You Are What You Say: Using Meeting Participants' Speech to Detect their Roles and Expertise

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

Our goal is to automatically detect the functional roles that meeting participants play, as well as the expertise they bring to meetings. To perform this task, we build decision tree classifiers that use a combination of simple speech features (speech lengths and spoken keywords) extracted from the participants' speech in meetings. We show that this algorithm results in a role detection accuracy of 83% on unseen test data, where the random baseline is 33.3%. We also introduce a simple aggregation mechanism that combines evidence of the participants' expertise from multiple meetings. We show that this aggregation mechanism improves the role detection accuracy from 66.7% (when aggregating over a single meeting) to 83% (when aggregating over 5 meetings).

1 Introduction

A multitude of meetings are organized every day around the world to discuss and exchange important information, to make decisions and to collaboratively solve problems. Our goal is to create systems that automatically understand the discussions at meetings, and use this understanding to assist meeting participants in various tasks during and after meetings. One such task is the retrieval of information from previous meetings, which is typically a difficult and time consuming task for the human to perform (Banerjee et al., 2005). Another task is to automatically record the action items being discussed at meetings, along with details such as when the action is due, who is responsible for it, etc.

Meeting analysis is a quickly growing field of study. In recent years, research has focussed on automatic speech recognition in meetings (Stolcke et al., 2004; Metze et al., 2004; Hain et al., 2005), activity recognition (Rybski and Veloso, 2004), automatic meeting summarization (Murray et al., 2005), meeting phase detection (Banerjee and Rudnicky, 2004) and topic detection (Galley et al., 2003). Relatively little research has been performed on automatically detecting the *roles* that meeting participants play as they participate in meetings. These roles can be functional (e.g. the *facilitator* who runs the meeting, and the *scribe* who is the designated note taker at the meeting), discourse based (e.g. the *presenter*, and the *discussion participant*), and expertise related (e.g. the *hardware acquisition expert* and the *speech recognition research expert*). Some roles are tightly scoped, relevant to just one meeting or even a part of a meeting. For example, a person can be the facilitator of one meeting and the scribe of another, or the same person can be a presenter for one part of the meeting and a discussion participant for another part. On the other hand, some roles have a broader scope and last for the duration of a project. Thus a single person may be the speech recognition expert in a project and have that role in all meetings on that project. Additionally, the same person can play multiple roles, e.g. the scribe can be a speech recognition expert too.

Automatic role detection has many benefits, espe-

Proceedings of the Analyzing Conversations in Text and Speech (ACTS) Workshop at HLT-NAACL 2006, pages 23–30,
New York City, New York, June 2006. ©2006 Association for Computational Linguistics

cially when used as a source of constraint for other meeting understanding components. For example, detecting the facilitator of the meeting might help the automatic topic detection module if we know that facilitators officially change topics and move the discussion from one agenda item to the next. Knowing who the speech recognition expert is can help the automatic action item detector: If an action item regarding speech recognition has been detected but the *responsible person* field has not been detected, the module may place a higher probability on the speech recognition expert as being the responsible person for that action item. Additionally, detecting who is an expert in which field can have benefits of its own. For example, it can be used to automatically direct queries on a particular subject to the person deemed most qualified to answer the question, etc. Basic information such as participant role and expertise needs to be robustly extracted if it is to be of use to the more sophisticated stages of understanding. Accordingly, we have based our role detection algorithm on simple and highly accurate speech features, as described in section 5.1.2.

(Banerjee and Rudnicky, 2004) describes the automatic detection of discourse roles in meetings. These roles included *presenter* (participants who make formal presentations using either slides or the whiteboard), *discussion participant* (participants involved in a discussion marked by frequent turn changes), *observer* (participants not speaking, but nevertheless consuming information during a presentation or discussion), etc. In this paper we focus on automatically detecting the *functional* and *expertise* based roles that participants play in a meeting. In the next section we describe the data that is used in all our role detection work in this paper. In subsequent sections we describe the role detection algorithm in more detail, and present evaluation results.

