SmartNotes: Implicit Labeling of Meeting Data through User Note–Taking and Browsing

Satanjeev Banerjee
Language Technologies Institute
Carnegie Mellon University
Pittsburgh, PA 15213
banerjee@cs.cmu.edu

Alexander I. Rudnicky
School of Computer Science
Carnegie Mellon University
Pittsburgh, PA 15213
air@cs.cmu.edu

Abstract

We have implemented SmartNotes, a system that automatically acquires labeled meeting data as users take notes during meetings and browse the notes afterwards. Such data can enable meeting understanding components such as topic and action item detectors to automatically improve their performance over a sequence of meetings. The SmartNotes system consists of a laptop based note taking application, and a web based note retrieval system. We shall demonstrate the functionalities of this system, and will also demonstrate the labeled data obtained during typical meetings and browsing sessions.

1 Goals of the *SmartNotes* System

Most institutions hold a large number of meetings every day. Several of these meetings are important, and meeting participants need to recall the details of the discussions at a future date. In a previous survey (Banerjee et al., 2005) of busy professors at Carnegie Mellon University we showed that meeting participants needed to recall details of past meetings on average about twice a month. Performing such retrieval is not an easy task. It is time consuming; in our study participants took on average between 15 minutes to an hour to recall the information they were seeking. Further, the quality of the retrieval is dependent on whether or not the participants had access to the notes at the meeting. On a scale of 0 to 5, with 5 denoting complete satisfaction with retrieval results, participants reported a satisfaction of 3.4 when they did not have notes, and 4.0 when they did.

Despite the prevalence of important meetings and the importance of notes, there is a relative paucity of technology to help meeting participants take notes easily at meetings. Some commercial applications allow users to take notes (e.g. OneNote[1]) and even record audio/video (e.g. Quindi[2]), but no product attempts to automatically take notes. Our **long term goal** is to create a system that makes note–taking easier by performing tasks such as automatically highlighting portions of the meeting that are likely to be important to the user, automatically detecting "note–worthy" phrases spoken during the meeting, etc.

To perform such note taking, the system needs to form an understanding of the meeting. Our **short term goal** is to create a system that can detect the topics of discussion, the action items being discussed, and the roles of the meeting participants. Additionally, these components must adapt to specific users and groups of users since different people will likely take different notes at the same meeting. Thus we wish to implement the note taking system in such a way that the user's interactions with the system result in *labeled meeting data* that can then be used to adapt and improve the meeting understanding components.

Towards these goals, we have built *SmartNotes* which helps users easily record and retrieve notes.

[1] http://office.microsoft.com/onenote
[2] http://www.quindi.com

Proceedings of the Human Language Technology Conference of the NAACL, Companion Volume, pages 261–264,
New York City, June 2006.

The system also records the user interactions to form labeled meeting data that can later be used to automatically improve the meeting understanding components. In the next section we describe the meeting understanding components in more detail. Next we describe SmartNotes itself, and show how it is currently helping users take and retrieve notes, while acquiring labeled data to aid each of the meeting understanding components. Finally we end with a discussion of what functionality we plan to demonstrate at the conference.

2 Automatic Meeting Understanding

Topic detection and segmentation: We are attempting to automatically detect the topics being discussed at meetings. This task consists of two subtasks: discovering the points in a meeting when the topic changes, and then associating a descriptive *label* to the segment between two topic shifts. Our current strategy for topic shift detection (Banerjee and Rudnicky, 2006a) is to perform an edge detection using such features as speech activity (who spoke when and for how long), the words that each person spoke, etc. For labeling, we are currently simply associating the agenda item names recorded in the notes with the segments they are most relevant to, as decided by a tf.idf matching technique. Topic detection is particularly useful during meeting information retrieval; (Banerjee et al., 2005) showed that when users wish to retrieve information from past meetings, they are typically interested in a specific discussion topic, as opposed to an entire meeting.

Action item detection: An obvious application of meeting understanding is the automatic discovery and recording of action items as they are discussed during a meeting. Arguably one of the most important outcomes of a meeting are the action items decided upon, and automatically recording them could be a huge benefit especially to those participants that are likely to not note them down and consequently forget about them later on.

