Meeting Adjourned: Off-line Learning Interfaces for Automatic Meeting Understanding

Patrick Ehlen, Matthew Purver, John Niekrasz, Kari Lee and Stanley Peters
CSLI, Stanford University
Stanford CA 94305, USA
{ehlen, mpurver, niekrasz, karilee}@stanford.edu, peters@csli.stanford.edu

ABSTRACT

Upcoming technologies will automatically identify and extract certain types of general information from meetings, such as topics and the tasks people agree to do. We explore interfaces for presenting this information to users after a meeting is completed, using two post-meeting interfaces that display information from topics and action items respectively. These interfaces also provide an excellent forum for obtaining user feedback about the performance of classification algorithms, allowing the system to learn and improve with time. We describe how we manage the delicate balance of obtaining necessary feedback without overburdening users. We also evaluate the effectiveness of feedback from one interface on improvement of future action item detection.

INTRODUCTION

When people get together to share information, they often do so in discrete events called *meetings*. As everyone knows, these can be more or less successful depending on the quality of information people share and the decisions they make. Many factors conspire against their success: People may lack adequate prior information that is relevant to the meeting, or they may be biased toward information they already share, or they may interpret decisions or tasks differently, or they may forget items or otherwise fail to record them or record them in a disorganized way. Still, meetings manage to help keep the world turning, so we continue to have them, and also to produce tools meant to mitigate these counteracting factors as much as possible.

A good rule of thumb for such tools is that they should try not to require more effort than they supposedly relieve. That rule can be hard to live by when technology gets involved, especially technology that aims to create an automatic record of meetings, which abound with the rapid, overlapping, unstructured speech of multiple people who are not speaking with machine recognition in mind. Moreover, meetings can vary greatly in how ordered or chaotic they are, and there are few general principles that govern their content.

But nearly all meetings share a few commonalities: People converge to discuss a certain set of topics, to make decisions related to those topics, and they often seek to agree on a mutual perspective of some imagined future, as well as the actions that need to be carried out in order to accomplish that future. Our efforts of late have attempted to exploit those commonalities with the goal of producing tools that may help to make meetings more productive and their outcomes more enduring; in particular, producing automated representations of the topics, decisions, and future actions that people discuss.

Since the content of meetings can be complicated, unstructured, and errorful, producing interfaces to these tools that follow the rule of thumb mentioned above is a tricky proposition. These interfaces must first and foremost be able to detect useful information from the maelstrom of signals a meeting provides. It must then represent that information in some intuitive way that makes sense to people. And since it will undoubtedly make many errors, those errors should be easily corrected or ignored by a user. It should also learn as much information as it can from the normal actions of that user, soliciting feedback in a way that is as close to invisible as possible. Finally, that feedback should prove effective at helping the automatic detection process to hone in on more relevant information, and should ideally learn to adapt to each person's style of working and to identify the information they care about most.

We present some ideas here for interfaces that present automatically detected *topics* and future actions or *action items*, methods of soliciting user feedback from those representations, and an evaluation of the quality of feedback solicited from one such interface that allows the participants from a meeting to add automatically-detected action items to their to-do lists.

Background

Recent advances in technologies that integrate speech processing, natural language understanding, vision, and multimodal interaction have led to several research projects that aim to produce tools that perceive what happens at a meeting, extract salient events, and produce a reliable record. The

ICSI Meeting Project [11, 12] produced automated and segmented transcripts from natural, multi-party speech during meetings, while the ISL Smart Meeting Room Task [20] and the M4 and AMI projects [14] instrumented meeting rooms to collect data on behaviors so the interactions of meeting participants could be analyzed to produce flexible records of their activities, while providing a supportive environment for collaboration.

As part of the DARPA CALO Meeting Assistant project, we have focused on data about the behaviors of people in meetings, assimilating speech, movement, and note-taking data to create a rich representation of a meeting that can be analyzed and reviewed at many levels. The CALO Meeting Assistant integrates its observations with those of a larger system of agents, which assesses meeting data in the context of the ongoing projects and workflow for each meeting participant. Thus, our meeting assistant aims to reach beyond an intelligent room that understands only the activities of people in meetings, and attempts to understand their overarching concerns and interpret their behaviors from the perspective of what their meetings mean to them.

Existing tools to view meeting data focus on rapid information retrieval by users, and are designed to optimize the speed and success of finding answers to a broad class of queries. But the lack of linguistic metadata available to them results in a focus on the playback of signals rather than the display of dialogical or semantic features of the meeting [13, 19].

Because our aim in the CALO Meeting Assistant project is to automatically extract useful information such as the action items and topics discussed during meetings, our approach to information presentation has a somewhat different goal. Not only do we need interfaces for users to browse audio, video, notes, transcripts, and artifacts of meetings, we also need an interface that selectively presents automatically extracted information from our algorithms in a convenient and intuitive manner. And it should also provide each user with the opportunity to modify or correct information when automated recognition falls short of the mark, preferably with as little burden as possible.

In fact, compelling users to provide feedback is essential, since our agents maintain multiple lexical and semantic hypotheses, and then rely on feedback from users about which hypotheses sound reasonable. Getting that feedback is not easy, but we explore two possibilities here: one that supports topic detection, and one that supports action item detection.

