant indicated that she had understood her rights and obligations, she was asked to read the consent form, ask any additional questions she may have, and finally sign the form. This full process typically took about 5 minutes.

Once the consent form was signed, the participant was asked to complete the initial pre-experiment survey. As mentioned earlier, the goal of this questionnaire was to gather demographic information about the participant, and information about the participant's note-taking habits in meetings. This questionnaire was presented through a Web browser on the same computer on which the experiment was later conducted. Participants generally took between 10 and 20 minutes to complete this questionnaire.

7.5.3 Demonstrating the Software and the Experiment

The next step was to demonstrate the SmartNotes note-taking software to the participant. To perform such demonstrations, a separate 1-minute "Demo Meeting" was created, complete with note-suggestions, and used for all participants, as follows. First, using this meeting, the experimenter demonstrated to the participant how to start the software for the meetings. He explained the meaning of each of the panels of the software – the Note Suggestion panel, the Scaffolding Text panel and the Note-Taking panel. (A screen-shot of the SmartNotes interface showing these different panels is shown in Figure 7.1.) He gave her her username and password, and showed her how to log into the SmartNotes system, how to take notes within the Note-Taking panel, and how to use the suggestions in the Note Suggestion panel. It was explained to her that she did not need to save her notes for each meeting – they were being automatically sent to the server and saved. She was also advised that although she could continue editing her notes after the meeting playback had ended, she should refrain from editing her notes for too long afterwards. Finally she was shown how to fill out the Web-based questionnaire after each meeting.

The actual experiment (repeatedly listening to meetings, taking notes, and answering questions in the post-condition questionnaires) was designed to run automatically, with pop-up instructions asking the participant to open a particular meeting, start taking notes, fill out a survey, etc. The full experiment was demonstrated to the participant using the "Demo Meeting". This demonstration was repeated twice. During the first time, the experimenter guided and instructed the participant at every step. During the second time, the participant was asked to do the experiment (using the same "Demo Meeting") on her own, with the experimenter sitting next to her but not helping her if she didn't need help. The goal was to familiarize the participants with the mechanics of doing the experiment before they started on the actual experiment.

7.5.4 Final Instructions Regarding Note-Taking and Suggestion-Usage

Once the demonstration was over and the participant felt she could do the experiment on her own, she was given some final instructions to ground her note-taking work. Specifically she was instructed to take notes such that a manager or a student-advisor who has a stake in the project being discussed in the meetings, but who wasn't able to attend the meeting, can get an overview of the meeting discussions. Towards this end, the participant was given a few simple examples of noteworthy information – action items, deadlines, work completed since the previous meeting, new and unexpected problems, etc. She was also cautioned that these were merely examples, and that these meetings were likely to contain other pieces of noteworthy information that did not fit these simple categories. The participant was advised to bring her own sense of what qualifies as noteworthy information to bear on this note-taking task.

With regard to the suggestions, the participant was told that in 3 of the 4 meetings, the system would make note-suggestions that they were free to use to create their notes as had been demonstrated a little earlier. It was clearly explained to the participant that she was under no obligation to use them. Instead, her goal was to create "good notes", with the judgment of goodness being left to her. The experimenter explained to the participant that suggestions would vary widely in quality between and within each of the 3 meetings that would have suggestions, and that she needed to decide for herself whether using the suggestions would help her note-taking or not. The aim of these instructions was to prevent the demonstration of how to use suggestions from being an implicit instruction to actually *use* them. Instead the goal was to clarify to the participants that their primary task was note-taking, and that they should use the suggestions only if they felt it would help them in taking notes.

Finally, the participant was cautioned about a few unusual aspects of the meetings. She was told to expect a few instances during the 4 meetings where participants would talk about writing pieces of information down in their notes. For example, a speaker might say "Why don't I take that down as an action item for you?" The participant was asked to assume that the original notes that these participants wrote are now lost, and that she should write down the pieces of information being discussed *if* she too considered them noteworthy. She was told to expect that the suggestions, if they appear, appear with a slight delay after the utterance has been completed. She was also cautioned that on rare occassions what seems to be a single utterance might be broken up into multiple consecutive suggestions. For example the utterance "I will do that task tomorrow" might get broken down into two suggestions: "I will do" and "that task tomorrow". (This situation happened when the speaker paused in the middle of a sentence, between the words "do" and "that" in this example. Although semantically the two portions form a single sentence, the silence-based end-pointer splits them into two utterances).

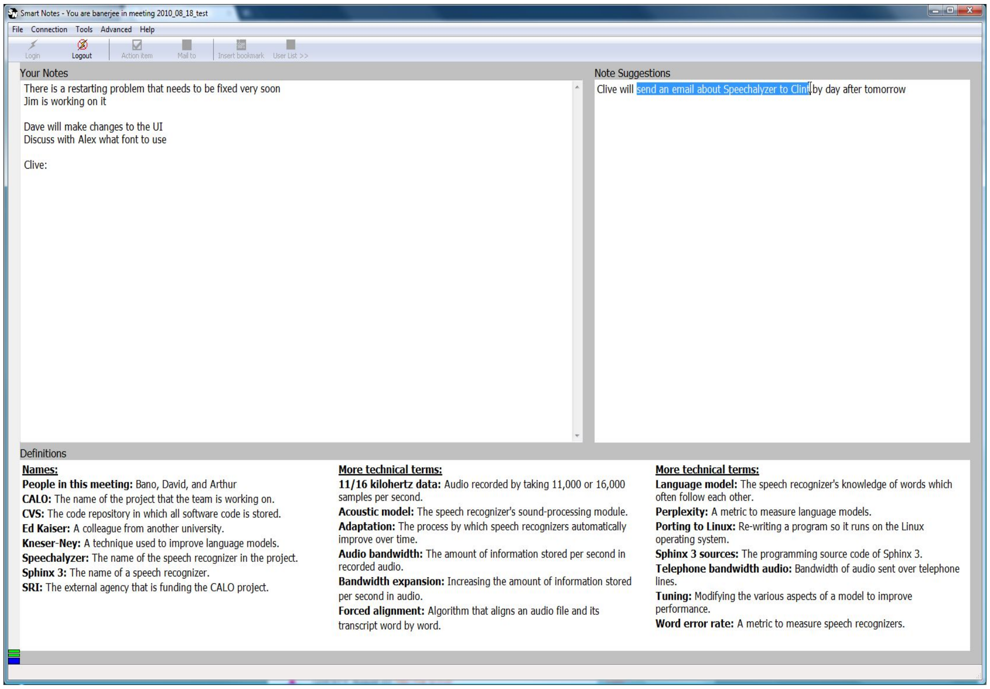
These instructions and demonstrations took approximately 20 to 25 minutes. Once the participant indicated that she had understood these instructions, she was asked to start the experiment.

7.5.5 Running the Experiment

As mentioned earlier, once the experiment started, it was entirely automatic. The experiment software randomly chose one of the condition orders that had not been used for any participant as yet. The participant listened to each meeting and took notes. For the Implicit and Wizard conditions, the experiment software displayed suggestions to the participant at appropriate times synchronized with the audio playback. At the end of each meeting, the participant was shown the post-condition questionnaire through a Web browser. At the end of the 4 meetings the participant was shown the final post-experiment questionnaire. This experiment process took between 30 and 40 minutes to complete.

7.5.6 Concluding the Experiment

Once the experiment was completed, the participant was paid $15, and was asked to sign the receipt sheet. The participant was then informed that the experiment session had concluded and was thanked for her time.



There is a restarting problem that needs to be fixed very soon
Jim is working on it

Dave will make changes to the UI
Discuss with Alex what font to use

Clive:

Clive will send an email about Speechalyzer to Clint by day after tomorrow

Names:
People in this meeting: Bano, David, and Arthur
CALO: The name of the project that the team is working on.
CVS: The code repository in which all software code is stored.
Ed Kaiser: A colleague from another university.
Kneser-Ney: A technique used to improve language models.
Speechalyzer: The name of the speech recognizer in the project.
Sphinx 3: The name of a speech recognizer.
SRI: The external agency that is funding the CALO project.

More technical terms:
11/16 kilohertz data: Audio recorded by taking 11,000 or 16,000 samples per second.
Acoustic model: The speech recognizer's sound-processing module.
Adaptation: The process by which speech recognizers automatically improve over time.
Audio bandwidth: The amount of information stored per second in recorded audio.
Bandwidth expansion: Increasing the amount of information stored per second in audio.
Forced alignment: Algorithm that aligns an audio file and its transcript word by word.

More technical terms:
Language model: The speech recognizer's knowledge of words which often follow each other.
Perplexity: A metric to measure language models.
Porting to Linux: Re-writing a program so it runs on the Linux operating system.
Sphinx 3 sources: The programming source code of Sphinx 3.
Telephone bandwidth audio: Bandwidth of audio sent over telephone lines.
Tuning: Modifying the various aspects of a model to improve performance.
Word error rate: A metric to measure speech recognizers.

Figure 7.1: Screen-shot of SmartNotes client showing the participant selecting a portion of a suggestion for inclusion in his notes. This screen-shot also shows the scaffolding provided to each participant – short definitions of names and technical terms discussed during the meeting.

7.6 Modified SmartNotes Interface

Two modifications were introduced to the SmartNotes interface for the purpose of this user study; figure 7.1 shows this modified interface. The first change is in the Note Suggestion panel. The note-suggestion-acceptance interface designed and implemented in previous chapters (and tested during the previous WoZ study), allowed participants to insert suggestions into their notes with a single click. When participants saw a suggestion they wished to include in their notes, a single click anywhere on the text of the suggestion would include *all* of the text of that particular suggestion into the notes. Participants could then edit this inserted suggestion as they saw fit.

This method is a quick way of inserting all the text of a suggestion into the notes, and is very useful when the entire suggestion is important. However, when only a portion of a suggestion needs to be inserted into the notes, the participant must insert all the text of the suggestion and then delete the unimportant portions. This is a time-consuming task, and may drastically reduce the time-saving benefit of note-suggestions. Indeed, one of the recurrent feature requests made by the 10 participants of the pre-experiment “pilot”-phase of the study was to enable selection and insertion of *portions* of suggestions. To this end, we implemented a “Sub-Suggestion Selection” interface and enabled it for all 24 participants of the main experiment.

In this selection mechanism, if participants saw a contiguous phrase in a particular suggestion that they wished to insert into their notes they could take the following steps:

- Select the phrase by clicking-and-dragging the mouse pointer, but without releasing the mouse button.
- Insert the selected phrase into the notes by simply releasing the mouse button.
- If the participant felt that he had selected and inserted text in error, he could undo the action by pressing control-Z on the keyboard.

Figure 7.1 shows such a sub-suggestion selection-and-insertion in progress. By combining the selection and insertion tasks into one click-drag-release action, the time taken to do the full action is reduced, as compared to separating the selection and insertion tasks into two separate actions. In addition to this suggestion insertion method, participants also retained the option of inserting all the text of a suggestion by clicking on it.

In the second modification, we replace the “Personal Notes” panel with a “Scaffolding Text” panel, labeled as “Definitions” in Figure 7.1. As described above, scaffolding texts include short descriptions and definitions of the names and technical terms used in each meeting; these were provided to the user study participants in the panel labeled “Definitions”. At the beginning of each meeting, this panel was refreshed and the pre-set list of definitions for that particular meeting alone were shown. For each meeting, the exact same text was shown to all participants, regardless of the condition being administered to a particular participant during that meeting. Note that participants could not select and insert text from this panel into their notes; both the click-and-insert and the select-and-insert mechanisms were restricted to text in the note-suggestions panel only.

