Automatic Annotation of Dialogue Structure from Simple User Interaction

Matthew Purver, John Niekrasz, and Patrick Ehlen

CSLI, Stanford University, Stanford CA 94305, USA
{mpurver,niekrasz,ehlen}@stanford.edu
http://godel.stanford.edu/

Abstract. In [1,2], we presented a method for automatic detection of *action items* from natural conversation. This method relies on supervised classification techniques that are trained on data annotated according to a hierarchical notion of dialogue structure; data which are expensive and time-consuming to produce. In [3], we presented a meeting browser which allows users to view a set of automatically-produced action item summaries and give feedback on their accuracy. In this paper, we investigate methods of using this kind of feedback as implicit supervision, in order to bypass the costly annotation process and enable machine learning through use. We investigate, through the transformation of human annotations into hypothetical idealized user interactions, the relative utility of various modes of user interaction and techniques for their interpretation. We show that performance improvements are possible, even with interfaces that demand very little of their users' attention.

1 Introduction

Few communicative events in a working day are more important than group decisions committing to future action. These events mark concrete progress toward shared goals, and are the bread and butter of face-to-face meetings. However, information produced in conversation is frequently forgotten or mis-remembered due to the limited means of memory, attention, and supporting technologies. Organizations are unable to review their own internal decisions, and individuals forget their own commitments. This information loss seriously impacts productivity and causes enormous financial hardship to many organizations [4].

The primary objective of our research is to assist meeting participants by automatically identifying *action items* in meetings: public commitments to perform some concrete future action. Some related work has sought to classify individual utterances or sentences as action-item-related [5,6,7,8,9]. These approaches have limited success when applied to multi-party conversational speech: individual utterances often do not contain sufficient information, as commitment arises not through utterances in isolation but through the dialogue interaction as a whole.

In contrast, our approach [1,2] employs shallow local dialogue structure, identifying short subdialogues and classifying the utterances within them as to their role in defining the action item. This approach improves accuracy and allows

A. Popescu-Belis, S. Renals, and H. Bourlard (Eds.): MLMI 2007, LNCS 4892, pp. 48–59, 2007.

extraction of specific information about individual semantic properties of the action item (such as what is to be done and who has agreed to take responsibility for it). However, one negative consequence of this approach is that data needs to be annotated with this structure – a complex and costly process.

In this paper we investigate the use of user interaction (in combination with classifier outputs) to automatically annotate previously unseen data, thus producing new training data and enabling a system to learn through use alone.

1.1 Action Item Detection

Our methods for annotating action items and automatically detecting them rely on a two-level notion of dialogue structure. First, short sequences of utterances are identified as *action item discussions*, in which an action item is discussed and committed to. Second, the utterances within these subdialogues are identified as belonging to zero or more of a set of four specific dialogue act types that can be thought of as *properties* of the action item discussion: *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 properties). A description of the annotation schema and classification method may be found in [1].

1.2 Data Needs and Implicit User Supervision

Because dialogue annotation is resource-intensive, we are interested in methods for producing annotated data with minimal human effort. Additionally, the characteristics of action item discussion vary substantially across users and meeting types. We therefore would like the system to learn "in the wild" and adapt to new observations without the need for *any* human annotation.

Rather, we would prefer to use *implicit user supervision* by harnessing subtle user interactions with the system as feedback that helps to improve performance over time. Implicit supervision of this kind has proved effective for topic segmentation and identification in meetings [10].

Figure 1 shows a broad view of our architecture for using implicit supervision. First, a set of utterance classifiers detects the action-item-related dialogue acts in a meeting and tags them. Then a subdialogue classifier identifies patterns of these tagged utterances to hypothesize action items. The relevant utterances are then fed into a summarization algorithm suitable for presentation in a user interface. 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.

1.3 User Interfaces for Meeting Assistance

The following question now arises: What kinds of interfaces could harness user feedback most effectively?