INTERFACE FOR TOPICS

Topic Detection

We can see the problem of topic detection as two sub-problems: *segmentation*, dividing the meeting into topically coherent segments, and *identification*, producing some description or representation of the topics discussed therein. These two problems could be approached separately. For example, a meeting can be segmented into likely topical areas using features indicative of topic shifts—such as cue phrases, speaker activity shifts, and changes in vocabulary—without prior knowledge about what those topics should look like [6, 2, 7]. Once segmented, identification could then proceed by classifying the segments according to some predefined list of topics [10] or an agenda list [3].

However, separating the processes of segmentation and identification has some probelms. For one, the notion of a topic can be highly subjective, since different people often come away from a meeting with differing perspectives on what the meeting was about and what topics were discussed. Even attempts made by objective readers to segment meetings into topics show wide variation and poor inter-annotator agreement on topic boundary placement, especially as the definition of "topic" used becomes more fine-grained [8].

Of course, the presence of an established agenda divided into topical subject areas can help achieve consensus; but in the absence of such information we may find that a particular user's ideal segmentation depends on their perspective on which topics are important. We therefore prefer an approach which treats segmentation and identification as joint problems, and attempts to solve them simultaneously by joint inference. We also prefer one which can easily be adapted to individual users and their individual topics of interest.

Generative Topic Model

We represent topics as probability distributions over words. We then model the meeting discourse as though it were generated by a set of underlying unseen topics, with each topically coherent discourse segment corresponding to a particular fixed weighted mixture of those topics. We use a variant of Latent Dirichlet Allocation [4], to learn a set of underlying topics jointly with a most likely segmentation (see Figure 1). Segmentation accuracy rivals that of other methods, while human judges rate the topics themselves highly on a coherence scale (see [17] for details on our topic detection method).

The learning is initially unsupervised, learning topics over multiple meetings and storing them in a central topic pool. This approach can be personalized to a user by learning topics only over meetings in which that user participates or perhaps which have been marked as "interesting." Moreover, this algorithm lends itself to further personalization and learning through supervision: If users were to provide feedback from some suitable interface on the topics or segments of interest, the learning can be constrained, producing a personalized topic model.

User Adaptation

How can such feed back be obtained? One potential method is to observe and make use of user interaction with some topic presentation interface. This could happen during the meeting, perhaps by providing users with note-taking software that monitors note-taking behavior and uses that behavior as potential feedback that can be applied to learning topics in the future [1]. However, that method only works if at least one person is typing notes "on-line" during the meeting, which might require more cognitive effort than meet-

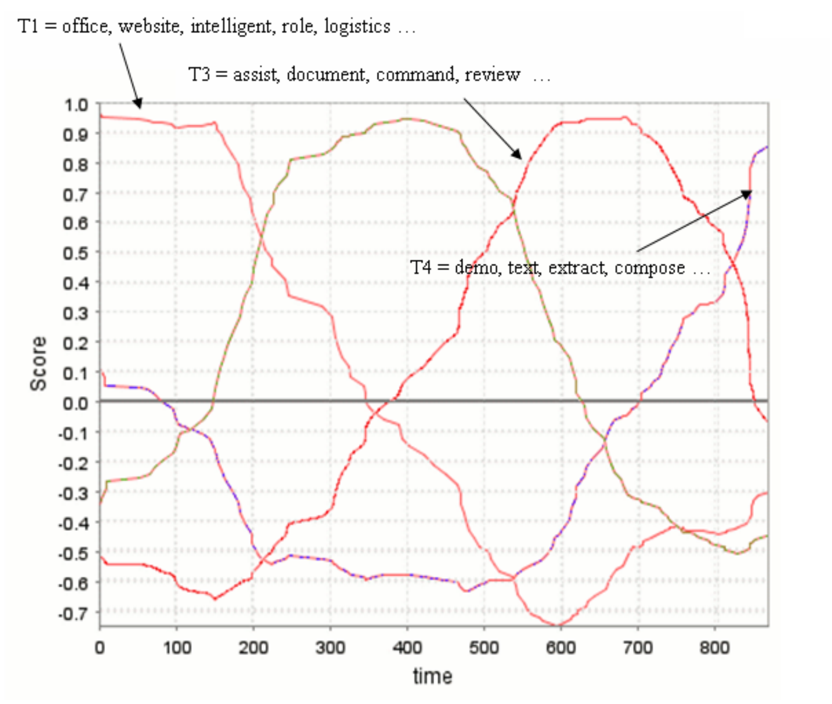


Figure 1. A meeting segmented into topic areas

ing participants are consistently willing to provide. An alternate interface that allows users to browse content after the meeting might provide better opportunities for obtaining user feedback.

Such an off-line approach would involve giving users post-meeting topic information and allowing feedback via useful interaction. Since we define topics as lists of weighted words that are probable over some period of time, the two dimensions where feedback could be applied would be on the weightings of those words as they relate to the topic, and on the timing of the topic boundaries, as graphed in Figure 1. But the average person doesn't think about topics in this way, so these dimensions do not immediately lend themselves to a clear method of presentation.