7.7 Participant Data

As mentioned above, there were 24 participants in the main experiment and each participant was exposed to a unique condition order in order to counter any learning effect. Before they started with the main experiment, participants were requested to provide demographical information through a questionnaire. The aggregate demographic information is as follows:

The participant population was overwhelmingly comprised of young students from local colleges, and had more females than males. Specifically, 19 of the 24 participants were students, 1 was unemployed, and the stated professions of the remaining 4: *manager*, *admin assistant*, *processor*, and *child care provider*. Although the participants had a wide range of ages – from 21 to 57 – the median age was only 25. There were 14 female participants and 10 male participants.

In addition to getting these pieces of demographic information, we wished to establish whether participants attended meetings often, and whether they took notes regularly or only occasionally at those meetings. Instead of asking participants to provide this information in generalities (e.g. "how many meetings do you attend in a week in general?") participants were asked to ground their answers in specifics by thinking of the most recent week during which they went to at least one meeting. For that week, participants were asked to report (a) the number of meetings they attended that week, and (b) how many of those meetings they took notes at. Participants were also asked to specify how long ago this specific week occurred. 17 participants answered these questions based on the current or previous week, and the remaining participants used a week no more than 4 weeks prior to the current week. Participants reported going to between 1 and 10 meetings in the week in question, with a median of 2 meetings. 9 participants took no notes in any of their meetings. Across all the 24 participants, the median number of meetings participants took notes in was 1. This shows that the participants in our study were not very frequent meeting attendees. However, 15 of the 24 participants reported taking notes in at least one of the meetings they went to recently. Thus note-taking was a familiar task with a majority of the participants of this study.

Finally, participants were asked to rate themselves on their note-taking skills. Participants were asked to choose one of the following 5 options: "Really Bad", "Bad", "Moderate", "Good", "Really good". These options were internally assigned to values 0 through 4 (in the same order). On average participants reported their skill level as being 2.5, i.e. between "Moderate" and "Good". (A more detailed explanation of how answer options were provided to participants is given in Section 7.11.)

7.8 Results: Suggestion Acceptance

In this section we present the first part of the results from the user study: participants' suggestion acceptance behavior. In the next sections we analyze the effect the suggestions had on the quality of participants' notes and on their note-taking effort. In the following section, we analyze the answers participants gave to the various questionnaires. As done in previous chapters, all statistical significance is computed using two sample bootstrap.

7.8.1 Goal: What to Measure

As mentioned in the Goals section (Section 7.2), the first question we wish to analyze is whether study participants were willing to accept note-suggestions shown to them during the meetings. Recall that for each participant, suggestions were shown from 3 different sources, all suggestions from a single source being shown in a single meeting randomly chosen from the 4 meetings of the experiment. These 3 sources formed separate experimental conditions and were named Wizard, Implicit (manual) and Implicit (ASR). (No note suggestions were shown in the remaining 4^{th} meeting, which was designated as the control condition.) In the Wizard condition, participants were shown the manual transcripts of utterances in the meetings that were labeled as "Definitely Show" by all human annotators. In the Implicit (manual) condition, participants were shown manually transcribed utterances that were classified as noteworthy by the automatically trained noteworthy-utterance detector. Finally in the Implicit (ASR) condition, participants were shown the automatic transcriptions of the utterances shown in the Implicit (manual) condition. These conditions are described in more detail in the "Design of User Study" section (Section 7.3).

Recall that for this experiment, participants were allowed to accept individual suggestions either wholly or partially, by clicking on the text of the chosen suggestion or by selecting a portion of the suggestion respectively. To measure the degree to which study participants used the note-suggestions shown to them, we calculate the two following values:

- The number and percentage of suggested utterances that were wholly or partially accepted
- The number and percentage of suggested words that were wholly or partially accepted

While the first value captures the number of suggestions that contained at least some information that was of value to the participant, the second value represents the usefulness of the suggestions at the finer word-level granularity.

7.8.2 Overview of Results

Tables 7.2 and 7.3 presents these suggestion acceptance results, while Figure 7.2 shows a graphical representation of the results (error bars are based on standard error). The three major columns of the table represent the suggestion acceptance results from the three conditions in which suggestions were shown to the participants. (Since there were no suggestions or suggestion acceptances in the control condition, that condition is not represented in this table). For each condition, the table shows the average and the standard deviation values for the number ("#") and the percentage ("%") of suggested utterances

Counting Acceptances of	**Wizard**				**Implicit (manual)**			
	#		%		#		%	
	Avg	Std	Avg	Std	Avg	Std	Avg	Std
Utterances	7.7	5.0	**36.6**	21.5	2.4	1.5	**34.5**	21.8
Words	61.4	41.4	32.8	21.5	24.3	19.3	28.1	22.2

Table 7.2: Average and standard deviation of the number and percentage of suggested utterances and words that were accepted by participants in the three conditions in which suggestions were shown (part 1 of 2, continued in table 7.3).

Counting Acceptances of	**Implicit (ASR)**			
	#		%	
	Avg	Std	Avg	Std
Utterances	1.3	1.3	**19.1**	19.2
Words	11.8	16.7	13.5	18.6

Table 7.3: Average and standard deviation of the number and percentage of suggested utterances and words that were accepted by participants in the three conditions in which suggestions were shown (part 2 of 2, continued from table 7.2).

or words accepted during that condition, averaged over all study participants. Since all 3 conditions were shown to every participant, these average values are computed over 24 different instances. In addition, since the order of the conditions was counter-balanced but the order of the meetings was kept constant, each condition-meeting pair occurred exactly 6 times.

7.8.3 Percentage of Suggested Utterances Accepted in the Wizard Condition

The first row of results in Tables 7.2 and 7.3 shows the number and percentage of suggested utterances that were accepted, either wholly or partially. Observe that the average acceptance percentage in the Wizard condition, 36.6%, is close to the 30% acceptance percentage in the earlier Wizard of Oz study presented in Chapter 6.3. Thus for this metric – the percentage of suggested utterances accepted by participants – the performance of the Wizards in these two experiments appear to be similar, even though the experiments involved different wizards, meetings, topics, participants, and meeting styles. This performance similarity may point to an underlying limitation of extractive summarization-based notes-assistance.

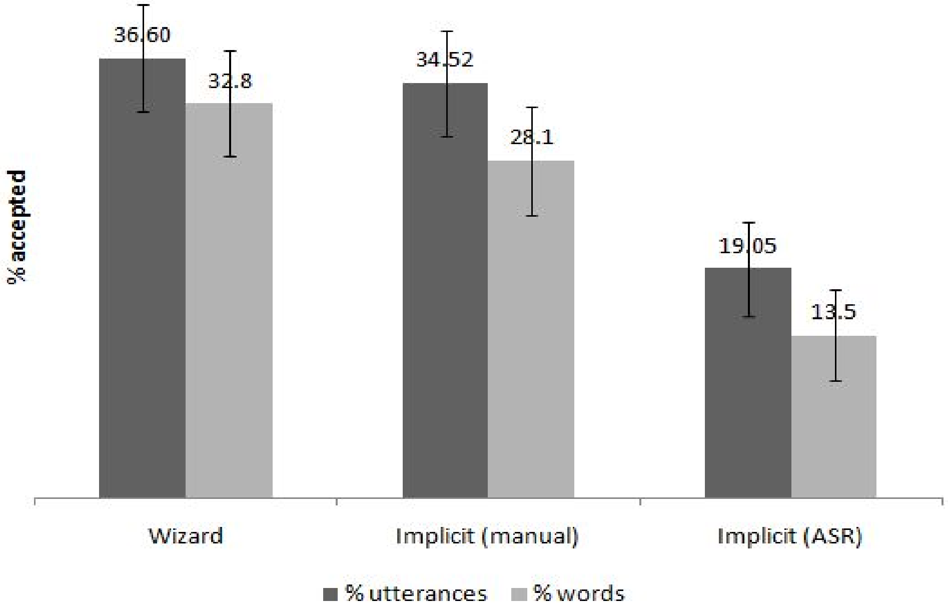


Figure 7.2: Percentage of suggestions accepted in each of the conditions where suggestions were made.

7.8.4 Percentage and Number of Suggested Utterances Accepted across Conditions

The average percentage of suggested utterances accepted in the Wizard and the Implicit (manual) conditions (36.6% and 34.5%) are not statistically different from each other. Thus, participants find approximately similar fractions of suggested utterances useful, whether the suggestions are made by the Wizard or by the implicitly trained noteworthiness detector – when using manual transcriptions. When using automatic transciptions however (the Implicit (ASR) condition), the fraction of utterances accepted by study participants drops to only 19.1%, a nearly 45% drop from the Implicit (manual) condition. The differences in the percentage of accepted suggestions between the Implicit (ASR) condition and the Wizard and Implicit (ASR) conditions are both statistically significant ($p < 0.01$). Recall that the exact same utterances are suggested to the participants in these two conditions; the only difference is in the transcription quality (perfect transcription in the Implicit (manual) condition versus 40% word error rate in the Implicit (ASR) condition). This shows the large negative impact of transcription errors on the usefulness of the suggestions.

Although the percentage of accepted utterances is similar between the Wizard and Implicit (manual) conditions, observe that the absolute numbers have a large difference – 7.7 versus 2.4 in the two conditions respectively. This is due to the fact that different numbers of utterances are suggested in these two conditions, as shown in table 7.1.

7.8.5 Correlation between a Suggestion's Word Error Rate and its Acceptance Rate

As noted above, suggestion acceptance is much lower in the Implicit (ASR) condition than in the Implicit (manual) condition – 19.1% versus 34.5%. Since the only difference between the suggestions in the two conditions is the presence of transcription errors in the Implicit (ASR) condition, this drop in acceptance rate can be attributed to the errors. To further explore the relationship between the suggestions' error rates and their acceptability, we perform the following analysis.

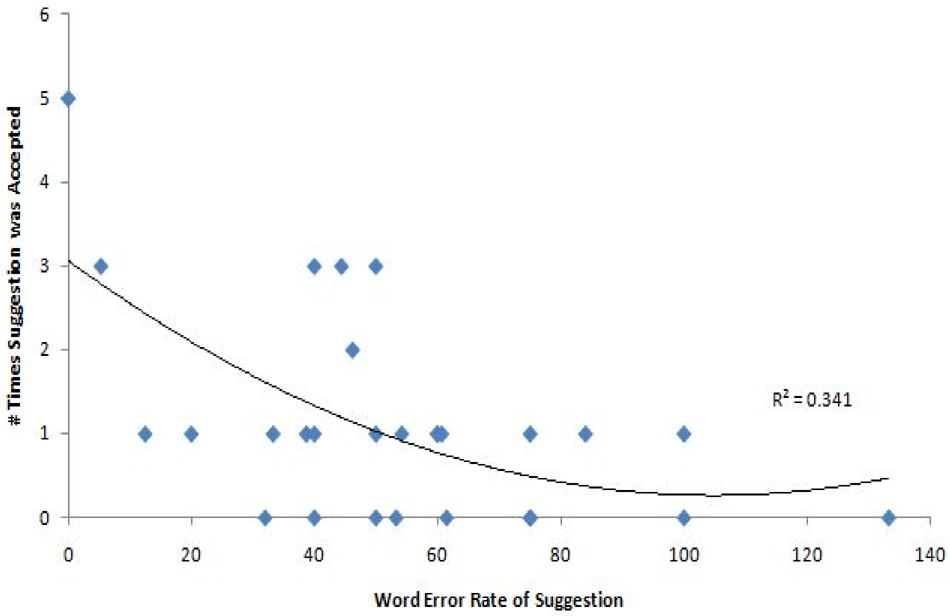


Figure 7.3: The number of times each suggestion was accepted (in Implicit (ASR) conditions), versus its Word Error Rate

Recall that there were 7 suggestions in the Implicit (Manual) condition of each of the 4 meetings used in the experiment, and that the ASR transcriptions of the same suggestions were shown in the Implicit (ASR) condition for those meetings. Recall too that each suggestion in each condition-meeting pair was presented to 6 different participants. Thus each Implicit (ASR) suggestion could have been accepted between 0 and 6 times. Further the Word Error Rate of each suggestion in the Implicit (ASR) condition can be computed by comparing it to the corresponding manually transcribed suggestion in the Implicit (Manual) condition of the same meeting. Figure 7.3 plots for each suggestion (across the 4 meetings) the number of times it was accepted (between 0 and 6, along the Y-axis), against its Word Error Rate (between 0 and 133.3%, along the X-axis). In addition, to summarize the relationship between the two variables (acceptance rante and word error rate), we fit a second-order polynomial trend-line to the data-points, as also shown in the figure.