2 The Y2 Meeting Scenario Data

Our research work is part of the Cognitive Assistant that Learns and Organizes project (CALO, 2003). A goal of this project is to create an artificial assistant that can understand meetings and use this understanding to assist meeting participants during and after meetings. Towards this goal, data is being collected by creating a rich multimodal record of meetings (e.g. (Banerjee et al., 2004)). While a large part of this data consists of natural meetings (that would have taken place even if they weren't being recorded), a small subset of this data is "scenario driven" – the *Y2 Scenario Data*.

Meeting #	Typical scenario
1	Hiring Joe: Buy a computer and find office space for him
2	Hiring Cindy and Fred: Buy com–puters & find office space for them
3	Buy printer for Joe, Cindy and Fred
4	Buy a server machine for Joe, Cindy and Fred
5	Buy desktop and printer for the meeting leader

Table 1: Typical Scenario Instructions

The Y2 Scenario Data consists of meetings between groups of 3 or 4 participants. Each group participated in a sequence of up to 5 meetings. Each sequence had an overall scenario – the purchasing of computing hardware and the allocation of office space for three newly hired employees. Participants were told to assume that the meetings in the sequence were being held one week apart, and that between any two meetings "progress" was made on the action items decided at each meeting. Participants were given latitude to come up with their own stories of what "progress" was made between meetings. At each meeting, participants were asked to review progress since the last meeting and make changes to their decisions if necessary. Additionally, an extra topic was introduced at each meeting, as shown in table 1.

In each group of participants, one participant played the role of the *manager* who has control over the funds and makes the final decisions on the purchases. The remaining 2 or 3 participants played the roles of either the *hardware acquisition expert* or the *building facilities expert*. The role of the hardware expert was to make recommendations on the buying of computers and printers, and to actually make the purchases once a decision was made to do so. Similarly the role of the building expert was to make recommendations on which rooms were available to fit the new employees into. Despite this role assign-

ment, all participants were expected to contribute to discussions on all topics.

To make the meetings as natural as possible, the participants were given control over the evolution of the story, and were also encouraged to create conflicts between the manager's demands and the advice that the experts gave him. For example, managers sometimes requested that all three employees be put in a single office, but the facilities expert announced that no 3 person room was available, unless the manager was agreeable to pay extra for them. These conflicts led to extended negotiations between the participants. To promote fluency, participants were instructed to use their knowledge of existing facilities and equipment instead of inventing a completely fictitious set of details (such as room numbers).

The data we use in this paper consists of 8 sequences recorded at Carnegie Mellon University and at SRI International between 2004 and 2005. One of these sequences has 4 meetings, the remaining have 5 meetings each, for a total of 39 meetings. 4 of these sequences had a total of 3 participants each; the remaining 4 sequences had a total of 4 participants each. On average each meeting was 15 minutes long. We partitioned this data into two roughly equal sets, the training set containing 4 meeting sequences, and the test set containing the remaining 4 sets. Although a few participants participated in multiple meetings, there was no overlap of participants between the training and the test set.

3 Functional Roles

Meeting participants have **functional roles** that ensure the smooth conduct of the meeting, without regard to the specific contents of the meeting. These roles may include that of the *meeting leader* whose functions typically include starting the meeting, establishing the agenda (perhaps in consultation with the other participants), making sure the discussions remain on–agenda, moving the discussion from agenda item to agenda item, etc. Another possible functional role is that of a the designated *meeting scribe*. Such a person may be tasked with the job of taking the official notes or minutes for the meeting.

Currently we are attempting to automatically detect the meeting leader for a given meeting. In our data (as described in section 2) the participant playing the role of the *manager* is always the meeting leader. In section 5 we describe our methodology for automatically detecting the meeting leader.

