Learning About Meetings

Been Kim and Cynthia Rudin

Massachusetts Institute of Technology
beenkim@csail.mit.edu,rudin@mit.edu

Abstract. Most people participate in meetings almost every day, multiple times a day. The study of meetings is important, but also challenging, as it requires an understanding of social signals and complex interpersonal dynamics. Our aim this work is to use a data-driven approach to the science of meetings. We provide tentative evidence that: i) it is possible to automatically detect when during the meeting a key decision is taking place, from analyzing only the local dialogue acts, ii) there are common patterns in the way social dialogue acts are interspersed throughout a meeting, iii) at the time key decisions are made, the amount of time left in the meeting can be predicted from the amount of time that has passed, iv) it is often possible to predict whether a proposal during a meeting will be accepted or rejected based entirely on the language (the set of persuasive words) used by the speaker.

1 Introduction

"A meeting is indispensable when you don't want to get anything done." [Kayser, 1990]

In the United States alone, an estimated 11 million meetings take place during a typical work day [Newlund, 2012]. Managers typically spend between a quarter and three-quarters of their time in meetings [Mackenzie and Nickerson, 2009], and approximately 97% of workers have reported in a large-scale study [Hall, 1994] that to do their best work, collaboration is essential. Most of us work directly with others every day, and want to be useful participants in meetings. Meeting analysis (the science of meetings) can potentially help us understand various aspects of meetings, find ways to allow us to be more effective participants in our meetings, and help us to create automated tools for meeting assistance. Meeting dynamics can be complex; often proposals are implicitly communicated and accepted [Eugenio et al., 1999]. Despite the plethora of meetings that happen each day, and despite a body of work on meeting analysis [e.g., see Romano Jr and Nunamaker Jr, 2001, for a review], we still cannot claim that we understand the topic of meetings well enough that these studies have led to quantifiable improvements in the overall quality of meetings, nor useful guidelines for more productive meetings. Perhaps if we step back and consider a basic science approach to meeting analysis, we would quantitatively uncover aspects of meetings that could eventually lead to true improvements.

In this work, we develop and use predictive modeling tools and descriptive statistics in order to provide a data-driven approach to the scientific study of meetings. We provide preliminary answers to several key questions: i) Can we automatically detect when the main decisions are made during the meeting, based only on the frequency and type of dialogue acts in a given period of time? In other words, based on the types of utterances people are making, can we determine whether the most important part of the meeting is occurring? ii) Is there any type of pattern of dialogue common to most or all meetings? We would like to know whether there is a "quintessential" pattern of dialogue that we can identify. iii) How long is a meeting going to last, in terms of "wrap-up" time, beyond the time that the main decisions are made? Sometimes this wrap-up time is substantial, and the meeting extends well beyond when the main decisions are made. iv) Can we predict whether a proposal made during a meeting will be accepted or rejected based entirely on the language (the set of persuasive words) used by the speaker? There have been many studies and commentaries focused on "persuasive words," but do those persuasive words truly correlate with successful outcomes during a meeting? Some of these questions can be answered using current machine learning approaches, but others cannot. In particular, finding patterns in dialogue acts (question ii) is not a question that is naturally suited to standard machine learning techniques. In this work, we design an algorithm to learn a sparse representation of a meeting as a graph of dialogue acts, which yields insight about the way social dialogue acts intersperse with work-related dialogue acts.

In our study, we use the most extensive annotated corpus of meetings data in existence, originating from the AMI (Augmented multi-party interaction) project [McCowan et al., 2005]. This dataset contains a large number of annotated meetings (over 12,000 human-labeled annotations). In total, there are 108,947 dialogue acts in total number of 95 meetings, and 26,825 adjacency pairs (explained below). This corpus is derived from a series of real meetings, controlled in the sense that each meeting has four participants who work as a team to compose a new design for a new remote control. Each participant takes a role, either project manager, marketing expert, industrial designer or interface designer. The participants are given training for their roles at the beginning of the task. Documents used to train participants and annotators are publicly available. The interaction between participants is unstructured, and each person freely chooses their own style of interaction. The length of meetings ranges from approximately 10 minutes to 45 minutes, which overlaps with the most common lengths of meetings [discussed by Romano Jr and Nunamaker Jr, 2001]. Here we provide detail on the annotations provided with the corpus.