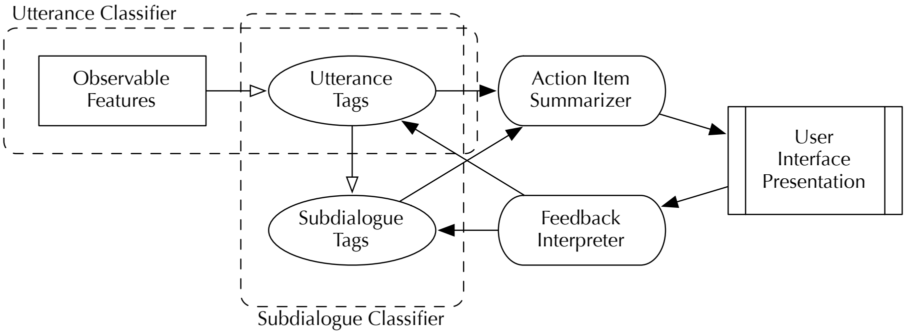

Fig. 1. An outline of the action item detection and feedback system

In [3], we described a meeting browser that allows participants to view a set of automatically hypothesized action item summaries after a meeting. Participants can confirm a hypothesized action item by adding it to a to-do list, or reject it by deleting it. They can also change the hypothesized properties (the who, what, and when for the action item) by selecting from alternate hypotheses or changing text directly. While a number of meeting browser tools allow users to inspect different facets of information that might be gleaned from a meeting (see [11] for an overview), our browser tool was specifically designed to harvest these ordinary user interactions for implicit user feedback.

Specifically, our browser is designed to verify two types of information: the *time* when an action item discussion occurred, and its description in *text*—gleaned from the language used in conversation—that contributes to our understanding of the action item's properties. Different feedback actions may give different information about each of these types. For example, overall confirmation tells us that the extracted text descriptions were adequate, and therefore implicitly confirms that the time period used to extract that text must also have been correct. Similarly, editing just one property of a hypothesized action item might tell us that the overall time period was correct, but that one aspect of the extracted text could have been better.

The distinction between temporal and textual information is important, since it is not clear which of these provides the most benefit for creating new accurate training data. And interfaces that emphasize one or the other may vary in their usefulness and cognitive demands. Certain common types of "meeting interfaces" (like handwritten notes or to-do lists) may provide information about the text of an action item, but not about the time it was discussed. Other in-meeting interfaces could provide temporal information instead of or in addition to text, such as flags or notes made during the meeting using a PDA, a digital pen, or electronic note-taking software like SmartNotes [12].

The distinction between user- and system-initiative is also important. A system-initiative approach might be to identify action items to the user while the meeting is ongoing, perhaps by popping up a button on a PDA which the user can confirm or reject. However, a user-initiative approach – allowing the user to either take notes or flag parts of the meeting where action item discussions occur – might provide more independent information, though perhaps at the cost of higher cognitive demands on the user. So this third characteristic of *initiative*, in addition to the temporal and textual characteristics, could have a profound effect on the quality of supervision a user can provide as it relates to producing valuable annotations.

Thus, we wish to address two basic questions in this paper. First, to what extent do the two informational types of *time* and *text* contribute to producing valuable data for implicit supervision of action item detection? Second, does user- or computer- *initiative* produce more valuable information?

2 Method

2.1 Outline

Our goal is to investigate the efficacy of various methods of learning from implicit human supervision by comparing various possible user interfaces that offer users different capabilities for providing feedback. But rather than implement all of these possible interfaces to record real feedback data, we compare the best possible results that each notional interface could achieve, given an "ideal" user. We simulate this ideal user by using our existing gold-standard annotations and positing them as user feedback. That is, we use varying amounts of the information provided by our gold-standard annotations in an attempt to reproduce those annotations in their entirety.

The procedure involves the following steps. First, a baseline classifier is trained on an *initial* set of human annotations of action items. That classifier then produces action item hypotheses for a second set of meetings, the *learning* set. Next, for each notional user interface, we translate our gold-standard human annotations of the learning set into idealized user feedback interactions – rejecting, adding, or otherwise editing the action items – while constraining the types of feedback information (time, text, and initiative) to comply with the constraints afforded by each interface.

These idealized interactions are then used to modify ("correct") our automatically-generated action item hypotheses from the learning set, producing a new updated set of hypotheses that are used as new training data. The accuracy of this updated set can be evaluated by comparing its corrected hypotheses to the existing human annotations. We can also evaluate the effect of our feedback on the classifiers by retraining them on the union of the initial dataset and the updated dataset, and then testing then again on a third, held-out *test* set.