4 Expertise

Typically each participant in a meeting makes contributions to the discussions at the meeting (and to the project or organization in general) based on their own expertise or skill set. For example, a project to build a multi–modal note taking application may include project members with expertise in speech recognition, in video analysis, etc. We define **expertise based roles** as roles based on skills that are relevant to participants' contributions to the meeting discussions and the project or organization in general. Note that the expertise role a participant plays in a meeting is potentially dependent on the expertise roles of the other participants in the meeting, and that a single person may play different expertise roles in different meetings, or even within a single meeting. For example, a single person may be the "speech recognition expert" on the note taking application project that simply uses off–the–shelf speech recognition tools to perform note taking, but a "noise cancellation" expert on the project that is attempting to improve the in–house speech recognizer. Automatically detecting each participant's roles can help such meeting understanding components as the action item detector.

Ideally we would like to automatically discover the roles that each participant plays, and cluster these roles into groups of similar roles so that the meeting understanding components can transfer what they learn about particular participants to other (and newer) participants with similar roles. Such a role detection mechanism would need no prior training data about the specific roles that participants play in a new organization or project. Currently however, we have started with a simplified participant role detection task where we do have training data pertinent to the specific roles that meeting participants play in the test set of meetings. As mentioned in section 2, our data consists of people playing two kinds of expertise–based roles – that of a hardware acquisition expert, and that of a building facilities expert. In the next section we discuss our

methodology of automatically detecting these roles from the meeting participants' speech.

5 Methodology

Given a sequence of longitudinal meetings, we define our role detection task as a three–way classification problem, where the input to the classifier consists of features extracted from the speech of a particular participant over the given meetings, and the output is a probability distribution over the three possible roles. Note that although a single participant can simultaneously play both a functional and an expertise–based role, in the Y2 Scenario Data each participant plays exactly one of the three roles. We take advantage of this situation to simplify the problem to the three way classification defined above. We induce a decision tree (Quinlan, 1986) classifier from hand labeled data. In the next subsection we describe the steps involved in training the decision tree role classifier, and in the subsequent subsection we describe how the trained decision tree is used to arrive at a role label for each meeting participant.

5.1 Training

5.1.1 Keyword List Creation

One of the sources of information that we wish to employ to perform functional and expertise role detection is the words that are spoken by each participant over the course of the meetings. Our approach to harness this information source is to use labeled training data to first create a set of words most strongly associated with each of the three roles, and then use only these words during the feature extraction phase to detect each participant's role, as described in section 5.1.2.

We created this list of keywords as follows. Given a training set of meeting sequences, we aggregated for each role all the speech from all the participants who had played that role in the training set. We then split this data into individual words and removed *stop words* – closed class words (mainly articles and prepositions) that typically contain less information pertinent to the task than do nouns and verbs. For all words across all the three roles, we computed the degree of association between each word and each of the three roles, using the chi squared method (Yang and Pedersen, 1997), and chose the top 200 high scoring word–role pairs. Finally we manually examined this list of words, and removed additional words that we deemed to not be relevant to the task (essentially identifying a domain–specific stop list). This reduced the list to a total of 180 words. The 5 most frequently occurring words in this list are: *computer, right, need, week* and *space*. Intuitively the goal of this keyword selection pre–processing step is to save the decision tree role classifier from having to automatically detect the important words from a much larger set of words, which would require more data to train.

5.1.2 Feature Extraction

The input to the decision tree role classifier is a set of features abstracted from a specific participant's speech. One strategy is to extract exactly one set of features from all the speech belonging to a participant across all the meetings in the meeting sequence. However, this approach requires a very large number of meetings to train. Our chosen strategy is to *sample* the speech output by each participant multiple times over the course of the meeting sequence, classify each such sample, and then aggregate the evidence over all the samples to arrive at the overall likelihood that a participant is playing a certain role.

To perform the sampling, we split each meeting in the meeting sequence into a sequence of contiguous windows each n seconds long, and then compute one set of features from each participant's speech during each window. The value of n is decided through parametric tests (described in section 7.1). If a particular participant was silent during the entire duration of a particular window, then features are extracted from that silence.

Note that in the above formulation, there is no overlap (nor gap) between successive windows. In a separate set of experiments we used *overlapping* windows. That is, given a window size, we moved the window by a fixed step size (less than the size of the window) and computed features from each such overlapping window. The results of these experiments were no better than those with non–overlapping windows, and so for the rest of this paper we simply report on the results with the non–overlapping windows.