Dialogue Acts: A dialogue act marks a characteristic of an utterance, representing the intention or the role of the utterance. It is possible to use dialogue act data to predict characteristics of future dialogue [Nagata and Morimoto, 1994], or to do automatic dialogue act tagging [Stolcke et al., 2000, Ji and Bilmes, 2005]. Dialogue acts include questions, statements, suggestions, assessment of suggestions (positive, negative or neutral) and social acts. Each sentence is often divided into pieces and tagged with different dialogues acts.[1] A sequence of

[1] More details of definition of dialogue acts can be found in [McCowan et al., 2005].

dialogue acts looks like this:

```
A: Suggestion
B: Commenting on A’s suggestion
C: Asking questions
A: Answering C’s question
B: Accepting A’s suggestion
```

Decision summary: A summary of decisions made in a meeting and related dialogue acts.

Discussion: A set of dialogue acts and their time stamps that support or reflect decisions described in the decision summary annotation.

Adjacency pairs: An adjacency pair encodes the relationship between two dialogue acts. It represents the reaction to the first dialogue act that is expressed within the second dialogue act; e.g., the first dialogue act can be a suggestion and the second dialogue act can be a positive or negative response to it, as demonstrated in the sequence of dialogue acts above.

We note that meetings can have many purposes, [e.g., see Romano Jr and Nunamaker Jr, 2001], however in this work we study only meetings where the purpose is to make a group judgment or decision (rather than for instance, to ensure that everyone understands, or to explore new ideas and concepts).

Our specific goal in this work is to contribute insights to the new scientific field of meeting analysis. We provide tentative answers to the above questions that can be used for constructing other studies. We do not claim that our results definitively answer these questions, only that they yield hypotheses that can be tested more thoroughly through other surveys. Most medical (and more generally, scientific) studies make or verify hypotheses based on a database or patient study. Hypotheses based on these single studies are sometimes first steps in answering important questions - this is a study of that kind, and we believe our results can be tested and built upon.

Our more general goal is to show that ML can be useful for exploratory study of meetings, including and beyond the questions studied here. This paper is not focused on a specific machine learning technique per se, its goal is to show that ML methods can be applied for exploratory scientific analysis of data in this domain, leading towards the eventual goal of increasing meeting productivity. We start with a question for which a simple application of machine learning tools provides direct insight.

2 Question i: Can we automatically detect when key decisions are made?

If we can learn the statistics of dialogue acts around the time when important decisions are about to be made or being made, we can potentially detect the critical time window of a meeting. One can imagine using this information in several ways, for instance, to know at what point in the meeting to pay more attention (or to join the meeting in the first place), or to use this as part of a historical meetings database in order to fast forward the meeting's recording to the most important parts.

In our setup, each feature vector for the detection problem is a time-shifted bag-of-dialogue-acts. The feature vector for a specific timeframe is defined by the counts of different dialogue acts (e.g., one example has 3 suggestions, 2 positive responses and 10 information exchanges). Using this representation means that the results do not depend on specific keywords within the meeting dialogue; this potentially allows our results to hold independently of the specific type and purpose of the meeting being held. Among dialogue acts, we use only a subset that are known to be relevant to the group decision making process [Eugenio et al., 1999], namely: action directive, offer, accept, reject, info-request, and information. Action directives represent all elicit forms of dialogue acts — dialogue acts that require actions from hearers. By limiting to these dialogue acts, we are now working with a total of 53,079 dialogue acts in the corpus.

Using this definition, a timeframe of dialogue becomes one example, represented by a 6-dimensional vector, where each element is the count of a particular dialogue act within the specified timeframe. The labels are 1 if the decisions are annotated as being made in that timeframe of interest and 0 otherwise. We considered timeframes with average size 70 timestamps (roughly 5 minutes, where each timestamp represents a dialogue act). This choice of timeframe comes from the minimum meeting time considered in the research done on meeting profiles by Panko and Kinney [1995].

We applied supervised and unsupervised classification algorithms to predict whether important decisions are being made within a given block of dialogue. The data were divided into 15 folds, and each fold was used in turn as the test set. The supervised algorithms were SVM with radial basis function (RBF) kernels, SVM with linear kernels, logistic regression, and Naïve Bayes with a Gaussian density assumption on each class and feature. For the unsupervised algorithms, which (purposely) discard the annotations, we used EM with Gaussian Mixture Models and Kmeans. From the resulting two clusters, we chose the better of the two possible label assignments. The AUC, precision, recall, and F-measure for each algorithm's performance on the testing folds are reported in Table 1. The sample mean and sample standard deviation of AUC values over the test folds are reported within the second column of the table.