Meeting participant role detection: Each meeting participant plays a variety of roles in an institution. These roles can be based on their function in the institution (managers, assistants, professors, students, etc), or based on their expertise (speech recognition experts, facilities experts, etc). Our current strategy for role detection (Banerjee and Rudnicky, 2006b) is to train detectors on hand labeled data. Our next step is to perform discovery of new roles through clustering techniques. Detecting such roles has several benefits. First, it allows us to build prior expectations of a meeting between a group of participants. For example, if we know person A is a speech recognition expert and person B a speech synthesis expert, a reasonable expectation is that when they meet they are likely to talk about technologies related speech processing. Consequently, we can use this expectation to aid the action item detection and the topic detection in that meeting.

3 SmartNotes: System Description

We have implemented SmartNotes to help users take *multi–media notes* during meetings, and retrieve them later on. SmartNotes consists of two major components: The note taking application which meeting participants use to take notes during the meeting, and the note retrieval application which users use to retrieve notes at a later point.

3.1 SmartNotes Note Taking Application

The note taking application is a stand–alone system, that runs on each meeting participant's laptop, and allows him to take notes during the meeting. In addition to recording the text notes, it also records the participant's speech, and video, if a video camera is connected to the laptop. This system is an extension of the Carnegie Mellon Meeting Recorder (Banerjee et al., 2004).

Figure 1 shows a screen–shot of this application. It is a server–client application, and each participant logs into a central server at the beginning of each meeting. Thus, the system knows the precise identity of each note taker as well as each speaker in the meeting. This allows us to avoid the onerous problem of automatically detecting who is speaking at any time during the meeting. Further, after logging on, each client automatically synchronizes itself with a central NTP time server. Thus the time stamps that each client associates with its recordings are all synchronized, to facilitate merging and play back of audio/video during browsing (described in the next sub–section).

Once logged in, each participant's note taking

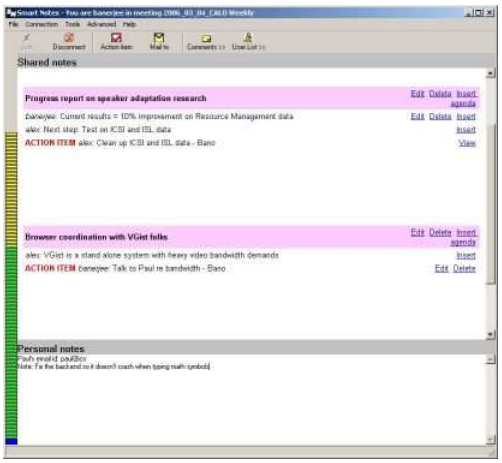

Figure 1: *Screen shot of the SmartNotes note–taking client*

area is split into two sections: a *shared* note taking area, and a *private* note taking area. Notes written in the shared area are viewable by all meeting participants. This allows meeting participants to share the task of taking notes during a meeting: As long as one participant has recorded an important point during a meeting, the other participants do not need to, thus making the note taking task easier for the group as a whole. Private notes that a participant does not wish to share with all participants can be taken in the private note taking area.

The interface has a mechanism to allow meeting participants to insert an agenda into the shared area. Once inserted, the shared area is split into as many boxes as there are agenda items. Participants can then take notes during the discussion of an agenda item in the corresponding agenda item box. This is useful to the participants because it organizes the notes as they are being taken, and, additionally, the notes can later be retrieved agenda item by agenda item. Thus, the user can access all notes he has taken in different meetings regarding "buying a printer", without having to see the notes taken for the other agenda items in each such meeting.

In addition to being useful to the user, this act of inserting an agenda and then taking notes within the relevant agenda item box results in generating (unbeknownst to the participant) **labeled data for the topic detection component**. Specifically, if we define each agenda item as being a separate "topic", and make the assumption that notes are taken approximately concurrent with the discussion of the contents of the notes, then we can conclude that there is a shift in the topic of discussion at some point between the time stamp on the last note in an agenda item box, and the time stamp on the first note of the next agenda item box. This information can then be used to improve the performance of the topic shift detector. The accuracy of the topic shift data thus acquired depends on the length of time between the two time points. Since this length is easy to calculate automatically, this information can be factored into the topic detector trainer.

The interface also allows participants to enter action items through a dedicated action item form. Again the advantage of such a form to the participants is that the action items (and thus the notes) are better organized: After the meeting, they can perform retrieval on specific fields of the action items. For example, they can ask to retrieve all the action items assigned to a particular participant, or that are due a particular day, etc.