Given a particular window of speech of a partic-

ular participant, we extract the following 2 speech *length* based features:

- Rank of this participant (among this meeting's participants) in terms of the length of his speech during this window. Thus, if this participant spoke the longest during the window, he has a feature value of 1, if he spoke for the second longest number of times, he has a feature value of 2, etc.

- Ratio of the length of speech of this participant in this window to the total length of speech from all participants in this window. Thus if a participant spoke for 3 seconds, and the total length of speech from all participants in this window was 6 seconds, his feature value is 0.5. Together with the rank feature above, these two features capture the amount of speech contributed by each participant to the window, relative to the other participants.

In addition, for each window of speech of a particular participant, and for each keyword in our list of pre–decided keywords, we extract the following 2 features:

- Rank of this participant (among this meeting's participants) in terms of the number of times this keyword was spoken. Thus if in this window of time, this participant spoke the keyword *printer* more often than any of the other participants, then his feature value for this keyword is 1.

- Ratio of the number of times this participant uttered this keyword in this window to the total number of times this keyword was uttered by all the participants during this window. Thus if a participant spoke the word *printer* 5 times in this window, and in total all participants said the word *printer* 7 times, then his feature value for this keyword is $5/7$. Together with the keyword rank feature above, these two features capture the number of times each participant utters each keyword, relative to the other participants.

Thus for each participant, for each meeting window, we extract two features based on the lengths of speech, and 2×180 features for each of the 180 keywords, for a total of 362 features. The true output label for each such data point is the role of that participant in the meeting sequence. We used these data points to induce a classifier using the Weka Java implementation (Witten and Frank, 2000) of the C4.5 decision tree learning algorithm (Quinlan, 1986). This classifier takes features as described above as input, and outputs class membership probabilities, where the classes are the three roles. Note that for the experiments in this paper we extract these features from the *manual transcriptions* of the speech of the meeting participants. In the future we plan to perform these experiments using the transcriptions output by an automatic speech recognizer.

5.2 Detecting Roles in Unseen Data

5.2.1 Classifying Windows of Unseen Data

Detecting the roles of meeting participants in unseen data is performed as follows: First the unseen test data is split into windows of the same size as was used during the training regime. Then the speech activity and keywords based features are extracted (using the same keywords as was used during the training) for each participant in each window. Finally these data points are used as input into the trained decision tree, which outputs class membership probabilities for each participant in each window.

5.2.2 Aggregating Evidence to Assign One Role Per Participant

Thus for each participant we get as many probability distributions (over the three roles) as there are windows in the test data. The next step is to aggregate these probabilities over all the windows and arrive at a single role assignment per participant. We employ the simplest possible aggregation method: We compute, for each participant, the average probability of each role over all the windows, and then normalize the three average role probabilities so calculated, so they still sum to 1. In the future we plan to experiment with more sophisticated aggregation mechanisms that jointly optimize the probabilities of the different participants, instead of computing them independently.

At this point, we could assign to each participant his highest probability role. However, we wish to ensure that the set of roles that get assigned to the

participants in a particular meeting are as diverse as possible (since typically meetings are forums at which different people of different expertise convene to exchange information). To ensure such diversity, we apply the following heuristic. Once we have all the average probabilities for all the roles for each participant in a sequence of meetings, we assign roles to participants in *stages*. At each stage we consider all participants not yet assigned roles, and pick that participant–role pair, say (p, r), that has the highest probability value among all pairs under consideration. We assign participant p the role r, and then *discount* (by a constant multiplicative factor) the probability value of all participant–role pairs (p_i, r_j) where p_i is a participant not assigned a role yet, and $r_j = r$. This makes it less likely (but not impossible) that another participant will be assigned this same role r again. This process is repeated until all participants have been assigned a role each.