Observe the general shape of the fitted polynomial trendline. At low word error rates, it indicates a high likelihood of acceptance, whereas for higher word error rates, this likelihood reduces, and nearly reaches 0. This graph supports the intuition that acceptability reduces as word error rates rise. According to the equation[1] of the fitted polynomial, the likelihood of at least one study participant accepting the suggestion drops below 1 at a Word Error Rate of 56.26%. Note that as shown in the graph, the R^2 value of the fitted polynomial is 0.341. As a sanity-check comparison, the R^2 value of a similar graph between the acceptance rates of manually transcribed utterances and the Word Error Rates of the corresponding ASR-transcribed utterances is only 0.035, indicating a lack of correlation, as is to be expected.

7.8.6 Percentage and Number of Words Accepted

These trends are repeated in the number and percentage of suggested words accepted by the study participants in the three conditions (last row of Tables 7.2 and 7.3). The difference between the percentage of suggested words accepted in the Wizard and Implicit (manual) conditions (32.8% versus 28.1%), although larger than the difference in the percentage of utterances accepted between the two conditions, is not statistically significant, whereas the differences between these two conditions and the Implicit (ASR) condition are both statistically significant. Yet again, the differences between the Wizard and Implicit (manual) conditions in the *number* of words accepted is large due to the larger number of words suggested in the Wizard condition to begin with. These word-level acceptance results show that study participants were indeed willing to accept and use nearly a third of the suggested words shown by the automatically trained noteworthy utterance detector in order to compose their notes.

[1] Equation of fitted polynomial: $y = 0.0003x^2 - 0.0525x + 3.0603$

7.8.7 Conclusion

Based on the results in this subsection, we can answer the question “will humans accept suggestions made by an implicitly trained noteworthiness detector?” in the affirmative. We have shown that when transcription accuracy is high, the percentage of suggested utterances and words accepted by study participants is not significantly different between the Wizard and Implicit conditions. The acceptance numbers and percentages fall when the utterances are transcribed using low accuracies, as is expected.

7.9 Results: Note-Taking Effort

The second question we wish to explore, as mentioned in the Goals section (Section 7.2), is what impact the automatically generated suggestions have on the *effort* required to create the notes. In this section, we present a metric for measuring note-taking effort, and present analyses thereof.

7.9.1 Measuring Note-Taking Effort

Our goal is to measure the effort it took study participants to write notes in the various conditions. We do so by calculating the percentage of the notes that participants typed manually, as opposed to inserting from the suggestions. In the control condition, for example, where participants write all the notes on their own, we rate the note-taking effort at 100%. On the other hand, if a participant manually typed 60% of his notes in a particular meeting, and created the remaining 40% by inserting text from the suggestions, we rate the note-taking effort for that meeting at 60%. We compute these percentages at the level of characters, rather than at the level of words, sentences or lines.

Note that for the sake of simplification, we do not track the edits that a participant may make after inserting text from the suggestions. For example, a participant may insert some text from the suggestions into the notes, and then cut and paste a portion of the text somewhere else in the notes. We do not track this behavior, and indeed we count the inserted piece of text only once for the numerator of the effort metric. Since study participants had the ability to select only those portions of the suggestions that they wanted they had less need to do post-selection edits than if they were forced to select the entire suggestion every time. Also, since they were under constant time pressure, we believe very few edits were done to the text inserted from the suggestions.

This metric only measures the visible portion of the entire note-taking effort. Note-taking is comprised of many sub-activities: understanding the discussion, judging the relative importance of different pieces of information, mentally composing the text of the note to

Condition	# Chars in Notes	# Manual Chars	% Effort
Control	480.2	480.2	100.0
Wizard	646.1	325.0	50.3
Implicit (manual)	540.5	398.9	73.8
Implicit (ASR)	480.0	407.5	84.9

Table 7.4: The effort expended in taking notes in each condition, averaged across all user study participants.

be written, and then ultimately writing down the note. When using the note-suggestions, participants must perform the additional tasks of reading the suggestions, deciding whether to use a suggestion, and if so whether to use the entire suggestion or a part thereof, etc. Of all these activities only keyboard events (typing the notes) and mouse events (accepting suggestions partially or wholly) are visible. Since it is not easy to combine mouse and keyboard events into a single metric, and since typing makes up the majority of the visible effort in note-taking, we use percentage of characters manually typed as an approximation of note-taking effort.

7.9.2 Results

The effort results are presented in Table 7.4. The column labeled "# Chars in Notes" shows the length of the final notes in each condition in characters (averaged over the 24 participants). Observe that the size of the notes varies exactly as the raw number of suggestions accepted in each condition, as shown in Tables 7.2 and 7.3. That is, when good suggestions are available, participants create longer notes than if they are left to create the entire notes on their own (or with poor suggestions as in the Implicit (ASR) condition). This is an intended effect of notes-suggestion – that they enable participants to include more information in their notes than they would have been able to if they had to write the notes on their own.

The second and third columns, labeled "# Manual Chars" and "% Effort" respectively, represent the number and fraction of characters in the final notes that were manually typed. Observe that in the Wizard condition, participants wrote only about 50% of the final set of notes on their own, creating the rest by selecting text from the suggestions. Since participants were able to select fewer suggestions from the Implicit (manual) predictions, they had to type more of the notes on their own leading to a higher effort of 73.8%; this effort is statistically significantly higher than that needed in the Wizard condition, and also statistically significantly lower than that needed in the Control condition ($p < 0.01$). It is marginally lower than the efforted needed in the Implicit (ASR) condition ($p < 0.05$). For the Implicit (ASR) condition, observe that although the average length of the notes was the same as for the Control condition, some portion of the notes were selected from the text, thus saving some typing effort. This effort difference from the Control condition was also statistically significant ($p < 0.01$).

These results imply that as suggestions improve, participants are simultaneously able to create longer notes while reducing their own note-taking effort. In the next section we evaluate the quality of the notes thus created.

7.10 Results: Note Quality

In the previous section we observed the effect that note-suggestions have on note-taking effort. In this section we will present an evaluation of the *quality* of the notes created by the study participants in various conditions, and compare them to some benchmark evaluation scores.

7.10.1 Measuring Note Quality

As mentioned previously, creating an absolute evaluation metric of note quality is a difficult task in general, due to the inherent subjectivity of note-taking: for most meetings, different participants of the same meeting are likely to favor different kinds of notes. This variability arises mainly from the differences in meeting participants' note-taking goals. For example, different meeting participants may have dissimilar levels of interest in the various topics being discussed at a meeting, and the set of notes they *aim* to write may be consequently different (regardless of what notes they *actually* write, which is influenced by their note-taking capabilities). Orthogonally, different participants may want to put the notes to different use – some may want to use them as a summary of the meeting, some as a list of tasks they need to perform, etc. – leading to a variety of opinions about the goodness of a particular set of notes.

Much of this variability is absent for the participants of our user study. Since none of them were actual participants in the meetings, and since none of them were going to put their notes to any use after the experiment, they all had the same note-taking goal – that is, the goal given to them as a part of the experiment instructions. Specifically, recall that study participants were told to take notes "such that someone who wasn't at the meeting, particularly a manager or a student advisor, can get an overview of the important information discussed at the meeting" (more details in Section 7.5.4). Given that all participants had the exact same motivation for note-taking, we assert the existence of "gold standard notes" that each participant should aspire to. This assertion is similar to that used in summarization evaluation. Although the quality of a summary can depend on its ultimate use, in the absence of a use-case, "reference summaries" are created, and the evaluation paradigm assumes that all summaries should aspire to those references.

For summarization evaluation, domain experts are asked to create these "reference summaries". Similarly, we constructed "gold standard notes" by recruiting two of the original participants of these meeting segments; we refer to them as "experts". Being participants of the original meetings (and the projects underlying them), these experts have maximal knoweldge of the topics of discussion. We asked these two experts to independently listen to these meeting segments and create notes, with the same note-taking goal as that given to the study participants. Unlike the study participants however, we allowed these experts to pause, rewind and fast-forward the audio as often as they needed, and in general to take as much time as necessary to construct a set of notes that they would consider "good". Based on these factors – the maximal domain knowledge of the notes' authors, the lack of any time constraints, and the lack of distractions from other speakers – we designate these notes as the "gold standard" for the specific note-taking motivation discussed above. We then evaluate the notes created by each participant for each meeting by computing the ROUGE F1 metric between those notes and these two gold standard notes, and reporting the average of these two values. Recall that ROUGE is an n-gram matching-based metric, and consists of three values – precision, recall, and Fmeasure. We have experimented with both unigram and bigram matching to compute ROUGE, but similar to our offline system evaluations, we have observed that both sets of results follow similar trends. We present unigram-matching based ROUGE scores in the next section.

An alternate technique for evaluating summaries is the Pyramid Method [34]. This method requires manual annotation of Summarization Content Units, whereas the ROUGE method selected above, while potentially less sensitive than the Pyramid Method, does not require such manual annotation of the gold summaries. In addition, using ROUGE gives us parity between the evaluations presented in this chapter and those done in previous ones.

7.10.2 Per-Condition Results

The ROUGE-based results of the note-quality analyses are presented in Table 7.5, along with several other benchmarks for comparison purposes. Portion (a) of the table presents the average ROUGE precision, recall and Fmeasure values for the notes written by the study participants in each of the 4 conditions averaged across all the meetings. By averaging across all meeting-condition pairs, we control for individual differences between meetings. By averaging across all condition orders, we control for any learning effects participants may carry from one condition to the next. Finally by using a within-subjects experiment design, we control for study participants' individual note-taking styles and differences.

Observe that as expected, notes taken in the Wizard condition, where note-suggestions consisted of utterances manually annotated as "Definitely Show", have the highest Fmeasure score among the 4 conditions. In fact this Fmeasure score is statistically significantly higher than those obtained in the remaining 3 conditions. This shows that when study participants were assisted in their note taking by human-curated suggestions, their notes were closest to that of experts.

Among the other conditions, observe that both Implicit (manual) and Implicit (ASR) conditions get slightly higher Fmeasure on average than the Control condition (in which participants did not have access to note-suggestions). However, these differences are not statistically significant, implying that having access to suggestions from the automatically trained noteworthiness predicter does not result in big improvements in the quality of the notes, when averaged across all the meetings.

This result varies from meeting to meeting however. Tables 7.6 and 7.7 disaggregate these results by both condition and meeting. Observe that for meetings 1 and 2, there is very little difference between the Fmeasure scores of the Wizard and Implicit (manual) conditions; in fact these differences are not statistically significant. On the other hand, in the remaining two meetings, the Fmeasure of the notes in the Wizard condition is statistically significantly higher than that of the notes in the Implicit (manual) condition. Note that these meeting-to-meeting differences in the usefulness of the automatically generated suggestions do not follow the same trend as the differences in the offline performance of the noteworthiness detector as shown in Table 7.1. This suggests that the offline performance of the noteworthiness detector is an insensitive indicator of the usefulness that meeting participants may derive out of note-suggestions based on the output of such a detector.