2.2 Datasets

For our experiments we use 18 meetings selected from the ICSI Meeting Corpus [13]. The ICSI corpus contains unscripted natural meetings, most of which are recordings of regularly occurring project group meetings. We have annotated a series of 12 meetings held by the Berkeley "Even Deeper Understanding" project group (code Bed) and an additional 6 meetings selected randomly from the rest of the corpus. There are a total of 177 action items in the 18 meetings, with 944 total utterances tagged. This makes an average of 9.8 action items per meeting and 5.3 utterances per action item.

We distribute the meetings randomly into the three sets described above: an *initial* set of meetings for training the baseline classifier, a *learning* set of meetings which the system will encounter "in the wild" and learn from through user feedback, and a *test* set used for an overall performance evaluation. Each of these sets has an equal proportion of meetings from the Bed series. These sets and their human annotations can be summarized as:

- An *initial* set of meetings I, and the set of *human* annotations D_{IH}
- A *learning* set of meetings L, and the set of *human* annotations D_{LH}
- A *test* set of meetings T, and the set of *human* annotations D_{TH}

2.3 Baseline Experiments with No User Input

Using the test set for evaluation, we compare the performance of the re-trained classifiers to three baseline measures, in which the user is not a factor. The first baseline, *Initial-only*, assumes no means of obtaining user input on the L meetings at all, and thus provides an expected lower bound to the performance: we simply ignore L and train classifiers only on I. The second, *Optimal*, is a ceiling which provides an upper bound by assuming perfect annotation of the L meetings: we train the classifiers on human annotations of both I and L. Finally, the third baseline, *Naively-retrained*, examines the effect of retraining the classifier on its own output for L (the automatically-produced hypotheses D_{LA}) without modification by user feedback. The baseline experiments are therefore:

- *Initial-only*: Train on D_{IH}; test against D_{TH}
- *Optimal*: Train on $D_{IH} \cup D_{LH}$; test against D_{TH}
- *Naively-retrained*: Train on $D_{IH} \cup D_{LA}$; test against D_{TH}

2.4 Experiments Involving User Input

For each type of user interface, we perform the following experimental steps:

- *Step 1*: Train on D_{IH}
- *Step 2*: Produce automatic hypotheses D_{LA}
- *Step 3*: Update D_{LA} based on user feedback, producing a dataset D_{LU}
- *Step 4*: Test the updated D_{LU} against the gold-standard D_{LH}
- *Step 5*: Retrain on $D_{IH} \cup D_{LU}$
- *Step 6*: Test against D_{TH} and the baselines

We can test the effectiveness of user feedback in two ways. Results can be presented in terms of the accuracy of the updated hypotheses for L (i.e. directly measuring agreement between D_{LH} and D_{LU}). Results can also be presented in terms of the overall effect on classifier performance as tested on T (i.e. measuring agreement between D_{TH} and D_{TA}, and comparing with the above baselines).

3 User Interfaces and Artificial Feedback

We characterize potential interfaces along three dimensions: *temporal* (whether an interface provides information about *when* action items were discussed), *textual* (whether it provides information about the *properties* of the action items), and *initiative* (whether interaction is initiated by computer, user, or both).

Accordingly, we can define a set of hypothetical user interfaces, summarized in Table 1. The *Proactive Button* provides only user-initiated temporal information, simply allowing the user to signal that an action item has just occurred at any time during the meeting – this could be realized as a virtual button on a tablet PC or PDA, or perhaps digital paper. The *Reactive Button* provides computer-initiated temporal information: it tells the user during a meeting when an action item is detected, requiring the user to confirm or reject (or ignore) the hypothesis. Again, this could be realized on a PC, PDA, or phone. Our third interface, *Post-meeting Notes*, assumes only textual information supplied by the user after the meeting is finished, either via note-taking software or by scanning hand-written notes. Our *In-meeting Notes* interface provides both textual and temporal information, assuming that the user is willing to take descriptive notes when action items are discussed (perhaps via collaborative note-taking software).