Table 1. Results for predicting key decision times

Method	**AUC ± Std.**	**Precision**	**Recall**	**F-measure**
SVM-Linear	0.87 ± 0.05	0.89	0.87	0.88
Logistic regression	0.87 ± 0.05	0.70	0.56	0.62
SVM-RBF	0.86 ± 0.06	0.86	0.95	0.90
EM-GMM	0.57 ± 0.06	0.68	0.48	0.56
NB-Gaussian	0.56 ± 0.10	0.88	0.81	0.84
Kmeans	0.48 ± 0.24	0.69	0.34	0.45

The prediction quality, with respect to the AUC, is very similar for SVM-Linear, Logistic Regression, and SVM-RBF. All three methods show high AUC values around 0.86 or 0.87. The three other methods do not perform nearly as well. Naïve Bayes has very strong independence assumptions that are clearly violated here. EM-GMM is a non-convex approach that has a tendency to get stuck in local minima. Both EM-GMM and K-Means are unsupervised, so they do not get the benefit of being trained on human-labeled data. We remark that the predictive performance given by some of these algorithms is quite high given that the imbalance ratio is only 3 to 1 (three "no decision made" examples for each "decision made" example). Logistic regression performed well with respect to the AUC, but not with respect to the other measures. AUC is a rank statistic, whereas the other measures are relative to a decision boundary. Thus, if the decision boundary is in the wrong place, precision, recall, and F-measure suffer, regardless of whether the positives are generally given higher scores than the negatives. (There is an in-depth discussion by Ertekin and Rudin [2011] on logistic regression's performance with respect to the AUC.) The opposite is true for Naïve Bayes, where the decision boundary seems to be in the right place, leading to good classification performance, but the relative placement of positives and negatives within the classes lead to rank statistics that are not at the level of the other algorithms. This could potentially be due to the choice of scoring measure used for the AUC, as the choice of scoring measure is not unique for Naïve Bayes. We chose to score by $P(y=1|\mathbf{x}) \approx P(y=1)\prod_j \hat{P}(x^j|y=1)$, where the empirical probability is computed over the training set. (Here $y=1$ indicates decisions being made, and x^j is the j^{th} feature value for example $\mathbf{x}$.) It is possible that even if an example scored highly for being a member of the positive class, it could score even more highly for being a member of the negative class, and thus be classified correctly. Thus, care should be taken in general when judging Naïve Bayes by the AUC; in this case, quantities computed with respect to the decision boundary (true positives, false positives, etc.) are more natural than rank statistics.

Table 2. Feature ranking using SVM coefficient

Ranking	**Dialogue Acts**	λ ± **Std.**
1	Information	0.30 ±0.031
2	Information Request	0.11 ± 0.03
3	Offer	-0.0076 ± 0.04
4	Action-directive	-0.0662 ±0.04
5	Reject	-0.20 ±0.03
6	Accept	-0.27 ±0.02

We can use the SVM coefficients to understand the distribution of dialogue acts during the important parts of the meeting. As it turns out, the important parts of the meeting are characterized mostly by information and information request dialogue acts, and very few offers, rejections, or acceptances. This is shown in Table 2. We hypothesize that at the important parts of the meeting, when the decisions have been narrowed down and few choices remain, the meeting participants would like to ensure that they have all the relevant information necessary to make the decision, and that the outcome will fit within all of their constraints.

This small experiment has implications for the practical use of machine learning for automated meeting recording and assistance software. First, it is possible to obtain at least 0.87 AUC in detecting the time frame when decisions are made. An algorithm with this level of fidelity could be useful, for instance, in searching through large quantities of meeting data automatically (rather than manually scrolling through each meeting). One could envision having software create an alert that key decisions are being made, so that upper-level management can then choose to join the meeting. Or the software could inform secretarial staff of an estimate for when the meeting is done in order to facilitate schedule planning. (To do this, however, it might be useful to incorporate knowledge from Section 4 about the expected length of the wrap-up time.)