Returning to the results disaggregated only by condition (portion (a) of Table 7.5), observe that the differences in the fmeasure values across the 4 conditions comes from the differences in the recall scores, since the precision levels do not differ by much. These differences in recall value probably arise from the differences in the amount of information accepted from the suggestions in the different conditions, and also in the sizes of the notes themselves. As seen in the suggestion acceptances tables (Tables 7.2 and 7.3), the sorted order of the conditions if they were ranked on the absolute number of words accepted from suggestions during those conditions (from the fewest acceptances – Control – to the most acceptances – Wizard) matches exactly the sorted order of the conditions if they were ranked on the sizes of the notes (Table 7.4), and also if they were sorted on the ROUGE recall value of the notes taken during those conditions. As study participants select more and more suggestions, the size of their notes increases, and the recall value of their notes improves. However, they only add those pieces of text that, in their judgment, truly contain important information; as a result the notes do not suffer a dip in precision value, as might be expected if text was added indiscriminately.

(a) **User Study Participants' Notes**	P	R	F
Control	0.65	0.24	0.34
Wizard	0.66	0.35	0.45
Implicit (manual)	0.65	0.28	0.38
Implicit (ASR)	0.64	0.25	0.35

(b) **Suggestions = Notes Benchmarks**	P	R	F
Wizard suggestions	0.53	0.47	0.49
Implicit (manual) suggestions	0.55	0.23	0.32
Implicit (ASR) suggestions	0.46	0.18	0.26

(c) **Other Benchmarks**	P	R	F
Expert 1 v 2	0.59	0.50	0.53
Original meeting participants	0.78	0.23	0.36
Mismatched meetings	0.33	0.14	0.19

Table 7.5: ROUGE analysis of different notes against Expert notes. (a) Evaluation of notes taken by study participants in the four conditions. (b) Evaluation of suggestions-as-notes, i.e. if exactly all the suggestions (and no other text) were used as notes. (c) Evaluation of 3 other benchmark sets of notes – the experts' notes evaluated against each other, the notes of the original participants of the meetings, and a sanity check: evaluation of study participants' notes from one meeting against Expert notes of other meetings. P, R and F stand for ROUGE-1 precision, recall and Fmeasure.

Condition	**Meeting 1**			**Meeting 2**			**Meeting 3**		
	P	R	F	P	R	F	P	R	F
Control	0.46	0.24	0.30	0.69	0.27	0.39	0.60	0.22	0.31
Wizard	0.50	0.34	**0.40**	0.65	0.38	**0.47**	0.63	0.29	**0.39**
Implicit (manual)	0.52	0.36	**0.40**	0.67	0.40	**0.49**	0.62	0.22	**0.31**
Implicit (ASR)	0.46	0.24	0.31	0.65	0.30	0.40	0.65	0.25	0.35

Table 7.6: ROUGE-1 precision (P), recall (R) and Fmeasure (F) evaluation values of study participants' notes, disaggregated by both condition and meeting (part 1 of 2, continued in Table 7.7).

Condition	Meeting 4		
	P	R	F
Control	0.61	0.31	0.39
Wizard	0.51	0.41	**0.45**
Implicit (manual)	0.57	0.24	**0.33**
Implicit (ASR)	0.61	0.28	0.38

Table 7.7: ROUGE-1 precision (P), recall (R) and Fmeasure (F) evaluation values of study participants' notes, disaggregated by both condition and meeting (part 2 of 2, continued from Table 7.6).

7.10.3 Comparison against Benchmarks

Poritions (b) and (c) of Table 7.5 present several "benchmark" results against which we can compare the per-condition results presented in portion (a) of the same table and discussed above. In portion (b), we present the "benchmark" of comparing the suggestions for each condition against the experts' notes. That is, these are the ROUGE-1 precision, recall and Fmeasure values that would have been obtained if, for each meeting, participants constructed their notes by accepting *all* the suggestions shown to them, and wrote no additional text on their own. Compare the evaluation numbers of each of the conditions in portion (b) to the corresponding conditions in portion (a). Observe that for the Wizard condition, participants would have received a higher Fmeasure score if they had simply selected all the Wizard's suggestions without writing any notes on their own. While this is not statistically significantly higher, the precision of the study participants' notes (0.66) was significantly higher and the recall (0.35) significantly lower than the corresponding numbers of the Wizard "suggestions = notes" benchmark ($p < 0.01$ in both cases; as mentioned previously, two-sample bootstrap can compute significance between any two sets of samples, even when they are not paired and are of unequal sizes). The higher precision and lower recall of the user study participants can be partly explained by the fact that for each meeting, there were many more words in all the suggestions put together (193) than there were in the participants' notes on average (92.4).

For the Implicit (manual) and Implicit (ASR) conditions, participants' notes using the suggestions outperforms the suggestions on their own. For the Implicit (manual) condition, participants' notes have marginally significantly more precision ($p < 0.05$), but not significantly more recall or Fmeasure. For the Implicit (ASR) condition, participants' notes have marginally higher precision and Fmeasure ($p < 0.05$) and the higher recall is only a statistical trend ($p < 0.1$). These results show that while the Implicit (manual) suggestions are poor as notes on their own, they help participants improve their notes slightly as compared to the Control condition (although not significantly so on average across all 4 meetings, as discussed above). On the other hand, the Implicit (ASR) suggestions are poor on their own, and do not seem to help participants improve their notes above the Control condition at all.

Finally, portion (c) presents 3 miscellaneous benchmarks for comparison. The first of these rows contains the ROUGE precision, recall and Fmeasure values of the experts' notes for each meeting, when compared against each other (and averaged over the 4 meetings). The high Fmeasure value shows that the notes of these experts overlap with each other more than any other set of notes overlap with them, implying a relatively higher degree of agreement among experts as to what information is noteworthy in each meeting. On the other hand, observe that these values are still far from the theoretical upper limit of 1.0 for the Fmeasure metric, underlying once again the subjectivity involved in the case of note evaluation.

The second row of portion (c) shows the evaluation of the notes that the original participants of these meetings wrote (when the meetings were being recorded), evaluated against the Expert notes. Observe that they have a similar recall as study participants in the Control condition (portion (a) of the same table), but much higher precision. Indeed, the original participants achieved the same recall while using 25% fewer words – whereas the notes of study participants in the Control condition had 80 words on average, the original participants had only 60. This similar recall despite smaller lengths, and much higher precision are outcomes of the fact that the original participants had a much better understanding of the meeting discussions that study participants. (Note also that as described above, the 2 experts against whom all notes are being evaluated were drawn from the pool of original participants. It is likely that the experts and the original participants have a shared understanding of the meeting contents, and thus the original participants' notes received a high precision value. Despite this difference in understanding of the meeting discussions, observe that study participants achieved a higher Fmeasure in both the Implicit (manual) and the Wizard conditions. Although part of this improvement can be attributed to the fact that the study participants were solely focussed on the note-taking task rather than also participating in the conversations, these improvements perhaps imply that the original participants would also have been benefitted if they had access to these suggestions.

The last row of portion (c) provides a "sanity check". ROUGE is sometimes justifiably criticized as a insensitive metric. To get a sense of its degree of sensitivity, we computed ROUGE between study participants' notes from each meeting and experts' notes of *other* meetings. The resulting precision, recall and Fmeasure numbers are substantially lower than all other "matched" comparisons, thus implying that the metric is at least sensitive *enough* to reliably distinguish between matched and unmatched comparisons.

	Control	Wizard	Implicit (manual)	Implicit (ASR)
Control	x	0.39	0.37	0.37
Wizard	0.39	x	0.41	0.41
Implicit (manual)	0.37	0.41	x	0.39
Implicit (ASR)	0.37	0.41	0.39	x

Table 7.8: Average ROUGE-1 Fmeasure when comparing notes taken by user study participants in different conditions. For each pair of conditions, only notes of the same meeting are compared to each other.

7.10.4 Comparing Conditions without External Judgement

In the previous sections, we evaluated the notes created by the user study participants by comparing them to notes created by external judges – experts, meeting annotators, the original meeting participants, etc. By doing so, we had assumed that the judges' notes were "canonical", and that for a particular set of notes, greater overlaps with those canonical notes implied higher quality. It is possible, however, to compute a relative ranking of the notes produced in the various conditions *without* making such an assumption and without using external notes. In this section we perform such an evaluation and compute the relative ranking of the four conditions in terms of the average quality of notes taken in those conditions.

We perform this evaluation in four iterations. In each iteration we hold the notes from one condition as the ground truth, and use those notes to evaluate the notes in the other conditions. Each such iteration will result in a ranking of the quality of the other 3 conditions. If the ranking of the conditions in these 4 iterations are consistent with each other, we can merge the rankings to create a global ranking of the 4 conditions, and compare that ranking to the one found previously by evaluating against expert notes.

The detailed algorithm to perform the evaluation in each iteration is as follows. Let N_{mci} denote the notes taken by participants who were presented with condition c in meeting m. Since there are 6 participants for each combination of m and c, let $i \in [1...6]$ denote the notes from these 6 participants for that combination. In each iteration, denote C as the condition being considered as the ground truth. Then, the average quality of each condition C' ($C' \neq C$) is computed as:

$$Q(C') = \forall_{m,i,j} AVERAGE(ROUGEF1(N_{mCi}, N_{mC'j}))$$

Given a ground truth condition C and a condition to be evaluated C', the above formula results in 36 computations of ROUGEF1 for each meeting m, and 144 across all 4 meetings for that condition. The final value of $Q(C')$ is thus the average of these 144 ROUGEF1 scores.

Table 7.8 shows the results of this evaluation. Since ROUGEF1 is a symmetric function (ROUGEF1(a, b) = ROUGEF1(b, a)), the conditions on either the rows or the columns can be considered the ground truth. Based on these evaluation numbers, we observe the following relative ranking of the conditions in the four iterations:

- With the Control condition as ground truth: Q(Wizard) $>$ Q(Implicit Manual) $\geq$ Q(Implicit ASR)
- With the Wizard condition as ground truth: Q(Implicit Manual) $\geq$ Q(Implicit ASR) $>$ Q(Control)
- With the Implicit Manual condition as ground truth: Q(Wizard) $>$ Q(Implicit ASR) $>$ Q(Control)
- With the Implicit ASR condition as ground truth: Q(Wizard) $>$ Q(Implicit Manual) $>$ Q(Control)

Thus the ranking of the conditions is consistent across the four iterations. When merged together, we form the following global ranking of the four conditions:

Q(Wizard) $>$ Q(Implicit Manual) $\geq$ Q(Implicit ASR) $>$ Q(Control)

Observe that this relative ranking of the conditions is precisely the same as the one found by evaluating against external judges, thus validating comparison against external judges as an evaluation strategy.

7.10.5 Conclusion

Figure 7.4 summarizes the note-taking effort and note quality results presented above (error bars are again based on standard error). Observe that the lowest effort and highest note-quality are both achieved in the Wizard condition. The implicit manual condition also features significant savings of effort as compared to the control condition, as well as slight improvement in note quality. The Implicit (ASR) condition requires much more effort, and the resulting quality of the notes is approximately the same as those in the control condition. These results imply that the automatically trained system helps meeting participants save on their note-taking effort, while improving their notes slightly. As has been mentioned above, the improvement in the note quality is more drastic in some meetings, and less in others. We conclude therefore that an automatically trained note-suggestion system can sometimes be helpful to meeting participants in creating more detailed notes with less effort.