Topic Presentation

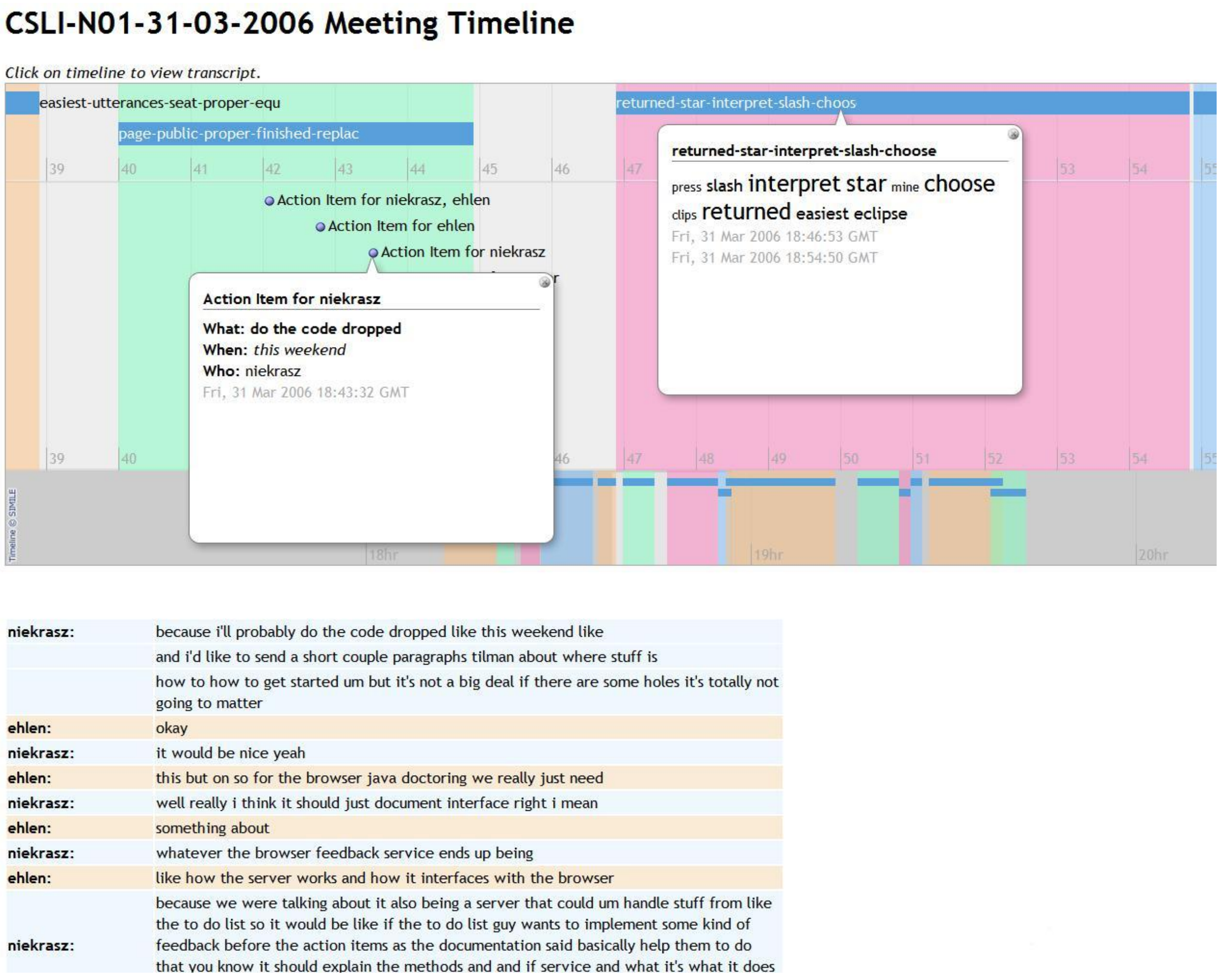
A solution we devised was to represent a meeting as a series of topical discussions on a timeline, using the MIT Simile Timeline widget [18]. As can be seen in Figure 2, the timeline consists of two horizontal bands: The bottom band is a birds-eye view of the entire meeting, while the top band shows minute-by-minute details. Segments of topic discussions are represented as regions shaded in different colors. In the birds-eye view, the meeting is viewed as a series of topics, and clicking within a topic will scroll the detailed view to that segment of the meeting. At the same time, a transcript panel below the timeline also scrolls the ASR trancsript to the selected point of the meeting, and begins to play audio from that point as well.

To represent the actual "content" of each topic, a blue band at the top displays the "topic title," represented as the five most characteristic keywords for this topic.[1] Clicking on the topic title displays the topic words in a word cloud, where words with higher relevance to the topic under discussion are displayed in larger font.

There are several possible ways to obtain useful feedback about topics from a timeline interface such as this. A user could be allowed to change topic boundaries by moving the edges of shaded topic regions, providing better definition for when discussion of a particular topic begins and ends. This user feedback would then provide implicit supervision for determining better overall or user-specific segmentation, and future runs of the generative topic model could be constrained to produce segmentations that are more consistent with user-defined boundaries. Users could also use manual topic shading to "mark & tag" topic areas they may wish to revisit. For instance, a user might highlight a certain segment of the meeting and apply a user-defined tag like "Q4 budget." With enough feedback of this kind, the system could then learn to associate these tags with particular probability distributions, and produce them automatically for generated topics in future

Word clouds also provide a potential source of feedback, if users are allowed to change what they see as the relative salience of words by upgrading or downgrading them. If a topic discussion about the budget shows the word "budget" in small font, the user could drag the word upwards to make it larger, thus providing feedback for the weightings of

[1]Keywords are extracted from the topic's underlying word probability distribution, by finding those words for which the probability associated with this topic exceeds that assigned by a general topic-independent distribution by the greatest amount.