3 Question ii: Is there a pattern of interactions within a meeting?

There is often a mixture of social and work-related utterances during a meeting, and it is not obvious how the two interact. In particular, we would like to study the way in which social acts (positive or negative) interact with work-related acts (i.e., acceptance or rejection of proposals). More abstractly, we would like to know if there is a "quintessential representation" of interactions within a meeting, where a representation is a directed graph of dialogue acts. If there is such a representation, we would like to learn it directly from data. To do this, we will present a discrete optimization approach that uses the notion of "template instantiation." The optimization will be performed via simulated annealing. First, let us formally define the problem of template discovery.

3.1 Formalization of the Problem of Template Discovery

We define a *template* as a graph, where each node takes a value in the set of dialogue acts, and the graph has directed edges from left to right, and additional backwards edges. Formally, let Λ be the set of possible dialogue acts (Λ^n is a string of length n). Specifically, define the set $\mathcal{D}$ as

$$\mathcal{D} : (n \in \{1, \cdots, L\}) \times \Lambda^n \times \Pi_{B,n}$$

so that $\mathcal{D}$ is a set of templates with length of size $n \leq L$, where if the template is of size n, there are n dialogue acts. There are forward edges between neighboring nodes (i.e. dialogue acts). $\Pi_{B,n}$ is the set of all possible backward arrows for the template, containing at most B backward arrows. One can represent $\Pi_{B,n}$ as the set of $n \times n$ lower diagonal binary matrices, 1's indicating backward arrows, with at most B values that are 1. Let X be the set of meetings, which are strings consisting of an arbitrary number of elements of Λ. Define the loss function $l : \mathcal{D} \times X \to \mathbb{Z}_+$ as

$$l(t, x) := \min_{t_j \in \text{instan}(t)} [\text{dist}(t_j, x)]$$

where dist is edit distance, dist : $\Lambda^{n_1} \times \Lambda^{n_2} \to \mathbb{Z}_+$ for strings of lengths n_1 and n_2. We define instan(t) as the set of *template instantiations* t_j of the template t, where a template instantiation is a path through the graph, beginning or ending anywhere within the graph. The path is a sequence of elements from Λ. Consider for instance the example template at the top of Figure 1. Two example template instantiations are provided just below that. The edit distance between two strings is the minimum number of insertions, deletions, and/or substitutions to turn one string into the other. Each meeting x is a string of dialogue acts, and each template instantiation t_j is also a string of dialogue acts, so the edit distance is well-defined.

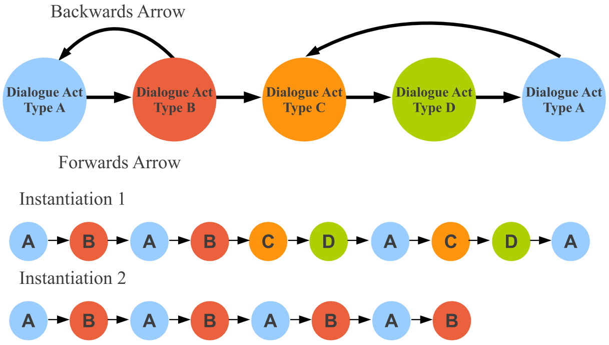


Fig. 1. Example template instantiations. The template is at the top, and two instantiations are below.

We assume that we are given meetings $x_1, \cdots, x_m$ drawn independently from an unknown probability distribution μ over possible meetings X. Define empirical risk $R^{\text{emp}}(t) : \mathcal{D} \to \mathbb{R}_+$ and true risk $R^{\text{true}}(t) : \mathcal{D} \to \mathbb{R}_+$ as:

$$R^{\text{emp}}(t) := \frac{1}{m} \sum_{i=1}^{m} l(t, x_i) \text{ and } R^{\text{true}}(t) := \mathbb{E}_{x \sim \mu}[l(t, x)].$$

We can bound the difference between the empirical risk and true risk for the problem of template discovery as follows:

Theorem 1. *For all $\delta > 0$, with probability at least $1 - \delta$ with respect to the independent random draw of meetings $x \sim \mu$, $x_i \sim \mu$, $i = 1, ..., m$, and for all* *templates t in $\mathcal{D}$ with length at most L and with number of backwards arrows at most B:*

$$R^{\text{true}}(t) \leq R^{\text{emp}}(t) + \sqrt{\frac{\log \sum_{n=0}^{L} \left[|\Lambda|^n \sum_{b=0}^{\min(B,n)} \binom{n(n-1)/2}{b} \right] + \log \frac{1}{\delta}}{2m}},$$

where $|\Lambda|$ is the number of elements in the set Λ.