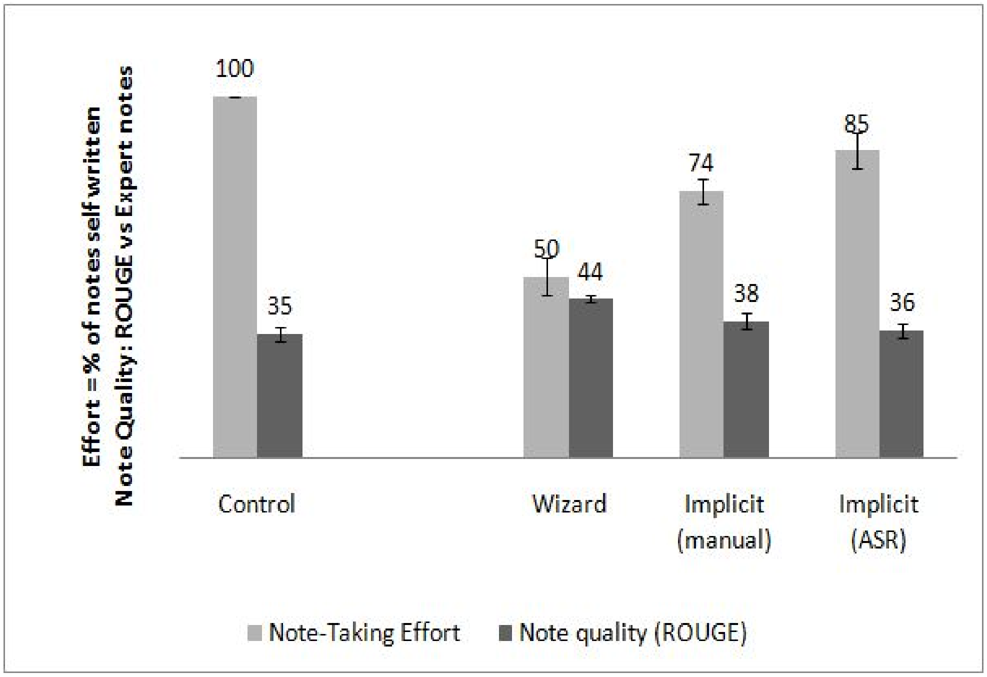


Figure 7.4: Note-Taking effort and note quality for the four conditions. Note quality is values are ROUGE-1 Fmeasure scores, presented on a scale of 0 to 100 rather than on a scale of 0 to 1..

7.11 Results: Questionnaire Analysis

In the previous sections we have presented concrete evaluations of user study participants' interactions with the suggestions, the amount of effort they expended in creating the notes, and the quality of the resulting notes in each of the 4 conditions. In addition to these objective observations of study participants' note-taking behavior, recall that we also asked participants to provide us with their perceptions about various aspects of the suggestions. In this section we present an analysis of their answers to these questions.

After each experimental meeting, the study participants were asked to rate 5 aspects of the suggestions:

- The usefulness of the suggestions.
- The intelligibility of the sugestions – whether it was easy or difficult to understand the text of the suggestions.
- The number of suggestions shown – whether they were shown too few or too many

suggestions.

- The timeliness of the suggestions – whether the suggestions arrived on time or too slowly to be useful.
- The degree of distraction caused by the suggestions.

In addition, participants were asked to rate their willingness to use a system that provided similar suggestions in their own meetings. The participants' answers, averaged over the 24 participants, are presented in Figure 7.5, along with the standard errors. Note that participants were not asked the questions in the order presented in the table; the order shown here is for ease of explanation.

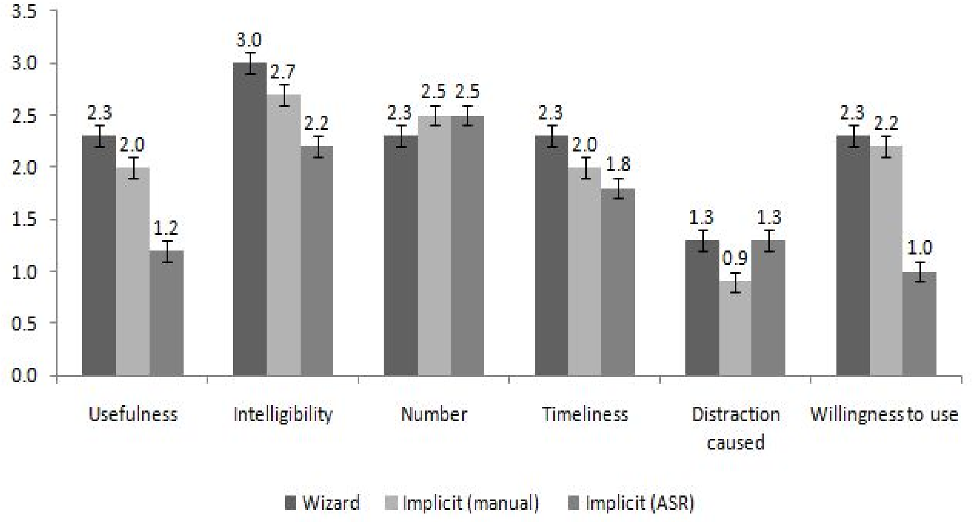


Figure 7.5: Average answers from study participants for each of the 6 questions asked after each condition. Each question had five possible answers with values 0 through 4. Section 7.11 presents the interpretations of these values. Error bars are drawn using Standard Error.

7.11.1 Usefulness of Suggestions

The first question asked study participants for their opinion about the *usefulness* of the suggestions offered to them in the immediately preceding condition. The wording of the question was:

> In general, how useful were the suggestions during the last meeting?

Our goal was to have study participants provide a Likert-like 5-point-scale answer to this question, ranging from 0 to 4, with 0 interpreted as "I found the suggestions not useful at all" and 4 interpreted as "I found the suggestions very useful". However, instead of providing the participants with interpretations of the extreme ends of the scale only, and requiring them to create their own interpretation of the numerical answer values between 0 and 4, we showed participants answer texts that provided them with an interpretation for each integer value from 0 through 4. In fact, study participants were not shown the numeric answer values at all. Instead, they were shown only the answer texts, and internally each answer text was assigned to a numeric value. Following are the answer texts shown to the participants, and the internal numeric value assigned to each answer:

- In general, I found the suggestions not useful at all during the last meeting. (Value = 0)
- In general, I found the suggestions not really useful during the last meeting. (Value = 1)
- In general, I am neutral about the usefulness of the suggestions during the last meeting. (Value = 2)
- In general, I found the suggestions useful during the last meeting. (Value = 3)
- In general, I found the suggestions very useful during the last meeting. (Value = 4)

Note that phrases underlined in the question and answers above were also underlined in the questionnaire presented to the participants in order to draw their attention to the crucial pieces of information in the question and the answers. The first group of bars in Figure 7.5 presents the average numeric answer provided by the 24 participants of the user study for each condition. Observe that, on average, participants rated the usefulness of the suggestions in the Wizard and Implicit (manual) conditions as statistically significantly higher than the usefulness of the suggestions in the Implicit (ASR) condition ($p < 0.01$ for both pairs of comparisons). This result shows the strongly negative effect that speech recognition errors have on suggestion usefulness.

The difference between the rated usefulness of the suggestions in the Wizard and the Implicit (manual) conditions is not statistically significant. That is, participants found the automatically generated suggestions nearly as useful as the Wizard generated ones. This is an encouraging result, and is perhaps related to the fact that participants accepted roughly the same *fraction* of suggestions in the two conditions (Table 7.2). At the same time, it appears that the fact that participants accepted many more suggestions in the Wizard condition, and that consequently they could create longer and more accurate notes with less effort than in the Implicit (manual) condition did not impact their *perception* of the usefulness of the suggestions in the two conditions.

7.11.2 Intelligibility of Suggestions

In the next question, participants were asked about the intelligibility of the suggestions provided to them. Specifically, the question asked:

> In general, how easy or difficult was it for you to understand the text of the suggestions?

As in the previous questions, participants were shown 5 answer choices, and each answer was internally assigned to a numeric value. The answer texts and numeric values were as follows:

- In general, I had a lot of difficulty understanding the text of the suggestions during the last meeting. (Value = 0)
- In general, I had some difficulty understanding the text of the suggestions during the last meeting. (Value = 1)
- In general, I am neutral about the understandability of the text of the suggestions during the last meeting. (Value = 2)
- In general, I could easily understand the text of the suggestions during the last meeting. (Value = 3)
- In general, I could very easily understand the text of the suggestions during the last meeting. (Value = 4)

The results of this question are presented in the second group of bars in Figure 7.5. Like in the case of the suggestion usefulness question, there was no statistical difference between the reported intelligibility of suggestions in the Wizard and Implicit (manual) conditions, while the intelligibility of the Implicit (ASR) condition was significantly lower ($p < 0.01$), probably due to speech recognition errors. The similarity in intelligibility scores between the Wizard and Implicit (manual) conditions is probably because both conditions used manually transcribed speech. Despite the perfect transcription, however, the average scores for these two conditions is around 3.0 which is interpreted as “easily understand”, and not the highest score of 4.0 or “very easily understand”. This is probably because some participants may not have understood some of the suggestions simply because they were technical in nature and difficult to understand for someone unfamiliar with the topics of discussion.

7.11.3 Number of Suggestions

The next question asked study participants for their opinion about the *number* of suggestions offered to them in the immediately preceding condition. The wording of the question was:

> Would you have liked more or fewer suggestions during the last meeting?

The answer choices shown to the participants and their corresponding internal numeric values were as follows:

- I would have liked far fewer suggestions during the last meeting. (Value = 0)
- I would have liked fewer suggestions during the last meeting. (Value = 1)
- The number of suggestions was just right during the last meeting. (Value = 2)
- I would have liked more suggestions during the last meeting. (Value = 3)
- I would have liked far more suggestions during the last meeting. (Value = 4)

The third group of bars in Figure 7.5 presents the average numeric answer provided by the 24 participants of the user study for each condition. Observe that, these values are similar to each other, and in fact there is no statistical difference between them. This is a surprising result because many more suggestions are shown in the Wizard condition than are shown in the Implicit conditions (Table 7.1). If participants wanted somewhat more suggestions than those shown in the Wizard condition, it is reasonable to expect that they would want *many* more than those shown in the Implicit conditions. Although the average answers for the Implicit conditions are indeed more than that in the Wizard condition, they are not significantly higher. It is possible that some participants did not expect increasing the number of suggestions would necessarily improve the quality of the suggestions, and so did not vote for more suggestions.

7.11.4 Timeliness of Suggestions

In the next question, participants were asked about the speed with which the suggestions appeared after the corresponding utterance was spoken by some speaker in the recorded meeting. The specific question was:

> In general, what do you think of the timing of the suggestions during the last meeting?

The answer choices shown to the participants and their corresponding internal numeric values were as follows:

- In general, the suggestions arrived very late during the last meeting. (Value = 0)

- In general, the suggestions arrived somewhat late during the last meeting. (Value = 1)
- In general, I am neutral about the timeliness of the suggestions during the last meeting. (Value = 2)
- In general, the suggestions arrived more or less on time during the last meeting. (Value = 3)
- In general, the suggestions arrived just on time during the last meeting. (Value = 4)

The fourth group of bars in Figure 7.5 presents the average timeliness ratings in the 3 conditions. The differences between the ratings are not statistically significant. The fact that there are any differenes at all is interesting because, as mentioned earlier, all suggestions were timed to be shown 3 seconds after they were completely spoken. Despite this design, participants appeared to slightly penalize the suggestions in the Implicit (ASR) condition. This outcome is possibly a spill-over effect from the poor ratings that this condition gets in the other questions.

7.11.5 Distraction Caused by Suggestions

In the last of the questions about the suggestions, participants were asked to what degree they found the suggestions distracting from their note-taking task. Specifically they were asked:

> In general, how distracting (or not) were the suggestions?