In addition to being beneficial to the participant, the action item form filling action results in generating **labeled data for the action item detector**. Specifically, if we make the assumption that an action item form filling action is preceded by a discussion of the action item, then the system can couple the contents of the form with all the speech within a window of time before the form filling action, and use this pair as a data point to retrain its action item detector.

3.2 SmartNotes Note Retrieval Website

As notes and audio/video are recorded on each individual participant's laptop, they also get transferred over the internet to a central meeting server. This transfer occurs in the background without any intervention from the user, utilizes only the left–over bandwidth beyond the user's current bandwidth usage, and is robust to system shut–downs, crashes, etc. This process is described in more detail in (Banerjee et al., 2004).

Once the meeting is over and all the data has been transferred to the central server, meeting participants can use the SmartNotes multi–media notes retrieval system to view the notes and access the recorded audio/video. This is a web–based application that uses the same login process as the stand–along note

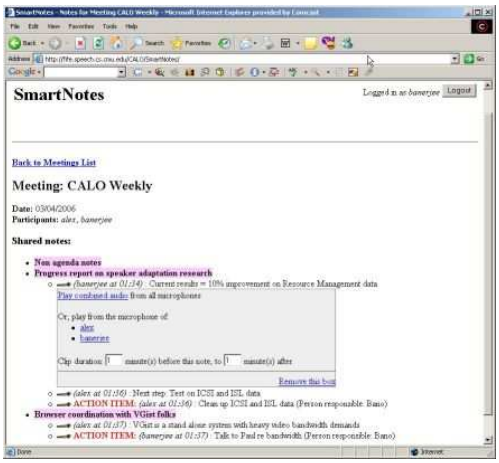

Figure 2: *Screen shot of the SmartNotes website*

taking system. Users can view a list of meetings they have recorded using the SmartNotes application in the past, and then for each meeting, they can view the shared notes taken at the meeting. Figure 2 shows a screen shot of such a notes browsing session. Additionally, participants can view their own private notes taken during the meeting.

In addition to viewing the notes, they can also access all recorded audio/video, indexed by the notes. That is, they can access the audio/video recorded around the time that the note was entered. Further they can specify how many minutes before and after the note they wish to access. Since the server has the audio from each meeting participant's audio channel, the viewer of the notes can choose to listen to any one person's channel, or a combination of the audio channels. The merging of channels is done in real time and is achievable because their time stamps have been synchronized during recording.

In the immediate future we plan to implement a simple key–word based search on the notes recorded in all the recorded meetings (or in one specific meeting). This search will return notes that match the search using a standard tf.idf approach. The user will also be provided the option of rating the quality of the search retrieval on a one bit satisfied/not–satisfied scale. If the user chooses to provide this rating, it can be used as a feedback to improve the search. Additionally, which parts of the meeting the user chooses to access the audio/video from can be used to form a model of the parts of the meetings most relevant to the user. This information can help the system tailor its retrieval to individual preferences.

4 The Demonstration

We shall demonstrate both the SmartNotes note taking client as well as the SmartNotes note–retrieval website. Specifically we will perform 2 minute long mock meetings between 2 or 3 demonstrators. We will show how notes can be taken, how agendas can be created and action items noted. We will then show how the notes and the audio/video from the 2 minute meeting can be accessed through the SmartNotes note retrieval website. We shall also show the automatically labeled data that gets created both during the mock meeting, as well as during the browsing session. Finally, if time permits, we shall show results on how much we can improve the meeting understanding components' capabilities through labeled meeting data automatically acquired through participants' use of SmartNotes at CMU and other institutions that are currently using the system.

References

S. Banerjee and A. I. Rudnicky. 2006a. A texttiling based approach to topic boundary detection in multi–participant conversations. Submitted for publication.

S. Banerjee and A. I. Rudnicky. 2006b. You are what you say: Using meeting participants' speech to detect their roles and expertise. In *Analyzing Conversations in Text and Speech Workshop at HLT–NAACL 2006*, New York City, USA, June.

S. Banerjee, J. Cohen, T. Quisel, A. Chan, Y. Patodia, Z. Al-Bawab, R. Zhang, P. Rybski, M. Veloso, A. Black, R. Stern, R. Rosenfeld, and A. I. Rudnicky. 2004. Creating multi-modal, user–centric records of meetings with the Carnegie Mellon meeting recorder architecture. In *Proceedings of the ICASSP Meeting Recognition Workshop*, Montreal, Canada.

S. Banerjee, C. Rose, and A. I. Rudnicky. 2005. 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.