Table 1. The set of user interfaces investigated

Interface	Temporal Information	Textual Information	Initiative
Proactive Button	Yes	No	User
Reactive Button	Yes	No	Computer
Post-Meeting Notes	No	Yes	User
In-Meeting Notes	Yes	Yes	User

3.1 Simulating Feedback from "idealized" Users

To simulate feedback as it would be produced by an "ideal" (perfectly informative and correct) user, we use information from our existing human annotations.

To simulate **user-initiated** feedback, we need to provide the textual descriptions and/or discussion times of each action item. Times are taken as the end time of the final utterance in an annotated action item subdialogue. Text information is taken from the properties annotated for each individual action item (the task description, the timeframe description, and the identity of the responsible party). Note that annotators were allowed to specify these properties as free text, paraphrasing or rewording the actual discussion as needed: there was no requirement to copy the words or phrases actually used in the transcripts. While they showed a tendency to re-use important words from the utterances themselves, this seems entirely natural and we expect that users would behave similarly. Importantly, annotated information about which utterances belong to a subdialogue, or which utterances play which dialogue act roles, is not used.

To simulate **computer-initiated** feedback, we must compare the automatic hypotheses with the gold-standard annotations, and provide negative or positive feedback accordingly. This requires a criterion for acceptability, which must vary depending on the interface's information content. Where temporal information only is concerned, we class a hypothesis as correct if its corresponding subdialogue period overlaps by more than 50% with that of a gold-standard subdialogue. Where textual information is concerned, we compare each property description using a string similarity metric, and class a hypothesis as correct if the similarity is above a given threshold (see below for more details).

3.2 Interpreting Feedback as Annotation

Given these varying degrees of feedback information, our task is now to infer a complete set of structured action item annotations (both the action item subdialogue periods, and the individual utterances which perform the various dialogue acts within those periods). The inference method depends, of course, on the amount and type of information provided – a summary is shown in Table 2.

The simplest case is that of overall **confirmation or rejection** of a computer-generated hypothesis (as provided by computer-initiated feedback, whether determined on a temporal or textual basis). In this case, we already have a record of which utterances were involved in generating the hypothesis, and their hypothesized dialogue act types; if confirmed, we can use these types directly as (positive) annotation labels; if rejected, we can mark all these utterances as negative instances for all dialogue act types.

The most complex case is that of independent **creation** of a new action item (as provided by user-initiated feedback). In this case, we have no information as to which utterances are involved, but must find the most likely candidates for each relevant dialogue act type; of course, we may have an indicative time, or textual descriptions, or both, to help constrain this process. Given only the time, we must use what prior information we have about the characteristics of the various dialogue act types: given our approach, this means using the existing subclassifiers – using each one to assign a confidence to each utterance within a realistic time window, and labeling those above a given confidence threshold. However, textual information (if available) can provide more independent evidence, at least for semantically informative dialogue act types. For the purely textual *task description* and *timeframe* acts, we can use an independent similarity measure to score each utterance, and label accordingly;[1] for *owner* acts, where the owner's identity usually arises through discourse reference (e.g. personal pronouns) rather than lexical information, we can filter potential utterances based on their referential compatibility. However, as the *agreement* act type does not correspond to any textual information that a user might realistically provide, it must always be assigned using the existing subclassifier.

[1] Our current similarity measure uses a heuristic combination of lexical overlap and semantic distance (using WordNet [14]); we plan to investigate alternatives in future.

Table 2. Inference procedures for each feedback type. Here, "likely" refers to the use of subclassifier confidences, "relevant" to the use of similarity measures.

Feedback	Text Info.	Time Info.	Procedure
`confirm`	(Given)	(Given)	Label all utterances as hypothesized.
`reject`	(Given)	(Given)	Label all utterances as negative.
`edit`	Yes	(Given)	Label most relevant utterance(s) in time window.
`create`	No	Yes	Label most likely utterances in time window.
`create`	Yes	Yes	Label most relevant utterances in time window.

Other cases such as **editing** of an individual property of an action item fall in between these two cases: we use a relevant similarity measure (if text is available) or the relevant subclassifier (if not) to label the most likely corresponding utterance(s) – but in these cases we have more constraining information (knowledge of the part of the subdialogue/hypothesis *not* being edited).