CSLI-N01-31-03-2006 Meeting Timeline

Click on timeline to view transcript.

Action Item for niekrasz, ehlen

Action Item for ehlen

Action Item for niekrasz

Action Item for niekrasz

What: do the code dropped
When: *this weekend*
Who: niekrasz
Fri, 31 Mar 2006 18:43:32 GMT

returned-star-interpret-slash-choose

press slash interpret star mine choose clips returned easiest eclipse
Fri, 31 Mar 2006 18:46:53 GMT
Fri, 31 Mar 2006 18:54:50 GMT

niekrasz:	because i'll probably do the code dropped like this weekend like
	and i'd like to send a short couple paragraphs tilman about where stuff is
	how to how to get started um but it's not a big deal if there are some holes it's totally not going to matter
ehlen:	okay
niekrasz:	it would be nice yeah
ehlen:	this but on so for the browser java doctoring we really just need
niekrasz:	well really i think it should just document interface right i mean
ehlen:	something about
niekrasz:	whatever the browser feedback service ends up being
ehlen:	like how the server works and how it interfaces with the browser
niekrasz:	because we were talking about it also being a server that could um handle stuff from like the to do list so it would be like if the to do list guy wants to implement some kind of feedback before the action items as the documentation said basically help them to do that you know it should explain the methods and and if service and what it's what it does

easiest-utterances-seat-proper-equ
page-public-proper-finished-replac
returned-star-interpret-slash-choos
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
18hr
19hr
20hr
Timeline © SIMILE

Figure 2. Timeline view of meeting showing automatically detected action items, regions shaded according to topic segmentation, word clouds reflecting topic word salience, and ASR-produced transcript segmented by dialogue acts.

words in that topic for later retraining.

INTERFACE FOR ACTION ITEMS

Action Item Detection

When people discuss the action items they intend to carry out, they use an interactive and collaborative process. Tasks and their details are defined, and commitment to them is established, not through individual utterances in isolation but through sequences of utterances that play particular roles in an evolving game. Thus to automatically detect action items, we rely on a shallow notion of dialogue structure; but because data from ASR transcripts of freeform meetings are highly variable and errorful, rule-based methods for inferring such structure do not yield good results, calling for a more robust approach. Our "hierarchical" approach to action item detection recognizes that discussions of commitments to future actions are characterized by the probable co-occurence of certain types of dialogue moves which can be individually detected. Clusters of these moves that appear within some window of time may then indicate probable action item discussions. (We designed this hierarchical approach specifically to overcome the complexities of dialogue, which make it unsuitable for "flat" approaches such as that implemented for textual sources like e-mail [5].)

Thus, detection is a two-stage process. In the first stage, we look for utterances that play the relevant roles in the commitment process, which can be thought of as a set of action-item-specific dialogue moves: *task description* (proposal or discussion of the task to be performed), *timeframe* (proposal or discussion of when the task should be performed), *ownership* (assignment or acceptance of responsibility by one or more people), and *agreement* (commitment to the action item as a whole or to one of its details). This is implemented using independent utterance classifiers (*"subclassifiers"*—linear-kernel support vector machines trained from manual annotations), each of which is trained to detect whether an utterance might represent one of the four action-item-related dialogue moves listed above.

The second stage involves a single subdialogue classifier (the *"superclassifier"*), which analyzes clusters of these patterns of tagged utterances and their confidence scores to hypothesize windows of *action item discussion* (see Figure 3). When these clusters indicate a hypothesized action item discussion, the relevant utterances that were previously tagged by the subclassifiers are fed into a summarization algorithm that attempts to extract details for the action item—the who, what, and when of the action—and render them in a format suitable for presentation in a user interface. Because there are often several utterances that convey these details, these utterances are ranked according to their classification score, allowing alternate hypotheses of details to be available to the end-user. For more discussion of the classification and summarization processes, see [15].

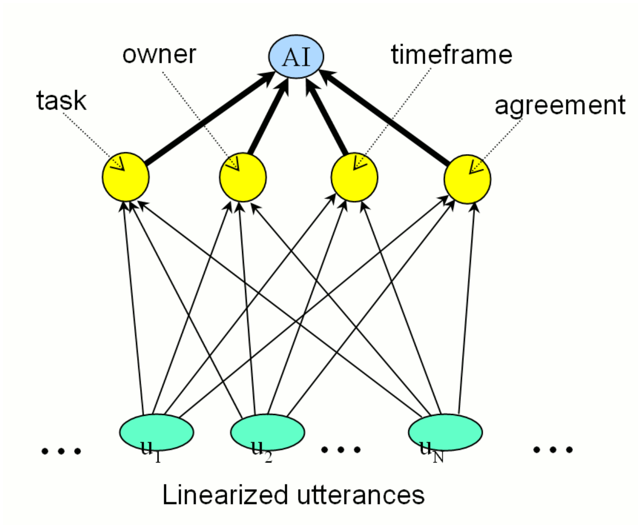


Figure 3. Defining an action item by classifying multiple utterances with dialog subclasses

Action Item Feedback

Figure 4 shows a broad view of our architecture for using implicit supervision from users to improve classifier performance. From the interface, a user's interactions with the summarized action items can be interpreted, providing feedback to a feedback interpreter that updates the hypothesized action items and utterance tags, which are ultimately treated as annotations that provide new training data for the classifiers.