The answer choices shown to the participants and their corresponding internal numeric values were as follows:

- In general, the suggestions did not distract me at all during the last meeting. (Value = 0)
- In general, the suggestions did not distract me much during the last meeting. (Value = 1)
- In general, I am neutral about the distractions from the suggestions during the last meeting. (Value = 2)
- In general, the suggestions distracted me quite a bit during the last meeting. (Value = 3)
- In general, the suggestions distracted me a lot during the last meeting. (Value = 4)

The fifth group of bars in Figure 7.5 present the results of this question. The differences between the 3 conditions are not statistically significant. In general, participants did not appear to be distracted much by the suggestions, even in the Wizard condition in which many suggestions were shown to them. This result is likely an outcome of the interface design decision of having the suggestions appear in a separate panel to one side of the note-taking panel, instead of appearing either inline or over the participants' notes. The suggestions were thus easy to ignore when not useful (such as in the Implicit (ASR) condition) and easy to use when useful, such as in the other conditions.

7.11.6 Willingness to Use a Similar System in their Own Meetings

Finally participants were asked the following question:

> If you had access to a system that made suggestions of a similar quality as the suggestions made during the last meeting, would you be willing to use such a system in your own meetings? (By "similar quality" we mean suggestions of similar frequency, usefulness, timing, understandability, and degree of distraction.)

The goal of this question was to capture an overall "satisfaction" score for the suggestions shown in the previous condition, and whether participants at least claim to wish to use such a system for their own meetings. The possible answers and answer values were as follows:

- I won't use a system with similar suggestions in my meetings. (Value = 0)
- I might not use a system with similar suggestions in my meetings. (Value = 1)
- I am neutral about using a system with similar suggestions in my meetings. (Value = 2)
- I might use a system with similar suggestions in my meetings. (Value = 3)
- I will use a system with similar suggestions in my meetings. (Value = 4)

The last group of bars in Figure 7.5 presents the results of this question. Observe that, similar to the results of the question on the usefulness of the suggestions, participants rated the Implicit (ASR) system statistically significantly lower than both the Wizard and the Implicit (manual) systems ($p < 0.01$), and the differences between the ratings given to those two latter systems was not statistically significant. Note that the two questions – regarding the usefulness of the suggestions and the participant's willingness to use this system for his own meetings – were presented far apart in the questionnaire. Thus, the similarity in the scores may reflect participants' true satisfaction with the systems, rather than a randomly assigned rating.

Similar to the results of the question on suggestions usefulness, it is encouraging to note that participants were nearly as willing to use the automatically trained suggestion system in their meetings as they were willing to use the human curated Wizard suggestion system in their meetings. This result implies that not only do participants' notes improve to some degree when using the Implicit (manual) suggestions, they are as satisfied with those suggestions as they are with the Wizard suggestions.

On the other hand, observe that participants gave the Wizard system a "willingness score" that was merely between the "neutral" and "might use" answers. This relative lack of enthusiasm for even the Wizard system comes despite the fact that participants' notes were statistically significantly closer to that of experts' notes when using the Wizard system, as compared to when participants had no suggestions at all. This low willingness score is possibly a result of a gap between the performance of the system and participants' prior expectations. Anecdotally, several participants admitted such a gap to the experimenter in discussions after the conclusion of the experiment. As with all systems that promise "intelligent behavior", it is likely that expectation management will play an important role in the commercial success of a notes-assistance system.

7.12 Results: Post-Experiment Questionnaire

At the end of the experiment, participants were provided with a final questionnaire in which they were asked to evaluate the major portions of the user interface. First they were asked to report the ease of using the suggestion-acceptance mechanism:

> How easy or difficult was it for you to use the click-to-insert notes suggestions interface?

The answer choices and associated internal answer values were:

- It was very difficult to use the click-to-insert interface. (Value = 0)
- It was difficult to use the click-to-insert interface. (Value = 1)
- I am neutral about the ease of using the click-to-insert interface. (Value = 2)
- It was easy to use the click-to-insert interface. (Value = 3)
- It was very easy to use the click-to-insert interface. (Value = 4)

Note that although the intent of the question was to ask about both forms of insertion mechanisms – both click-to-insert and select-to-insert – the question was incorrectly worded as above instead of being worded as “How easy or difficult was it for you to insert suggestions into your notes?” which would have included both forms of insertion interfaces. From the free-form answers to other questions in this post-experiment questionnaire it *appears* that participants did not in fact distinguish between clicking to insert and selecting to insert, but we do not know for sure.

On average participants gave this question a score of 3.7, i.e. between “easy” and “very easy”, thus signifying the ease of use of this suggestion insertion interface.

In a free-form text field, participants were asked to explain what specifically they liked about the suggestion selection interface. The single biggest reason given was the ease with which text could be transferred into the notes. Participants were also asked to specify what they did *not* like about the suggestion interface. 8 participants said they had no problems, 4 said they did not like the spacing between the suggestions, and 3 mentioned they did not like that some utterances got split into multiple lines.

Separate from the suggestion-acceptance interface, participants were asked what they liked about the note-taking portion of the interface. 10 participants mentioned that they liked the fact that it was a simple Notepad-like interface that “got out of the way”. 4 said they would have liked a different font, but did not mention any particular font that they would have preferred.

Finally participants were given one last text box to provide any last thoughts about the software. 19 of the 24 participants took this opportunity to provide feedback. 5 participants wanted suggestions that were more “to-the-point”. This perhaps means that participants desired suggestions that were more abstractive in nature than the extracted utterances that they were shown during the experiment. Along these lines, one participant mentioned that she would have liked to see just phrases pop-up which she could insert into her notes. Another participant wrote that she would have liked to see names and dates be suggested as notes. Yet another participant gave the opposite advice – to present a full transcription of the meeting so that she could pick and choose the words and phrases she needed to create her notes. One astute participant honed in on a crucial problem that becomes evident in the results above – the poor quality of the speech recognition; he advised focusing all effort on improving this aspect of the software first. Finally, one participant wondered whether having a notes-suggestion system that worked too well might discourage active note-taking and thus leave participants overly reliant on an artificial system that may or may not work in a new meeting in the future.

7.13 Summary

In this chapter we have tested the usefulness of note-suggestions to meeting participants, when those note-suggestions are generated by an automatically trained noteworthy-utterance detector. Towards this end, we have conducted a within-subjects user study were study-participants mimicked meeting participants by taking notes while listening to 4 recorded meetings. In three of the meetings they were given note-suggestions from, respectively, the Wizard, the Implicitly trained system using manual transcripts, and the Implicitly trained system using ASR transcripts. We have shown the following major results:

- Study participants are willing to accept similar fractions of suggested utterances in both the Wizard and the Implicit (manual) conditions.
- As a result of these suggestion acceptances, participants can create more detailed notes, and save between 26% (in the Implicit (manual) condition) to nearly 50% (in the Wizard condition) of their note-taking effort by simply using the suggested texts.
- Participants' notes improve significantly when using the Wizard suggestions, and modestly when using the Implicit (manual) suggestions.
- Participants perceive the Wizard and the Implicit (manual) suggestions as being somewhat useful for note-taking, and are moderately willing to try those systems in their own meetings.
- Speech recognition errors have a drastic negative impact on the usefulness of the suggestions, both in concrete terms (number of suggestions used, note-taking effort saved, and note quality) and also in participants' perceptions of their usefulness.

Chapter 8

Conclusions

8.1 Summary

In this thesis, our goal was to explore novel approaches to extracting supervision from human's interactions with systems. In Chapter 2 we reviewed past work on supervision extraction, with or without explicitly asking humans to provide feedback. We showed that most previous work has focussed on domains and tasks where the system's actions mimic the human's, leading to easy interpretation of human actions as labeled data. Our goal was to investigate novel tasks where the human and the system actions are starkly different from each other, and special techniques must be applied to interpret human actions as feedback.

We performed our investigations within the realm of meeting understanding. In chapter 4 we showed that busy meeting participants miss meetings regularly, and find it time-consuming (and sometimes impossible) to find the information discussed at those meetings. On the other hand, we showed through a user study that if participants are given a recorded meeting, they can quickly find the information they are interested in if the meeting is pre-segmented into the different agenda items. These findings indicate a need for automatic agenda segmentation of meetings.

The only visible human actions in meetings are the meeting participants' speech and their notes, whereas the ideal labeled data needed for training a meeting segmenter is example meetings paired example segmentations (in the form of a list of starting and stoping timestamps for the different agenda items discussed during each meeting, for example). These two actions – note taking and segmenting meetings – are very different, so we proposed a novel approach to extracting supervision. Specifically, we proposed a 3-step general recipe to design an interface in order to extract labeled data. While this recipe is not guaranteed to work for every task and every domain, we applied it (retrospectively) to a previous work on a specialized interface design, and also to the current task of supervision extraction for meeting segmentation.

We implemented and deployed SmartNotes, the specially designed note-taking interface. We showed in Chapter 5 that over a period of many weeks, participants were willing to use the newly designed note-taking interface. We evaluated the labeled data collected in real meetings, and showed that it is statistically significantly better than similar data collected through a simple standard unsupervised baseline, and of similar quality as data created through a state-of-the art unsupervised algorithm. Although this automatically extracted data was not as accurate as manually created data, it was competitive with other methods that require similar amounts of manual labor. We thus conclude that even when the human and the system tasks are starkly different, it may be possible to extract supervision from the human task, particularly through careful design of the human-system interface.

Another finding from the user study conducted in Chapter 4 is the importance of notes. For participants who need to retrieve information from past meetings, whether they have access to notes or not strongly influences the likelihood that their information need will be met. This finding provides motivation for the second task explored in this thesis – automatic note-taking assistance.

In Chapter 6 we explored the task of noteworthy-utterance detection. We evaluated the standard 2-class formulation of this problem, where utterances are labeled as either "noteworthy" or "not noteworthy". We showed that such a formulation results in extreme skew in the labeled data, and that for many border-line utterances annotators make random choices as to whether to label it "noteworthy" or not. We experimented with a 3-class formulation, and showed that indeed inter-annotator agreement improved, despite the fact that annotators now have more classes to choose from. We also showed that a noteworthy-utterance detector trained on this data performs better than one trained on the 2-class formulation.

Later in Chapter 6, we presented an algorithm to automatically extract utterances labeled as noteworthy or not from the notes and utterances in previous related meetings. We showed that this data is significantly *more similar* to the notes that the original meeting participants took than manually annotated data. Further, we showed that even noteworthy-utterance detectors trained on such automatically extracted data substantially outperform those trained on manually annotated data, again on the metric of similarity to the original notes of the meeting.

Finally in chapter 7, we undertook a within-subjects user study to perform an extrinsic evaluation of a note-suggestion system developed on the basis of the automatically trained noteworthiness classifier. We showed that using manually transcribed note-suggestions, participants were able to take more notes of slightly better quality than when they were shown no suggestions at all, while simultaneously lowering the note-taking effort. We also showed the strongly negative impact of speech recognition errors on the usefulness of the note-suggestions.

8.2 Contributions

The technical contributions of this thesis are:

- A formal user survey of busy professionals that uncovered the information needs of meeting participants who miss previous meetings or simply need to recall information from previous meetings they attended (Chapter 4).
- A novel general approach to extracting implicit supervision from humans when human and system actions do not match (Chapters 3 and 5).
- A novel approach to extracting labeled data for meeting segmentation research from meeting participants' notes through a specially designed note-taking interface (Chapter 5).
- A novel approach to developing a complete end-to-end-automated note-suggestion system that, with no explicit manual supervision, uses meeting participants' notes in previous meetings to learn to identify noteworthy utterances in future meetings in order to assist meeting participants in taking notes (Chapter 6).
- A formal user study to evaluate the usefulness of such a fully-automated note-suggestion system (Chapter 7).