Note that an alternative to purely computer- or user-initiated feedback exists: we can attempt to interpret user-initiated feedback as *implicitly* giving feedback on the computer's hypotheses. This way, a user specification of an action item could be interpreted as an implicit confirmation of a suitable overlapping hypothesis (and a rejection of an unsuitable or non-overlapping one) if one exists, and only as an independent creation otherwise. We investigate both approaches.

3.3 Research Questions

By analyzing these re-interpreted annotations as idealized feedback coming from different types of interfaces, we hope to answer a few questions.

First, in comparing the two types of *time* data and *text* data, will either of these types prove to be more informative than the other? Will either type prove itself not valuable at all? A "yes" to either of these questions could save us time that we might otherwise spend testing interfaces with no inherent promise.

Similarly, since relying on user initiative can burden the user in a way that a system-initiative interface doesn't (by requiring the user to keep another task "in mind" while doing other things), is there any value to a *user-initiative* system over and above a *system-initiative* one? And is there a notable benefit to combining both initiatives, by treating user-initiated actions as implicit confirmation for system-initiated ones?

Our final question is how the overall performances compare to the baseline cases. How close is the overall performance enabled by feedback to the ideal performance achieved when large amounts of gold-standard annotated data are available (our *Optimal* ceiling)? Does the use of feedback really provide better performance than naively using the classifier to re-annotate (the *Naively-retrained* baseline)? If not, we must consider the possibility that such semi-supervised feedback-based training offers little benefit over a totally unsupervised approach, and save users (and ourselves) some wasted effort.

4 Results

We evaluate the experimental results in two separate ways. The first evaluation directly evaluates the quality of the new training data inferred from feedback on the L dataset. Tables 3 & 4 report the accuracy of the updated annotations D_{LU} compared with the gold-standard human annotations D_{LH}, both in terms of the kappa metric (as widely used to assess inter-annotator agreement [15]) and as F-scores for the task of retrieving the utterances which should be annotated. Kappa figures are given for each of the four utterance classes, showing the accuracy in identifying whether utterances belong to the four separate classes of *description*, *timeframe*, *owner*, and *agreement*, together with the figure for all four classes taken together (i.e. agreement on whether an utterance should be annotated as action-item-related or not). Table 3 shows these results calculated over individual utterances; Table 4 shows the same, but calculated over 30-second intervals – note that training data which assigns, say, the agreement class to an incorrect utterance, but one which is within a correct subdialogue, may still be useful for training the overall classifier.

The second evaluation shows the effect on overall performance of re-training on this new inferred data. Table 5 reports the classifier performance on the test set T, i.e. the accuracy of the hypothesized D_{TA} compared with the gold-standard D_{TH}. We show results as F-scores for two retrieval tasks: firstly, identifying the individual component utterances of action items; and secondly, identifying the presence of action items within 30-second intervals (an approximation of the task of identifying action item subdialogues).

The training data quality results (Tables 3 & 4) suggest several possible conclusions. Firstly, we see that using either temporal or text information alone allows improvement over raw classifier accuracy, suggesting that either can be useful in training. Secondly, combining both types of information (in-meeting notes) does best. Thirdly, textual information seems to be more useful than temporal (post-meeting notes do better than either button). This is useful to

Table 3. Utterance-level accuracy for the training annotations inferred from user feedback (D_{LU}) in comparison to the gold-standard human annotations (D_{LH})

Interface	Utterance Classes				Average	
(& implicit hyp use)	agreement	description	timeframe	owner	Kappa	F_1
(Raw hypotheses)	0.06	0.13	0.03	0.12	0.13	0.15
Proactive Button	0.22	0.30	0.17	0.37	0.35	0.36
–"– (implicit)	0.21	0.35	0.16	0.35	0.39	0.41
Reactive Button	0.15	0.31	0.09	0.28	0.32	0.33
Post-meeting Notes	0.26	0.65	0.75	0.29	0.56	0.56
–"– (implicit)	0.10	0.32	0.13	0.15	0.25	0.27
In-meeting Notes	0.26	0.72	0.75	0.40	0.61	0.62
–"– (implicit)	0.21	0.61	0.32	0.26	0.52	0.53