The proof is in the Appendix.

This framework for macro-pattern discovery leads naturally to an algorithm for finding templates in real data, which is to minimize the empirical risk, regularized by the number of backwards arrows and the length of the template.

3.2 Macro-Patterns in Meetings

Let us explain the data processing. We selected the annotations that were meaningful in this context, namely: socially positive act, socially negative act, negative assessment act and positive assessment act. This allows us only to focus on assessments (either social or work related), as they may have a generally more powerful effect on the trends in the conversation than other dialogue acts. That is, the assessments create a "macro-pattern" within the meeting that we want to learn. Repeated dialogue acts by the same person were counted as a single dialogue act. The selected data contain 12,309 dialogue acts, on average 130 acts per meeting.

We would like to know if all or most meetings have something in common, in terms of the pattern of dialogue acts throughout the meeting (e.g., a smaller sub-conversation in which certain dialogue acts alternate, followed by a shift to a different sub-conversation, and so on). If this were the case, a template for meetings could be a directed graph, where each meeting approximately follows a (possibly repetitive) path through the graph. Time might loosely follow from left to right. We thus learn the directed graph by optimizing the following regularized risk functional using discrete optimization:

$$\begin{aligned} &F(\text{template } t) \\ &= \frac{1}{m} \sum_{\text{meetings } i} \min_{t_j \in \text{instan}(t)} [\text{dist}(t_j, \text{meeting } i)] + C_1 \text{length}(t) + C_2 \text{backw}(t), \quad (1) \end{aligned}$$

which is a regularized version of the empirical risk defined above. Intuitively, (1) characterizes how well the set of meetings match the template using the first term, and the other two terms are regularization terms that force the template to be sparser and simpler, with most of the edges pointing forwards in time. The smaller templates have the advantage of being more interpretable, also according to the bound above, smaller templates encourage better generalization. The length(t) is the number of nodes in template t. The value of backw(t) is the number of backwards directed edges in the template graph. We chose $C_1 = 1$ and $C_2 = 0.1$ to force fewer backwards edges than forwards edges, meaning a graph that follows more linearly in time.[2]

Template instantiations always follow paths that exist within the template, whereas a real meeting will likely never exactly follow a template (unless the template is overly complex, in which case the algorithm has wildly overfit the data, which is prevented by validation). Our calculation for $\min_{t_j \in \text{instan}(t)} [\text{dist}(t_j, \text{meeting } i)]$ in the first term of (1) is approximate, in the sense that the min over all templates is calculated over all template instantiations that are approximately the same length as the meeting i, rather than over all instantiations. As it is likely the minimum would be achieved at an instantiation with approximately the same length as the meeting, this is a reasonable assumption to make in order to make the calculation more tractable.

We optimize (1) over templates using simulated annealing, as shown in Algorithm 1. Simulated annealing probabilistically decides whether it will move to a neighboring state or stay at the current state at each iteration. The neighborhood is defined as a set of templates that are edit distance 1 away from the current template under allowable operations. The allowable operations include insertion of a new node between any two nodes or at the beginning or end of the template, deletion of a node, insertion or deletion of backwards directed edges. For the proposal distribution for simulated annealing, each operation is randomly chosen with uniform probability. The acceptance probability of the new template is 1 if the proposed objective function value is less than the current value. If the proposed value is larger than the current value, the acceptance function accepts with probability $\exp(-\Delta F/T)$ where ΔF represents the difference in objective function value, namely the proposed function value minus the current function value, and T is the current temperature (lines 19-27 in Algorithm 1). The annealing schedule is $T = T_0 \cdot 0.95^k$, where $k = 800$ is the annealing parameter and $T_0 = 1000$ is the initial temperature. After every k accepted steps, we restarted the optimization starting at the best point so far, resetting the temperature to its initial value. The maximum iteration number was set at 4000.

We ran the algorithm starting from 95 different initial conditions, each initial condition corresponding to one of the true meetings in the database. These 95 experiments were performed in order to find a fairly full set of reasonable templates, and to also determine whether the algorithm consistently settled on a small set of templates. The results were highly self-consistent, in the sense that in 98% of the total experimental runs, the algorithm converged to a template that is equivalent to, or a simple instantiation of, the one shown in Figure 2. This template has a very simple and strong message, which is that the next judgment following a negative assessment is almost never a socially positive act.