8.3 Future Work

The experimental results point to several avenues of future research. First, the notes-suggestion user study results in Chapter 7 indicate that the greatest impediment to the usefulness of a fully automated system is the quality of the speech recognizer. As one of the user study participants mentioned, a lot of attention must be paid to improving the accuracy of the recognizer for meeting speech. This is both a necessary and challenging task. One possible approach to improving speech recognition in meetings is to perform acoustic and language model adaptation. Since our entire note-suggestion approach is premised on the existence of a *sequence* of meetings with largely overlapping participants, such adaptation approaches hold promise.

Beyond improvements in speech recognition accuracy, improvements are also needed in the note suggestions themselves. Recall that even with human-quality transcription and human-quality noteworthiness detection (the "Wizard" condition in the user study conducted in Chapter 7), participants accepted only about 37% of the suggestions, and their notes were still significantly worse than Experts' notes. This result may signal a short-coming of an extractive-summarization based system. While such a simple system was adequate in creating suggestions with enough value that participants were willing to utilize them, to get more traction it is likely that more sophisticated techniques must be used to create *abstractive* summaries. Various approaches to abstractive summarization have been proposed in the past, both deep-semantic (such as via dialog structure recognition) as well as shallow (word- or phrase-spotting).

Some additional features may also play a role in such abstraction. While researchers have shown that prosodic features are often redundant when used alongside ngram-based features, this assertion needs to be retested when moving to a phrase-spotting-type system. Availability of other environmental information – like the project documents, emails, etc. – are also likely to help the system get a better understanding of important information in the meetings.

Finally, we have explored note-suggestion work only from the point of view of meetings. However, the techniques developed here are likely also equally applicable in the context of lectures and presentations. While in the domain of meetings, the obvious extrinsic metric to be optimized is worker productivity, in the case of an academic environment, other metrics such as the students knowledge-gain will likely also need to be considered. Specifically, whether to show a note-suggestion may be guided not only based on its importance, but also on the pedagogical impact of having the student see that suggestion. Will it help him remember that piece of information better? Will it distract him from paying full attention to the teacher's speech? Note too that the system's task can potentially be *easier* in classroom lectures, if its goal is to only recognize the speech of the lecturer. The more he lectures about a particular topic, and the more external information (such as the contents of the slides, and the textbooks) are made available to the system, the more accurate its recognizer is likely to be. This is a fascinating area of research, and we expect speech-based lecture notes-assistance systems to play a big role in the near future.

Appendix A

Script of Instructions given to User Study Participants

A.0.1 Introduction and Overview

Thank you for participating in this experiment. This experiment is designed to last no more than an hour and a half, and you will be paid $15 at the end of that period. In this experiment you will listen to 4 recorded segments of meetings. As you listen to each meeting you will take notes in a new note-taking program that we have developed. After each meeting segment, you will fill out a short questionnaire. Also you'll have two more questionnaires, one right at the beginning, and one right at the end.

A.0.2 Consent Form Overview

Now, before we go any further, I would like you to sign the participant consent form. In this form you are allowing us to record your notes and the answers you give in the questionnaires, and allowing us to analyze them for our research. In return, we are promising that we will not associate your identity with the data – the data will be all anonymous. Also we will not share this data outside CMU. We might use snippets of this data in research papers and presentations, but again we won't use your name or identity. Now go ahead and read and sign the form. Feel free to can ask me any questions you have!

A.0.3 Do the Pre-Test Survey

Okay, first I'll have you fill out the first big survey. Once you are done with this survey please inform me.

A.0.4 SmartNotes Demo

As I told you earlier, you will have to listen to recorded meeting segments and take notes. To take notes, you will be using a special note-taking program called SmartNotes. I will now show you how to use SmartNotes. Then I'll show you how the experiment will run.

1. The first step is to start SmartNotes from the Start menu. Go ahead and click on the Start menu.

2. You'll notice that there are a bunch of these ".smn" files. During the experiment you'll be instructed to click on these one by one (the instructions will tell you exactly which one to click on). Right now let's start with the "Demo.smn" file. Go ahead and click on it.

3. So this is what SmartNotes looks like. There are three boxes, and each will contain different pieces of information.

4. At the bottom left, there will be definitions of words and phrases spoken in the meeting that may be unfamiliar to you. There will be three columns, and two groups – names of people, places, products, and technical terms. They will be in alphabetic order. Before you start taking notes, please read these definitions and familiarize yourself with the terminology. This will make it a lot easier for you to understand the meeting and take notes.

5. At the right you have the "Note Suggestions" box. As the audio plays, the program will put sentences it thinks are important in this box. If you think that some of these sentences are usful for the notes, you can click on them, and they'll get inserted into your notes – I will show you how in a minute.

6. Finally you will write your notes in this "Your Notes" area. To do, you have to first log in. Here is your login user name and password. Both the user name and the password is the same.

7. To log in, you'll click on the "Login" button [point to it]. Go ahead and click on it. Now enter your username and password, and hit "Sign in". Now go ahead and hit "Join".

8. To take notes, click on the "Your Notes" area, and start typing! Remember, you never have to save your notes, that happens automatically.

9. Of course there are no suggestions right now, because there is no audio playing just yet. I'll show you how to use suggestions in a bit.

10. Once you are done taking notes, and you are instructed to close SmartNotes, just click on the X. There is no need to save your notes, they are automatically saved. So go ahead and quit.

11. Now I will show you how the experiment will run. The experiment is fully automated, and it will give you instructions from time to time. You'll just have to follow the instructions.

12. Here's the first instruction, asking you to open the Demo.smn file from the Start menu. Go ahead and click okay, and then open the file.

13. This next message reminds you to read through the phrase definitions. Don't click on "Okay" before you've read through the suggestions. Once you've read the definitions, click on the taskbar here, and then click on Okay.

14. Now this message tells you to Login. You'll click on Login and put in your password. Then you'll click on "Join". So let's do that.

15. Now this message is telling you that you are ready to listen to the meeting. Remember, once the meeting starts, you won't be able to stop or pause the audio. So, if you need a bathroom break, this is a good time to go for it. Once you are ready, hit the Okay. (Remember, the real meetings are going to be longer.)

16. Check out that suggestion. You can include it in your notes in two ways. First, you can click on it. Go ahead and click on it and see what happens.

17. There is another way to include the suggestion in your notes – you can select portions of it. When you let go of the mouse, it gets inserted into your notes. Try it out!

18. You can even select multiple suggestions at once, and include them in your notes. The suggestions get inserted into your notes wherever your cursor is.

19. By the way, that "ding ding" you hear at the end signals that the meeting has ended. Until you hear it, the meeting is still going on, even if people pause.

20. Once the meeting ends, you'll see message asking you to shut down the note taking software. Press okay, and just close it down.

21. As soon as you do, you'll get a questionnaire. This questionnaire asks you about your experience in this particular meeting. You'll get one such questionnaire after each meeting. Fill it out, and hit submit. For now, we'll just skip over it, so why don't you hit exit.

22. And now it's going to ask you to start the next meeting. Instead of doing that, let's do the whole thing again, without me talking you through it.

23. In the end, you'll get one more survey to sort of get at the overall experience you had, and then you'll be done!

A.0.5 Guidance on Note-Taking

As you listen to each meeting, your goal is to take notes so that someone who wasn't at the meeting – particularly the person running the project, the academic advisor, etc., can get an overview of the important matter discussed at the meeting. Let me give you some examples of important information. Most of these should sound pretty reasonable to you.

- Action items – plans for work in the near or distant future.
- Deadlines or other time-sensitive information.
- Reporting on work done or progress since last week.
- Bringing up a new problem.

Remember these are just examples. As you listen to these meetings, you'll notice that there are lots of sentences that do not fit these categories. You have to bring your own important/unimportant judgment to bear on this problem.

You are not writing an essay here, so don't be overly worried about grammar, spelling, punctuation, etc. On the other hand remember that someone who is unfamiliar with your writing style needs to understand what you have written. So, for example, if you want to use abbreviations you may, but ask yourself if someone else can easily understand what the abbreviations mean.

There will be some technical jargon in these meetings, that's why you should familiarize yourself with the phrase definitions. Even after you do read them though, there will still be some things that you may not completely understand. Don't worry! Your goal is to figure out what the main important ideas are, and you should be able to do that even if you don't understand every word of what's being said.

Now let me give you a heads up about a few things you're going to notice during the experiment:

1. Sometimes you'll hear the people talking about writing notes, like "let me write down an action item". Or, you might hear the laptop keyboard. Assume that whatever notes they took are now lost. So if you think what they are saying is important, put it in your notes. On the other hand, maybe not everything they're writing down is really that important. So, use your own judgment.

2. Sometimes you might hear people talk about the note-taking software such as the “Action Item” button. Although they were using a version of the software you are about to use, you do not have access to any of the bells and whistles that they did. You must take all your notes within the note-taking box as I’ve already shown you.

3. When the system decides that it wants to show a suggestion there’s usually a little gap between when someone speaks the sentence, and when it shows up as a suggestion. Just be aware of that.

4. Sometimes you’ll notice that a single sentence is broken up into multiple suggestions. Like “finish this task” might be one suggestion and “by next week” might be the next one. If you want to include both sentences in your notes, remember you can do so in two ways - by clicking on them one by one, or by selecting them both at once

A.0.6 Final Advice regarding Suggestions

Finally, remember that in one of the 4 meetings you will have no suggestions at all. In the remaining 3 meetings there will be suggestions shown to you. You will notice that these suggestions vary widely in quality – sometimes they will be completely unrelated to things people are saying, and sometimes they might be relevant. Remember your goal is not to simply select all the suggestions and create your notes. Your goal is to create a good set of notes. Use the suggestions only if you think that doing so will help you create better notes, or help you create notes faster. Okay? Good luck!

Bibliography

[1] Mark S. Ackerman, Brian Starr, Debby Hindus, and Scott D. Mainwaring. Hanging on the ‘wire: a field study of an audio-only media space. *ACM Trans. Comput.-Hum. Interact.*, 4(1):39–66, 1997. 6

[2] S. R. Ahuja, J. Robert Ensor, and David N. Horn. The rapport multimedia conferencing system. In *COCS ’88: Proceedings of the ACM SIGOIS and IEEECS TC-OA 1988 conference on Office information systems*, pages 1–8, New York, NY, USA, 1988. ACM. 6

[3] J. Allan, J. Carbonell, G. Doddington, J. Yamron, and Y. Yang. Topic detection and tracking pilot study: final report. In *Proceedings of the DARPA Broadcast New Transcription and Understanding Workshop*, 1998. 33

[4] M. S. Bachler, S. J. Buckingham Shum, D. C. De Roure, D. T. Michaelides, and K. R. Page. Ontological mediation of meeting structure: Argumentation, annotation, and navigation. In *Proceedings of the* 1^{st} *International Workshop on Hypermedia and the Semantic Web (HTWS2003)*, Nottingham, UK, 2003. 24

[5] Satanjeev Banerjee, Carolyn P. Rosé, and Alexander I. Rudnicky. The necessity of a meeting recording and playback system, and the benefit of topic–level annotations to meeting browsing. In *Proceedings of the Tenth International Conference on Human-Computer Interaction*, Rome, Italy, September 2005. 23

[6] Regina Barzilay and Lillian Lee. Catching the drift: Probabilistic content models, with applications to generation and summarization. In *HLT-NAACL 2004: Proceedings of the Main Conference*, pages 113–120, 2004. 7

[7] D. Beeferman, A. Berger, and J. Lafferty. Statistical models for text segmentation. *Machine Learning*, 34(1-3):177 – 210, 1999. 7, 48, 55