Table 4. 30-second interval accuracy for the training annotations inferred from user feedback (D_{LU}) in comparison to the gold-standard human annotations (D_{LH})

Interface	Utterance Classes				Average	
(& implicit hyp use)	agreement	description	timeframe	owner	Kappa	F_1
(Raw hypotheses)	0.13	0.23	0.11	0.22	0.19	0.27
Proactive Button	0.46	0.71	0.30	0.67	0.65	0.68
–”– (implicit)	0.41	0.75	0.44	0.63	0.64	0.67
Reactive Button	0.34	0.51	0.35	0.46	0.42	0.45
Post-meeting Notes	0.39	0.80	0.89	0.43	0.75	0.77
–”– (implicit)	0.19	0.41	0.24	0.29	0.36	0.43
In-meeting Notes	0.43	0.91	0.89	0.53	0.82	0.84
–”– (implicit)	0.43	0.89	0.68	0.62	0.79	0.81

Table 5. The classification accuracy of the retrained classifiers' hypotheses on the test set (D_{TU}), and those of the baseline classifiers

	Utterances			30-sec Windows		
Interface	Prec.	Recall	F_1	Prec.	Recall	F_1
Initial-only	0.08	0.13	0.09	0.21	0.27	0.22
Naively-retrained	0.09	0.20	0.12	0.23	0.45	0.30
Optimal	0.19	0.26	0.22	0.47	0.44	0.45
Proactive Button	0.18	0.25	0.21	0.47	0.48	0.46
Reactive Button	0.20	0.26	0.22	0.49	0.55	0.50
Post-meeting Notes	0.14	0.20	0.17	0.41	0.41	0.40
In-meeting Notes	0.16	0.27	0.20	0.42	0.52	0.46

know for interface design; it seems likely that temporal synchrony of interface actions and actual discussion of action item may be less exact with real users, so a design which does not have to rely on this synchrony may be advantageous (see below for a discussion of future experiments to investigate this.)

We also note that user initiative does seem to provide extra information above that provided by a purely system-initiative approach (proactive beats reactive). Of course, this may be influenced by the current low overall performance of the classifiers themselves, and may change as performance improves; however, it does suggest that a new technology such as action item detection, with inherently high error rates, is best applied without system initiative until accuracy improves. Similarly, implicit use of computer hypotheses harms the performance of the text-based notes; although it seems to marginally help the proactive button.

We note that the absolute level of agreement for utterance-level annotations is poor, and well below that which might be expected from human annotators. However, some utterance classes do well in some cases, with task description and timeframe utterances giving good agreement with the notes-based interfaces (unsurprisingly, these are the utterance classes which convey most of the information which a text note might contain). And all interfaces achieve respectable levels when considered over 30-second intervals, allowing us to expect that these inferred data could to some extent replace purely human annotation.

The overall performance results (Table 5) show that all kinds of feedback improve the system. All interfaces outperformed the *Naively-retrained* and *Initial-only* baselines; in fact, at the 30-second interval level, all perform approximately as well as our *Optimal* ceiling. However, we hesitate to draw strong conclusions from this, as the test set is currently small – we also note that differences in training data accuracy do not seem to translate directly into the performance differences one might expect, and this may also be due to small test set size.

5 Conclusions and Further Work

These results now lead us directly to the important problem of balancing cognitive load with usefulness of feedback. As intuition predicts, interfaces which supply more information perform better, with synchronous note-taking the most useful. But there is a cognitive price to pay with such interfaces. For now, our results suggest that interfaces like the proactive or reactive buttons, which likely demand less of users' attention, can still significantly improve results, as can post-meeting note-taking, which avoids distractions during the meeting.

Of course, these idealized interactions only give us a Platonic glimpse of what can be expected from the behavior of actual people working in the shadows of the real world. So our next step is to run an experiment with human subjects using these types of interfaces during actual meetings, and compare actual use data with the results herein in order to understand how well our observations of these simulated interfaces will extend to actual interfaces, and to ascertain the level of cognitive effort each interface demands. Then we can hope to determine an optimal balance between the demands of a "meeting assistant" system and the demands of the meeting participants who use it.