6 Evaluation

We evaluated the algorithm by computing the accuracy of the detector's role predictions. Specifically, given a meeting sequence we ran the algorithm to assign a role to each meeting participant, and computed the accuracy by calculating the ratio of the number of correct assignments to the total number of participants in the sequence. Note that it is also possible to evaluate the window–by–window classification of the decision tree classifiers; we report results on this evaluation in section 7.1.

To evaluate this participant role detection algorithm, we first trained the algorithm on the training set of meetings. The training phase included keyword list creation, window size optimization, and the actual induction of the decision tree. On the training data, a window size of 300 seconds resulted in the highest accuracy over the training set. The test at the root of the induced tree was whether the participant's rank in terms of speech lengths was 1, in which case he was immediately classified as a *meeting leader*. That is, the tree learnt that the person who spoke the most in a window was most likely the meeting leader. Other tests placed high in the tree included obvious ones such as testing for the keywords *computer* and *printer* to classify a participant as a hardware expert.

We then tested this trained role detector on the testing set of meetings. Recall that the test set had 5 meeting sequences, each consisting of 5 meetings and a total of 20 meeting participants. Over this test set we obtained a role detection accuracy of 83%. A "classifier" that randomly assigns one of the three roles to each participant in a meeting (without regard to the roles assigned to the other participants in the same meeting) would achieve a classification accuracy of 33.3%. Thus, our algorithm significantly beats the random classifier baseline. Note that as mentioned earlier, the experiments in this paper are based on the manually transcribed speech.

7 Further Experiments

7.1 Optimizing the Window Size

As mentioned above, one of the variables to be tuned during the training phase is the size of the window over which to extract speech features. We ran a sequence of experiments to optimize this window size, the results of which are summarized in figure 1. In this set of experiments, we performed the evaluation on two levels of granularity. The larger granularity level was the "meeting sequence" granularity, where we ran the usual evaluation described above. That is, for each participant we first used the classifier to obtain probability distributions over the 3 roles on every window, and then aggregated these distributions to reach a single role assignment for the participant over the entire meeting sequence. This role was compared to the true role of the participant to measure the accuracy of the algorithm. The smaller granularity level was the "window" level, where after obtaining the probability distribution over the three roles for a particular window of a particular participant, we picked the role with the highest probability, and assigned it to the participant *for that window*. Therefore, for each window we had a role assignment that we compared to the true role of the participant, resulting in an accuracy value for the classifier for every window for every participant. Note that the main difference between evaluation at these two granularity levels is that in the "window" granularity, we did not have any aggregation of evidence across multiple windows.

For different window sizes, we plotted the accuracy values obtained on the test set for the two evalu-