Action Item Presentation and User Feedback Mechanisms

So, as can be seen in see Figure 3, our action item detector consists of not one classifier but *five*, each of which could stand to improve from feedback about its performance. However, when keeping in mind our rule of thumb not to create interfaces that are more effort than they're worth, obtaining good feedback becomes a tall order.

Subclassifier Feedback

Our solution was to use implicit supervision from users by creating an interface that sends each meeting participant a list of the action items it detected, allowing each user to add action items to their to-do lists, along with the supporting details. As can be seen in Figure 5, action items detected from a meeting are presented in rows, and each lists supporting details of "What to do," "When to do it" and "Who should do it." Each of these fields corresponds to information associated with one of the dialogue act types used in detection, and therefore to the output of one of the subclassifiers (with the exception of the *agreement* classifier). Feedback on the individual properties can therefore be used to provide implicit supervision for the corresponding subclassifiers.

Under the "What to do" and "When to do it" fields, we present a descriptive phrase extracted from the top utterance the classifier finds most relevant to the field (for example, an utterance "I'll get back to you on that tomorrow" might provide the phrase "tomorrow" under the "When to do it" field). By single-clicking on the field, a user sees a list of the top six most likely phrases, any of which can replace the current description with a single click. Users may also double-click the description and edit or retype the field if they they wish to do so before the action item is added to their to-do list. Allowing the user to edit these results is essential when using ASR, which often substitutes proper names or acronyms with similar-sounding words from the lexicon; allowing the user to edit these allows acquisition of new vocabulary that can later be included in the language model.

Extracting the "Who should do it" information is slightly more complex. When people assign or agree to do tasks in a meeting, they rarely use one another's names, instead referring to one another using pronouns as in "Can you do that?" or "Yes, I'll do that." For this reason, displaying the most relevant utterance picked by the classifier rarely provides information that's actually useful to the user (unless the user can remember who "you" or "I" was referring to). For this reason, we perform an extra step of reference resolution on such utterances (see [9]), and display a hypothesis about which person or persons agreed to do the task. These can be displayed as a list of ordered hypotheses that can be selected, though it is also useful to provide the names of the other participants in the meeting, and also an "unassigned" option.

Turning interaction into useful data for classifier re-training now requires a further step: inferring a set of training data in which utterances are tagged as positive or negative instances. If a hypothesized description is accepted unchanged, this is straightforward: the utterance(s) from which the hypothesis was made are marked as positive instances. If a description is changed to one of the alternatives provided, the utterance(s) associated with the chosen alternative are marked as positive instances, and all others as negative. When users elect to edit or retype the field, we must do more in order to find the correct utterances to mark as positive: our current method is to search for possible utterances near the area of action item discussion which are lexically close to the given description (for the what/when properties) or referentially consistent with it (for the who property).

Superclassifier Feedback

In addition to these subclassifier feedback mechanisms, a fourth field ("My Actions" in Figure 5) shows buttons that solicit feedback for the superclassifier, using the assump-

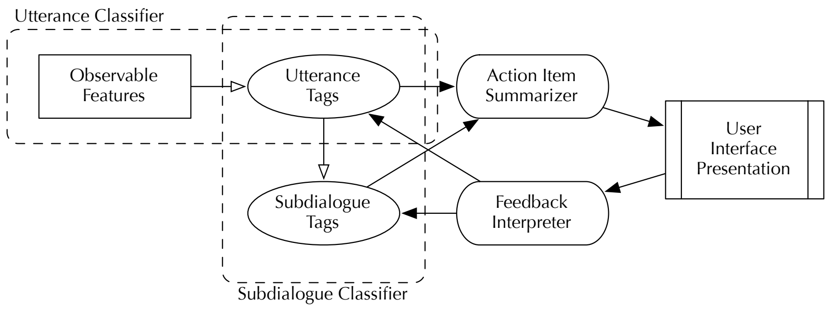


Figure 4. An outline of the action item detection and feedback system.

tion that users will only want to add valid, reasonably-well-formed descriptions of tasks to their to-do list. In doing so, they provide information about how the superclassifier fared on each detected action item.

One complication, however, is that effective feedback in this case requires not two but *three* possible user actions in response to an action item. These actions can be roughly summarized as "This is a good action item," "This is not an action item" and "This is an action item but delete it anyway." The latter two require an important distinction to be clear for the user. If an action item has been correctly detected and the user chooses "Add action item to to-do list" (shown as a thunderbolt icon), we want to record this as an instance of positive feedback for the superclassifier, and also to mark each utterance selected in the property fields (What, When and Who) as instances of positive feedback. So the case of positive feedback is straightforward.

However, when an action item is deleted, there is no way to distinguish a priori whether the user is deleting it because it is a bad instance of an action item—which should result in negative feedback—or if the action item is being deleted because the user doesn't want to add it for some reason (e.g., the action item is assigned to someone else, or already has it on the to-do list, or wishes to ignore it)—which should not result in negative feedback, since the action item is still presumably valid.