Future research will also involve efforts to improve overall classifier performance, which is currently low. Although action item detection is a genuinely hard problem, we believe that improvement can be gained both for utterance and subdialogue classifiers by investigating new classification techniques and feature sets (for example, the current classifiers use only basic lexical features).

Our final set of future plans involve exploring new and better ways for interpreting user feedback, updating automatically-produced hypotheses, and making decisions about retraining. Individual meetings display a high degree of variability, and we believe that feedback on certain types of meetings (e.g. planning meetings) will benefit the system greatly, while other types (e.g. presentations) may not. We will therefore investigate using global qualities of the updated hypotheses to determine whether or not to retrain on certain meetings at all.

References

1. Purver, M., Ehlen, P., Niekrasz, J.: Detecting action items in multi-party meetings: Annotation and initial experiments. In: Renals, S., Bengio, S., Fiscus, J.G. (eds.) MLMI 2006. LNCS, vol. 4299, pp. 200–211. Springer, Heidelberg (2006)
2. Purver, M., Dowding, J., Niekrasz, J., Ehlen, P., Noorbaloochi, S., Peters, S.: Detecting and summarizing action items in multi-party dialogue. In: Proceedings of the 8th SIGdial Workshop on Discourse and Dialogue, Antwerp, Belgium (2007)
3. Ehlen, P., Purver, M., Niekrasz, J.: A meeting browser that learns. In: Proceedings of the AAAI Spring Symposium on Interaction Challenges for Intelligent Assistants (2007)
4. Romano, Jr. N.C., Nunamaker, Jr. J.F.: Meeting analysis: Findings from research and practice. In: Proceedings of the 34th Hawaii International Conference on System Sciences (2001)
5. Cohen, W., Carvalho, V., Mitchell, T.: Learning to classify email into "speech acts". In: Proceedings of Empirical Methods in Natural Language Processing, pp. 309–316 (2004)
6. Corston-Oliver, S., Ringger, E., Gamon, M., Campbell, R.: Task-focused summarization of email. In: Proceedings of the 2004 ACL Workshop Text Summarization Branches Out (2004)
7. Bennett, P.N., Carbonell, J.: Detecting action-items in e-mail. In: Proceedings of the 28th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, Salvador, Brazil, ACM Press, New York (2005)
8. Gruenstein, A., Niekrasz, J., Purver, M.: Meeting structure annotation: data and tools. In: Proceedings of the 6th SIGdial Workshop on Discourse and Dialogue, Lisbon, Portugal (2005)
9. Morgan, W., Chang, P.C., Gupta, S., Brenier, J.M.: Automatically detecting action items in audio meeting recordings. In: Proceedings of the 7th SIGdial Workshop on Discourse and Dialogue, Sydney, Australia. Association for Computational Linguistics, pp. 96–103 (2006)
10. Banerjee, S., Rudnicky, A.: Segmenting meetings into agenda items by extracting implicit supervision from human note-taking. In: IUI 2007. Prooceedings of the International Conference on Intelligent User Interfaces, Honolulu, Hawaii, ACM Press, New York (2007)
11. Tucker, S., Whittaker, S.: Accessing multimodal meeting data: Systems, problems and possibilities. In: Bengio, S., Bourlard, H. (eds.) MLMI 2004. LNCS, vol. 3361, pp. 1–11. Springer, Heidelberg (2005)
12. Banerjee, S., Rudnicky, A.: Smartnotes: Implicit labeling of meeting data through user note-taking and browsing. In: Proceedings of the Human Language Techonolgy Conference of the NAACL (2006) (companion volume)
13. Janin, A., Baron, D., Edwards, J., Ellis, D., Gelbart, D., Morgan, N., Peskin, B., Pfau, T., Shriberg, E., Stolcke, A., Wooters, C.: The ICSI meeting corpus. In: ICASSP. Proceedings of the 2003 International Conference on Acoustics, Speech, and Signal Processing (2003)
14. Miller, G.A.: WordNet: A lexical database for English. Communications of the ACM 38(11), 39–41 (1995)
15. Carletta, J.: Assessing agreement on classification tasks: The kappa statistic. Computational Linguistics 22(2), 249–255 (1996)