[2] We also varied C_1 and C_2 and the results were consistent. While doing so, we kept C_1 below 5 so that each reduction in the size of the template costs us no more than 5 in edit distance. We kept C_2 below 0.5 to ensure that we would not remove a backwards arrow to sacrifice accuracy.

Algorithm 1 Discrete optimization algorithm for learning meeting templates

```
Input: Starting template $t_0$, $m$ meetings, constants $C_1$, $C_2$, initial temperature $T_0$.

Initialize $t = t_0$, $t_{\text{best}} = t_0$, iter $= 0$, $k = 800$, set $F(t_o)$.
while (!converged) $\cap$ (iter $<$ Maximum Iteration) do
    if mod(iter) $= k$ then
        $T \leftarrow T_0$ where $T_0$ is initial temperature, $t \leftarrow t_{\text{best}}$
    end if
    $N_t \leftarrow$ get_neighborhood_templates($t$)
    choose $t^{'} \in N_t$ uniformly at random
    length($t^{'}$) $\leftarrow$ number of nodes in template $t^{'}$
    backw($t^{'}$) $\leftarrow$ number of backwards edges in template $t^{'}$
    for all meetings $i \leq m$ do
        instan($t^{'}$) $\leftarrow$ set of template instantiations (sufficient to use only ones with
        length $\approx$ length of meeting $i$)
        for all instantiations $t_j \in$ instan($t^{'}$) do
            $d_{ij}^{t^{'}}$ = edit_distance($t_j$, meeting $i$)
        end for
        $d_i^{t^{'}} \leftarrow \min_j(d_{ij}^{t^{'}})$
    end for
    $F(t^{'}) \leftarrow \frac{1}{m} \sum_i d_i^{t^{'}} + C_1\text{length}(t^{'}) + C_2\text{backw}(t^{'})$
    if $F(t^{'}) < F(t)$ then
        $t \leftarrow t^{'}$, accept jump with probability 1, iter $\leftarrow$ iter $+ 1$, $F(t) \leftarrow F(t')$
        if $F(t^{'}) < F(t_{\text{best}})$ then
            $t_{\text{best}} \leftarrow t^{'}$
        end if
    else
        $t \leftarrow t^{'}$ with probability $\exp(-\Delta F/T)$ where
        $\Delta F = F(t^{'}) - F(t)$. If accepted, then iter $\leftarrow$ iter $+ 1$, $F(t) \leftarrow F(t')$
    end if
    $T \leftarrow T_0 * 0.95^{\text{iter}}$ (update temperature)
end while
```

The converse is also true, that a socially positive act is rarely followed by a negative assessment.

To assess the correctness of this template, we note that of the 1475 times a socially positive act appears, it is adjacent to a negative assessment 141 times. Of the 991 times a negative assessment appears, 132 times a socially positive act is adjacent to it. One can contrast these numbers with the percent of time a positive assessment is adjacent to a socially positive act (68%), though there are generally more positive assessments than either socially positive acts or negative assessments, which intrinsically lowers the probability that a socially positive act would be adjacent to a negative assessment; however, even knowing this, there could be more powerful reasons why socially positive acts are associated with positive assessments rather than negative assessments.

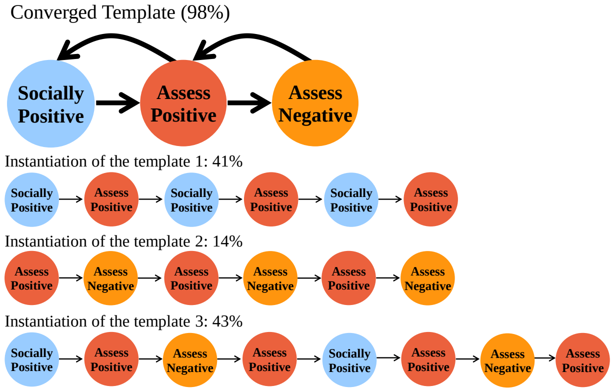


Fig. 2. The template representing the interaction of social and work related acts. Specific template instantiations and how often they occurred in the experiment are also provided.