To capture this distinction, we took inspiration from a similar solution used in e-mail to assist in spam detection. An e-mail message offers a similar 3-way choice: For valid e-mail messages, either the user wishes to keep it, or to delete it for whatever reason. But the user is also informed that a third option exists—the "spam" button—that will mark certain types of e-mail messages as spam, with the aim of helping classification and hopefully reducing the future number of spam messages the user receives.

In a similar vein, we included a "junk" or "This is not an action item" button, with which a user can mark a potential action item as a bad selection, and in doing so provide negative feedback on classifier choices that should be discounted in future retraining. Figure 5 shows the "junk" button as a thunderbolt with an "X" through it, distinguishable from the third choice of deleting an action item for any other reason, represented by an icon of a thunderbolt going into a recycle bin. This method of obtaining negative feedback instances asks the user to exchange a small amount of effort (pressing a button) with the promise of less effort in the future as the classifiers improve at distinguishing valid action item utterances from invalid ones.

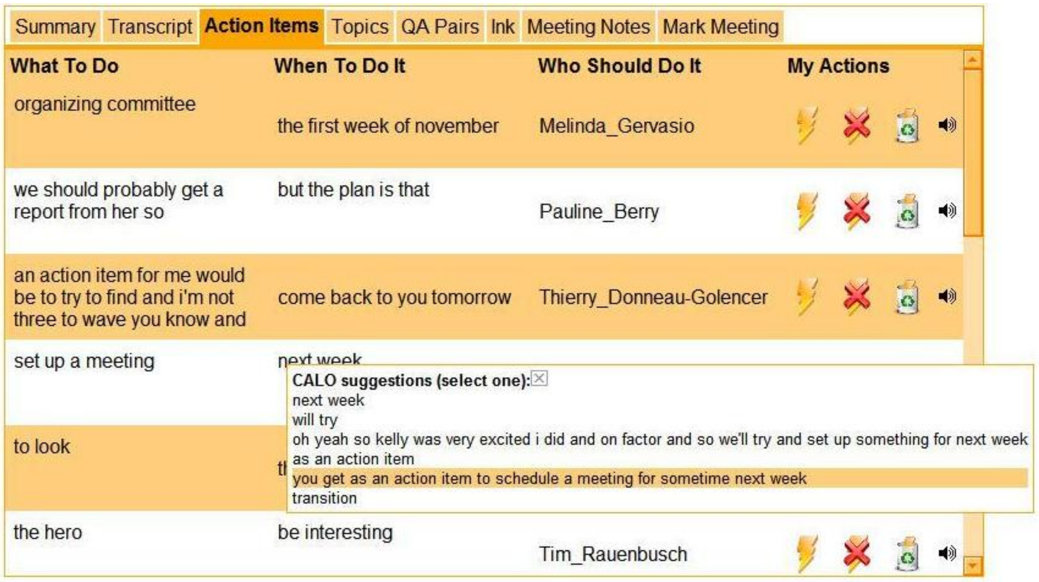
Action Item User Review Process

To provide an overall picture, each user's review process proceeds as follows: After a meeting has finished and an ASR transcript is generated along with hypothesized topic segments and action items, the user receives an e-mail indicating the meeting data is available for review. By following the e-mailed link the user will see a list of detected action items, as shown in Figure 5. For the action items that look valid, the user may select different descriptive utterances or manually edit the "What" and "When" fields, followed by pressing the "Add action item" button, which causes the action item to disappear from the detected items list and reappear in the user's to-do list, at the same time sending positive feedback for the subclassifiers and the superclassifier.

For the remaining action items, the user can press a button to listen to audio for the surrounding segment of the meeting if think these may still contain useful reminders or actions. If not, they can be deleted, "junked," or simply ignored. "Junked" action items will provide negative feedback, while deleted or ignored ones will not.

Action Item Feedback Evaluation

With these feedback mechanisms in place, the question remains of how effective that feedback will be at retraining classifiers, leading to improved action item detection in the future. Because the DARPA CALO program requires an annual evaluation, this gave us an opportunity to test that question.



What To Do	When To Do It	Who Should Do It	My Actions
organizing committee	the first week of november	Melinda_Gervasio	
we should probably get a report from her so	but the plan is that	Pauline_Berry	
an action item for me would be to try to find and i'm not three to wave you know and	come back to you tomorrow	Thierry_Donneau-Golencer	
set up a meeting	next week		
to look	tl		
the hero	be interesting	Tim_Rauenbusch	

CALO suggestions (select one):
next week
will try
oh yeah so kelly was very excited i did and on factor and so we'll try and set up something for next week
as an action item
you get as an action item to schedule a meeting for sometime next week
transition

Figure 5. Meeting browser action item feedback screenshot.

DARPA CALO CLP Evaluation Data

As part of the DARPA CALO evaluation, CALO components are put to the test during a "Critical Learning Period" (CLP) where evaluators use the CALO components as part of their everday work cycle. The evaluation took place at SRI International over a week-long period, with 18 SRI researchers using the components on a daily basis. As part of this, the participants held daily meetings and were trained to use system components in the same way a naive end-user might do. This process resulted in 18 meetings, divided into four sequences attended by different groups with different areas of expertise. Each sequence consists of 4 or 5 meetings, with most of the same participants, discussing related topics.