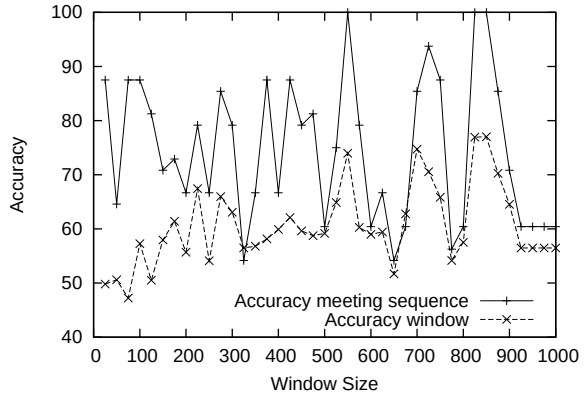

Figure 1: *Effect of Different Window Sizes on Detection Accuracy*

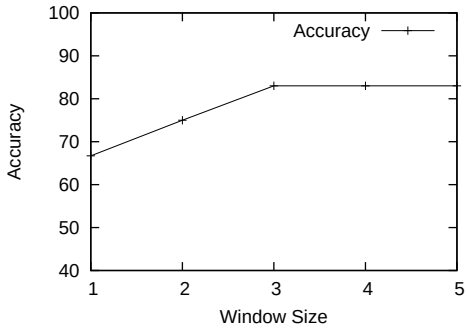

Figure 2: *Accuracy versus Number of Meetings over which Roles were Detected*

ation granularities, as shown in figure 1. Notice that by aggregating the evidence across the windows, the detection accuracy improves for all window sizes. This is to be expected since in the window granularity, the classifier has access to only the information contained in a single window, and is therefore more error prone. However by merging the evidence from many windows, the accuracy improves. As window sizes increase, detection accuracy at the window level improves, because the classifier has more evidence at its disposal to make the decision. However, detection at the meeting sequence level gets steadily worse, potentially because the larger the window size, the fewer the data points it has to aggregate evidence from. These lines will eventually meet when the window size equals the size of the entire meeting sequence.

A valid concern with these results is the high level of noise, particularly in the aggregated detection accuracy over the meeting sequence. One reason for this is that there are far fewer data points at the meeting sequence level than at the window level. With larger data sets (more meeting sequences as well as more participants per meeting) these results may stabilize. Additionally, given the small amount of data, our feature set is quite large, so a more aggressive feature set reduction might help stabilize the results.

7.2 Automatic Improvement over Unseen Data

One of our goals is to create an expertise based role detector system that improves over time as it has access to more and more meetings for a given participant. This is especially important because the roles that a participant plays can change over time; we would like our system to be able to track these changes. In the Y2 Scenario Data that we have used in this current work, the roles do not change from meeting to meeting. However observe that our evidence aggregation algorithm fuses information from all the meetings in a specific sequence of meetings to arrive at a single role assignment for each participant.

To quantify the effect of this aggregation we computed the role detection accuracy using different numbers of meetings from each sequence. Specifically, we computed the accuracy of the role detection over the test data using only the last meeting of each sequence, only the last 2 meetings of each sequence, and so on until we used every meeting in every sequence. The results are summarized in figure 2. When using only the last meeting in the sequence to assign roles to the participants, the accuracy is only 66.7%, when using the last two meetings, the accuracy is 75%, and using the last three, four or all meetings results in an accuracy of 83%. Thus, the accuracy improves as we have more meetings to combine evidence from, as is expected. However the accuracy levels off at 83% when using three or more meetings, perhaps because there is no new information to be gained by adding a fourth or a fifth meeting.

8 Conclusions and Future Work

In this paper we have discussed our current approach to detecting the functional and expertise based roles of meeting participants. We have induced decision

trees that use simple and robust speech based features to perform the role detection. We have used a very simple evidence aggregation mechanism to arrive at a single role assignment per meeting participant over a sequence of meetings, and have shown that we can achieve up to 83% accuracy on unseen test data using this mechanism. Additionally we have shown that by aggregating evidence across a sequence of meetings, we perform better than if we were to use a single meeting to perform the role detection. As future work we plan to remove the constraints that we have currently imposed – namely, we will attempt to learn new roles in test data that do not exist in training data. Additionally, we will attempt to use this role information as inputs to downstream meeting understanding tasks such as automatic topic detection and action item detection.

9 Acknowledgements

This work was supported by DARPA grant NBCH-D-03-0010. The content of the information in this publication does not necessarily reflect the position or the policy of the US Government, and no official endorsement should be inferred.

References

S. Banerjee and A. I. Rudnicky. 2004. 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 2004 – ICSLP)*, Jeju Island, Korea.

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.

CALO. 2003. http://www.ai.sri.com/project/CALO.

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

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

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

G. Murray, S. Renals, and J. Carletta. 2005. Extractive summarization of meeting recordings. In *Proceedings of Interspeech 2005*, Lisbon, Portugal, September.

J. Quinlan. 1986. Induction of decision trees. *Machine Learning*, 1:81–106.

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

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

I. Witten and E. Frank. 2000. *Data Mining - Practical Machine Learning Tools and Techniques with Java Implementations*. Morgan–Kaufmann, San Francisco, CA.

Y. Yang and J. Pedersen. 1997. A comparative study on feature selection in text categorization. In *Proceedings of the International Conference on Machine Learning*, pages 412–420, Nashville, US. Morgan Kaufmann Publishers.