From a social perspective, this result can be viewed as somewhat counterintuitive, as one might imagine wanting to encourage a colleague socially in order to compensate for a negative assessment. In practice, however, sometimes positive social acts can sound disingenuous when accompanying a negative assessment. This can be demonstrated using the following pairs of dialog from within the AMI Corpus (that are not truly side by side): 1) "But I thought it was just completely pointless." "Superb sketch by the way." 2) "Brilliantly done." "It's gonna inevitably sort of start looking like those group of sort of ugly ones that we saw stacked up." 3) "It'd be annoying." "Yeah, it was a pleasure working with you." 4) "No, a wheel is better." "All thanks to Iain for the design of that one." In these cases a positive social act essentially becomes a negative social act.

The algorithm introduced here found a macro pattern of assessments that is prevalent and yields insight into the interaction between social and work related dialogue acts. This demonstrates that machine learning can be used to illuminate fundamental aspects of this domain. This knowledge is general, and could be used, for instance, to help classify assessments when they are not labeled, and to potentially help reconstruct missing/garbled pieces of dialogue. These are necessary elements in building any type of meetings analysis assistant. To our knowledge, the macro pattern we found has not been previously noted in the literature.

3.3 Experimental Comparison with Other Methods

In this section, we compare our results with other methods, particularly Profile Hidden Markov Model (HMM) and Markov chain. We focus on comparison of results in this section, however, further in depth discussion is provided in Section 6.

Profile HMM is a tool to find and align related sequences, and match a new sequence to known sequences. The structure of a Profile HMM has forward arrows between match states, along with delete and insertion nodes that allow any string to fit into the template. Profile HMM shares similar features with our algorithm, in that it finds a common sequence given sequences with different lengths. To create a Profile HMM, there are a series of heuristic steps. One way of learning Profile HMM, used to produce Table 3, is the following. First, we choose the length of the HMM. Second, we estimate the parameters in the HMM model using pseudo-counts for the prior. Third, we find the most likely string to be emitted from each match state.

As mentioned above, the length of a Profile HMM is generally specified by the user. Table 3 shows Profile HMMs with a variety of lengths specified. We were not able to recover the full pattern with Profile HMM that our method produced, but we were able to see, for instance, that socially positive acts are often next to positive assessments, which is one of the instantiations of the template our method discovered.

Table 3. Profile HMM (SP: Socially positive, AP: Assess positive, AN: Assess negative)

Length specified	**Result**
3	SP → AP → AP
5	SP → AP → SP → AP → AP
10	SP → AP → SP → AP → AP → AP → AP → AP → AP → AP
20	SP → AP → SP → AP → AP → AP → AP → AP → AP → AP → AP → AP → AP → AP → AP → AP → AP → AP → AP → AP

We note that a first-order Markov chain could be used to model transitions between nodes, however, since a Markov chain records all possible transition probabilities, it is not clear in general how to turn this into a template that would yield insight. On the other hand, once we provide a template like the one in Figure 2, it is easy to see the template within the Markov chain. The first-order Markov chain for the meetings data, learned using maximum likelihood estimation, is provided in Figure 3. This learned Markov chain supports our learned template, as the four highest transition probabilities exactly represent our learned template — with high transition probabilities between socially positive and assess positive, and between assess negative and assess positive. The reason that socially negative did not appear in Figure 2 (as it should not) is because the number of socially negative dialogue acts is much smaller than the number of other dialogue acts (only 1%). This is an important difference between a Markov chain and our algorithm — our algorithm picks the dominant macro pattern, and has the ability to exclude a state (i.e. dialogue act) if there is no strong pattern involving that state. The Markov chain, on the other hand, is required to model transition probabilities between all states that exist within the data. We also note that it is possible for our method to choose forward arrows with very low transition probabilities in the Markov chain, so looking at only the high probability transitions will not suffice for constructing a template that minimizes our risk functional.