As part of the post-meeting review process, many (though not all) participants provided feedback on detected action items using the interface described above. The raw action item hypotheses which they saw were produced using a previously annotated set of fully supervised training data, taken from meetings recorded in the previous year; these meetings were held at various CALO institutions and involved several different participants and discussion topics. We will refer to this initial set of training data as Supervised Data (S). The feedback produced from user interaction was then used to infer a new set of training data, which we will refer to here as Feedback Data (F). We then tested the effect of this feedback data on classifier improvement, to evaluate whether a feedback-retrained classifier would perform closer to human annotations of action items than one trained only on the initial fully supervised set.

Baseline vs. Feedback Comparison

To provide a gold standard for comparison, we manually annotated the final meeting in each sequence for action items and their properties, using the annotation criteria described in [16]. We can then directly compare the performance of (a) the baseline system trained only on set S, with (b) a system trained on set S plus the data in set F for the relevant meeting sequence. Note that as we are evaluating on the final meeting in each sequence, we train on the feedback data from all meetings in that sequence except the final one. Performance improvement over the baseline system indicates an effective feedback mechanism.

Performance is shown in terms of precision, recall and F-score for the overall task of detecting action item discussions. Precision tell us how many of the system's hypotheses overlapped a gold standard action item discussion; recall tells us how many of the gold standard action item discussions were detected; and F-score is the harmonic mean of precision and recall. Note that our definition of a match is lenient: we do not require that all consituent utterances are exactly identified, but that a discussion is identified which overlaps with a gold-standard action item discussion. Table 1 shows the absolute figures for both systems; Table 2 shows the improvement deltas (in terms of absolute score difference and in terms of error reduction (relative reduction in the difference between the score and 1.00)).

Table 1. Absolute performances: baseline vs. retrained

	Baseline (S)			Retrained (S+F)		
	Rec	Pre	F1	Rec	Pre	F1
Seq 1 (OP)	0.50	1.00	0.67	0.63	1.00	0.77
Seq 2 (PE)	0.27	0.60	0.37	0.45	0.57	0.51
Seq 3 (SU)	0.67	0.60	0.63	0.22	0.33	0.27
Seq 4 (SY)	0.67	0.67	0.67	0.78	0.71	0.74

The results show that sizeable improvements are achieved in 3 of the 4 meeting sequences, with error reductions from 20 to 40%, which gives us encouragement that our interface and approach to implicit supervision can provide real bene-

Table 2. Performance improvement with feedback

	Absolute			Error Reduction		
	Rec	Pre	F1	Rec	Pre	F1
Seq 1 (OP)	0.13	0.00	0.10	26%	0%	30%
Seq 2 (PE)	0.18	*-0.03*	0.14	25%	*-8%*	22%
Seq 3 (SU)	*-0.45*	*-0.27*	*-0.36*	*-136%*	*-68%*	*-97%*
Seq 4 (SY)	0.11	0.04	0.07	33%	12%	21%

fits. However, performance was badly negatively affected in sequence 3 (SU).

Closer examination of the raw user feedback data revealed some unusual feedback behavior for a user set of training data for that sequence, perhaps indicating some technical difficulties when providing feedback (e.g., providing 38 instances of feedback for a single action item, where only four should have been possible, some of which was contradictory). So there is clearly some variability in feedback behavior, and we need to investigate the nature of this variability and its impact on overall performance.

In general, note that these results are preliminary; we now inted to investigate whether performance can be further improved, and negative impacts reduced, by modifying the training data inference methods. As the methods we currently use to infer training data from feedback are relatively naive (e.g. confirmations are taken as supplying positive feedback for all properties, no matter how low the initial classifier confidence from which they were produced; alternative candidates for utterances are selected using only simple lexical distance metrics), we believe that improvements are there to be made.

CONCLUSION

We presented some methods for display of automatically detected and abstracted meeting data, in particular the *topics* and *action items* discussed by meeting participants. While some abstracted data does not lend itself well to useful presentation to end users, we believe topic information can be displayed effectively in a timeline format that allows topic discussions to be viewed as colored bands throughout the meeting. This way meetings can be viewed from a birds-eye view as collections of topical discussions, and topic discussion boundaries can be redefined or tagged by the user, providing data for future retraining and classification. In addition, topics presented as word clouds allow users to deliver human feedback on the relative salience of terms in each topic.

For action items detected during a meeting, we described an interactive list with GUI actions that allow the user to fine-tune the *who*, *what* and *where* of an action item before adding it to a to-do list, harvesting these user actions as feedback data. An analysis of feedback data from the DARPA CALO 2007 CLP sessions shows that such feedback have the potential to be quite useful in retraining the system to offer action items in line with human selected ones. With the amount of feedback we used here, this process appears to be somewhat delicate, which may wash out with more feedback data; we intend to investigate that further.

In the future, we will explore whether similar methods of obtaining feedback from the timeline interface would be useful for retraining on topic segmentation and identification. We are also engaged in research on the detection of decsions, and intend further research into how those might be displayed on a timeline, or otherwise. Finally, we're curious how online methods of feedback—in which meeting participants provide feedback during the meeting rather than after—compare with the methods described herein, and how these methods compare to ordinary in-meeting note-taking.