[8] M. Burke, B. Amento, and P. Isenhour. Error correction of voicemail transcripts in scanmail. In *Proceedings of the SIGCHI conference on Human Factors in Computing Systems*, pages 339 – 348, Montreal, Canada, 2006. 9

[9] W Cohen. Learning rules that classify e-mail. In *AAAI Spring Symposium on Machine Learning in Information Access*, Palo Alto, CA, 1996. 9

[10] Peter Cook, Clarence Ellis, Mike Graf, Gail Rein, and Tom Smith. Project nick: meetings augmentation and analysis. *ACM Trans. Inf. Syst.*, 5(2):132–146, 1987. 6

[11] G Cormack. Trec 2006 spam track overview. In *The Fifteenth Text REtrieval Conference (TREC 2006)*, Gaithersburgh, MD, 2006. 11

[12] R. Cutler, Y. Rui, A. Gupta, J. J. Cadiz, I. Tashev, and L. He. Distributed meetings: A meeting capture and broadcasting system. In *Proceedings of the ACM Multimedia Conference*, 2002. 5, 6

[13] P. Ehlen, M. Purver, and J. Niekrasz. A meeting browser that learns. In *AAAI Spring Symposium: Interaction Challenges for Artificial Assistants*, Palo Alto, CA, 2007. 9

[14] J. A. Failes and D. R. Olsen. Interactive machine learning. In *International Conference on Intelligent User Interfaces*, pages 39–45, Miami, FL, 2003. 11

[15] M. Galley, K. McKeown, E. Fosler-Lussier, and Hongyan Jing. Discourse segmentation of multi–party conversation. In *Proceedings of the 41st Annual Meeting on Association for Computational Linguistics*, volume 1, pages 562 – 569, Sapporo, Japan, 2003. 7, 55

[16] Michel Galley. A skip-chain conditional random field for ranking meeting utterances by importance. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, Sydney, Australia, 2006. 59, 69, 71

[17] M. Garden and G. Dudek. Semantic feedback for hybrid recommendations in recommendz. In *The 2005 IEEE International Conference on e-Technology, e-Commerce and e-Service*, Hong Kong, 2005. 11, 12

[18] S. Godbole, A. Harpale, S. Sarawagi, and S. Chakrabarti. Document classification through interactive supervision of document and term labels. In 15^{th} *European Conference on Machine Learning (ECML) and the* 8^{th} *European Conference on Princicples and Practive of Knowledge Discovery in Databases (PKDD).*, Pisa, Italy, 2004. 12

[19] A. Gruenstein, J. Niekrasz, and M. Purver. Meeting structure annotation: Data and tools. In *Proceedings of the* 6^{th} *SIGdial Workshop on Discourse and Dialogue*, Lisbon, Portugal, September 2005. 39, 48, 49

[20] T. Hain, J. Dines, G. Garau, M. Karafiat, D. Moore, V. Wan, R. Ordelman, and S. Renals. Transcription of conference room meetings: An investigation. In *Proceedings of Interspeech 2005*, Lisbon, Portugal, September 2005. 7

[21] M. Hearst. TextTiling: Segmenting text into multi–paragraph subtopic passages. *Computational Linguistics*, 23(1):33–64, 1997. 33, 49, 50

[22] D. Huggins-Daines and A. I. Rudnicky. Implicitly supervised language model adaptation for meeting transcription. In *Proceedings of HLT-NAACL 2007*, Rochester, NY, 2007. 73

[23] A. Ionescu, M. Stone, and T. Winograd. Workspacenavigator: Tools for capture, recall and reuse using spatial cues in an interactive workspace. In *Stanford Computer Science Technical Report*, 2002. 6

[24] M. E. Jennex and L. Olfman. Organizational memory / knowledge effects on productivity, a longitudinal study. In 35^{th} *Annual Hawaii International Conference on System Sciences (HICSS'02)*, Big Island, Hawaii, 2002. 6

[25] T. Kemp and A. Waibel. Unsupervised training of a speech recognizer: Recent experiments. In *Proceedings of the Sixth European Conference on Speech Communication and Technology (Eurospeech 99)*, pages 2725–2728, Budapest, Hungary, 1999. 10

[26] M. Kumar, N. Garera, and A. I. Rudnicky. Learning from report-writing behavior of individuals. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, Hyderabad, India, 2007. 13

[27] Chin-Yew Lin. Rouge: A package for automatic evaluation of summaries. In *Proceedings of the ACL-04 Workshop: Text Summarization Branches Out.*, pages 74–81, Barcelona, Spain, 2004. 73, 80

[28] A Lisowka. Multimedia interface design for the multimodal meeting domain: Preliminary indications from a query analysis study. In *Technical Report IM2.MDM-11*, November 2003. 24, 26, 30

[29] Feifan Liu and Y. Liu. Correlation between rouge and human evaluation of extractive meeting summaries. In *Proceedings of the ACL-HLT*, Columbus, OH, 2008. 59, 69, 71, 83

[30] S. Maskey and J. Hirschberg. Comparing lexical, acoustic/prosodic, structural and discourse features for speech summarization. In *Proceedings of Interspeech 2005*, Lisbon, Portugal, 2005. 64, 65, 66

[31] F. Metze, Q. Jin, C. Fugen, K. Laskowski, Y. Pan, and T. Schultz. Issues in meeting transcription – the isl meeting transcription system. In *Proceedings of the 8th International Conference on Spoken Language Processing (Interspeech 2004 – ICSLP)*, Jeju Island, Korea, 2004. 7

[32] Y. F. Mohammad and T. Nishida. Naturaldraw: Interactive perception based drawing for everyone. In *Proceedings of the* 12^{th} *International Conference on Intelligent User Interfaces*, pages 251–260, Honolulu, HI, 2007. 9

[33] G. Murray, S. Renals, and J. Carletta. Extractive summarization of meeting recordings. In *Proceedings of Interspeech 2005*, Lisbon, Portugal, September 2005. 7, 63, 64, 65, 73, 76, 82

[34] Ani Nenkova and Rebecca Passonneau. Evaluating content selection in summarization: The pyramid method. In Daniel Marcu Susan Dumais and Salim Roukos, editors, *HLT-NAACL 2004: Main Proceedings*, pages 145–152, Boston, Massachusetts, USA, May 2 - May 7 2004. Association for Computational Linguistics. 116

[35] NIST. http://www.itl.nist.gov/iad/mig//tests/rt. 7

[36] Gerald Penn and X. Zhu. A critical reassessment of evaluation baselines for speech summarization. In *Proceedings of the ACL-HLT*, Columbus, OH, 2008. 59, 69, 71, 83

[37] Matthew Purver, Patrick Ehlen, and John Niekrasz. Shallow discourse structure for action item detection. In *Proceedings of the HLT-NAACL workshop 'Analyzing Conversations in Text and Speech'*, New York, NY, June 2006. 7

[38] Matthew Purver, Konrad Körding, Thomas Griffiths, and Joshua Tenenbaum. Unsupervised topic modelling for multi-party spoken discourse. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics (COLING-ACL)*, pages 17–24, Sydney, Australia, July 2006. Association for Computational Linguistics. 7, 10, 53, 56

[39] H. Raghavan, O. Madani, and R. Jones. Active learning with feedback on features and instances. In *The Journal of Machine Learning Research*, pages 1655–1686, 2006. 13

[40] K. Riedhammer, D. Gillick, B. Favre, and D. Hakkani-Tur. Packing the meeting summarization knapsack. In *Proceedings of the 9th International Conference of the ISCA (Interspeech 2008)*, pages 2434–2437, Brisbane, Australia, 2008. 80

[41] N. C. Romano and J. F. Nunamaker. Meeting analysis: Findings from research and practice. In *Proceedings of the 34^{th} Annual Hawaii International Conference on System Sciences*, Hawaii, 2001. 2

[42] I. Ruthven and M. Lalmas. A survey on the use of relevance feedback for information access systems. In *Knowledge Engineering Review, 18 (2)*, pages 95–145, 2003. 12

[43] Paul E. Rybski and Manuela M. Veloso. Using sparse visual data to model human activities in meetings. In *Workshop on Modeling Other Agents from Observations, International Joint Conference on Autonomous Agents and Multi-Agent Systems*, 2004. 7

[44] J. Ben Schafer, Joseph Konstan, and John Riedi. Recommender systems in e-commerce. In *EC '99: Proceedings of the 1st ACM conference on Electronic commerce*, pages 158–166, New York, NY, USA, 1999. ACM. 8

[45] J. Shen and T. G. Dietterich. Active em to reduce noise in activity recognition. In *2007 International Conference on Intelligent User Interfaces*, pages 132–140, Honolulu, HI, 2007. 11

[46] E. W. Stein and V. Zwass. Actualizing organizational memory with information systems. *Information Systems Research*, 6(2):127 – 137, June 1995. 6

[47] A. Stolcke, X. Anguera, K. Boakye, O. Cetin, A. Janin, M. Magimai-Doss, C. Wooters, and J. Zheng. The sri-icsi spring 2007 meeting and lecture recognition system. In *CLEAR 2007 and RT 2007, Springer Lecture Notes in Computer Science 4625*, pages 450–463, 2008. 7

[48] A. Stolcke, C. Wooters, N. Mirghafori, T. Pirinen, I. Bulyko, D. Gelbart, M. Graciarena, S. Otterson, B. Peskin, and M. Ostendorf. Progress in meeting recognition: The icsi–sri–uw spring 2004 evaluation system. In *NIST RT04 Meeting Recognition Workshop*, Montreal, 2004. 7

[49] G. Tur, A. Stolcke, L. Voss, J. Dowding, B. Favre, R. Fernandez, M. Frampton, M. Frandsen, C. Frederickson, M. Graciarena, D. Hakkani-Tür, D. Kintzing, K. Leveque, S. Mason, J. Niekrasz, S. Peters, M. Purver, K. Riedhammer, E. Shriberg, J. Tien, D. Vergyri, and F. Yang. The calo meeting speech recognition and understanding system. In *Proceedings of the IEEE Spoken Language Technology Workshop*, pages 69–72, Goa, India., 2008. 7

[50] Luis von Ahn, Ruoran Liu, and Manuel Blum. Peekaboom: a game for locating objects in images. In *CHI '06: Proceedings of the SIGCHI conference on Human Factors in computing systems*, pages 55–64, New York, NY, USA, 2006. ACM Press. 17

[51] A. Waibel, M. Bett, and M. Finke. Meeting browser: Tracking and summarizing meetings. In *Proceedings of the Broadcast News Transcription and Understanding Workshop*, pages 281–286, Lansdowne, VA, 1998. 6

[52] J. S. Weber and M. E. Pollack. Entropy-driven online active learning for interactive calendar management. In *Proceedings of the 2007 International Conference on Intelligent User Interfaces*, pages 141–150, Hawaii, USA, 2007. 9

[53] R. W. White, I. Ruthven, and J. M. Jose. A study of factors affecting the utility of implicit relevance feedback. In *Proceedings of the* 28^{th} *Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 35–42, Salvador, Brazil, 2005. 9

[54] Shasha Xie, Y. Liu, and H. Lin. Evaluating the effectiveness of features and sampling in extractive meeting summarization. In *Proceedings of the IEEE Workshop on Spoken Language Technology*, Goa, India, 2008. 73, 76

[55] H. Yu, T. Tomokiyo, Z. Wang, and A. Waibel. New developments in automatic meeting transcription. In *Proceedings of the International Conference on Spoken Language Processing*, Beijing, China, 2000. 7

[56] X. Zhu and G Penn. Summarization of spontaneous conversations. In *Proceedings of Interspeech 2006*, Pittsburgh, PA, 2006. 63