REFERENCES

1. Satanjeev Banerjee and Alex Rudnicky. Using simple speech-based features to detect the state of a meeting and the roles of the meeting participants. In *Proceedings of the 8th International Conference on Spoken Language Processing (INTERSPEECH - ICSLP)*, 2004.

2. Satanjeev Banerjee and Alexander Rudnicky. A TextTiling based approach to topic boundary detection in meetings. In *Proceedings of the 9th International Conference on Spoken Language Processing (INTERSPEECH - ICSLP)*, Pittsburgh, Pennsylvania, September 2006.

3. Satanjeev Banerjee and Alexander Rudnicky. Segmenting meetings into agenda items by extracting implicit supervision from human note-taking. In *Prooceedings of the International Conference on Intelligent User Interfaces (IUI'07)*, Honolulu, Hawaii, January 2007. ACM.

4. David Blei, Andrew Ng, and Michael Jordan. Latent Dirichlet allocation. *Journal of Machine Learning Research*, 3:993–1022, 2003.

5. Simon Corston-Oliver, Eric Ringger, Michael Gamon, and Richard Campbell. Task-focused summarization of email. In *Proceedings of the 2004 ACL Workshop Text Summarization Branches Out*, 2004.

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

7. Maria Georgescul, Alexander Clark, and Susan Armstrong. Exploiting structural meeting-specific features for topic segmentation. In *Actes de la 14ème Conférence sur le Traitement Automatique des Langues Naturelles*, Toulouse, France, June 2007. Association pour le Traitement Automatique des Langues.

8. Alexander Gruenstein, John Niekrasz, and Matthew Purver. Meeting structure annotation: data and tools. In *Proceedings of the 6th SIGdial Workshop on Discourse and Dialogue*, Lisbon, Portugal, September 2005.

9. Surabhi Gupta, John Niekrasz, Matthew Purver, and Daniel Jurafsky. Resolving “you” in multi-party dialog. In *Proceedings of the 8th SIGdial Workshop on Discourse and Dialogue*, Antwerp, Belgium, September 2007.

10. Pei-Yun Hsueh and Johanna Moore. Automatic topic segmentation and labeling in multiparty dialogue. In *Proceedings of the 1st IEEE/ACM Workshop on Spoken Language Technology (SLT)*, Palm Beach, Aruba, 2006.

11. A. Janin, D. Baron, J. Edwards, D. Ellis, D. Gelbart, N. Morgan, B. Peskin, T. Pfau, E. Shriberg, A. Stolcke, and C. Wooters. The ICSI meeting corpus. In *Proceedings of the 2003 International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, April 2003.

12. Adam Janin, Jeremy Ang, Sonali Bhagat, Rajdip Dhillon, Jane Edwards, Javier Marcías-Guarasa, Nelson Morgan, Barbara Peskin, Elizabeth Shriberg, Andreas Stolcke, Chuck Wooters, and Britta Wrede. The ICSI meeting project: Resources and research. In *Proceedings of the 2004 ICASSP NIST Meeting Recognition Workshop*, 2004.

13. Konstantinos Koumpis and Steve Renals. Content-based access to spoken audio. *IEEE Signal Processing Magazine*, 22(5):61–69, 2005.

14. Anton Nijholt. Meetings, gatherings and events in smart environments. In S. Spencer, editor, *Proceedings of VRCAI 2004: ACM SIGGRAPH International Conference on Virtual-Reality Continuum and its Applications in Industry*, pages 229–232, Singapore, June 2004. ACM.

15. Matthew Purver, John Dowding, John Niekrasz, Patrick Ehlen, Sharareh Noorbaloochi, and Stanley Peters. Detecting and summarizing action items in multi-party dialogue. In *Proceedings of the 8th SIGdial Workshop on Discourse and Dialogue*, Antwerp, Belgium, September 2007.

16. Matthew Purver, Patrick Ehlen, and John Niekrasz. Detecting action items in multi-party meetings: Annotation and initial experiments. In Steve Renals, Samy Bengio, and Jonathan Fiscus, editors, *Machine Learning for Multimodal Interaction: Third International Workshop, MLMI 2006, Revised Selected Papers*, volume 4299 of *Lecture Notes in Computer Science*, pages 200–211. Springer, 2006.

17. 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.

18. SIMILE. Semantic Interoperability of Metadata and Information in unLike Environments. Available from http://simile.mit.edu/timeline/, 2007.

19. Simon Tucker and Steve Whittaker. Accessing multimodal meeting data: Systems, problems and possibilities. In Samy Bengio and Hervé Bourlard, editors, *Machine Learning for Multimodal Interaction: First International Workshop, MLMI 2004, Revised Selected Papers*, volume 3361 of *Lecture Notes in Computer Science*, pages 1–11. Springer, 2005.

20. Alex Waibel, Tanja Schultz, Michael Bett, Matthias Denecke, Robert Malkin, Ivica Rogina, Rainer Stiefelhagen, and Jie Yang. SMaRT: The smart meeting room task at ISL. In *International Conference on Acoustic, Speech, and Signal Processing (ICASSP)*, pages 752–755, Hong Kong, April 2003.