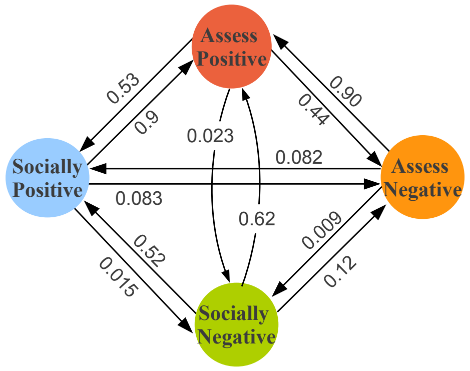
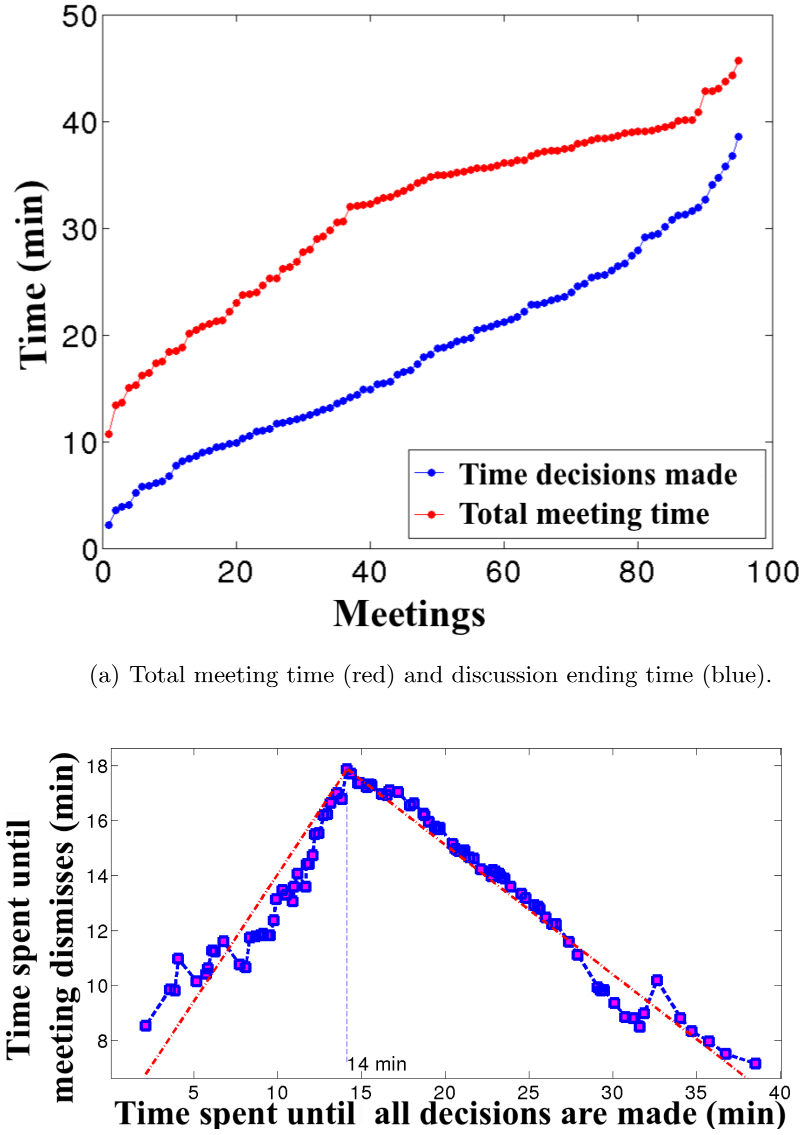


Fig. 3. Learned Markov Chain

4 Question iii: How long is this meeting going to last given that the decision has already been made?

Meetings sometimes last longer than expected. Even when all the decisions seem to be made, it often takes some time to work out the details and formally end the meeting. We would like to know whether it is possible to predict when the meeting is going to be over if we know when the decisions are made.

Figure 4(a) displays meetings along the horizontal axis, ordered by total meeting time. The annotated time when key decisions are all made (discussion finishing times) are indicated by blue squares, and the total meeting times are indicated by red squares for each meeting. Figure 4(b) shows the time between when key decisions are made on the x-axis, and the wrap-up time on the y-axis (this is the time spent after key decisions are made and before the meeting finishes). With a simple piecewise linear formula, one can predict with relatively high accuracy what the wrap-up time will be, given the time at which the key decisions are finished being made. Denoting the time to complete key decisions



(a) Total meeting time (red) and discussion ending time (blue).

(b) “Wrap-up” time vs. time until decisions are made.

Fig. 4. Total meeting time and wrap-up time

by x, the estimated wrap-up time is as follows: If $x \leq$ about 14 minutes, the wrap-up time is about $0.923 + 4.78$ minutes. If $x >$ about 14 minutes, the wrap-up time is $-0.47x + 24.53$ minutes. There are some interesting implications: if the meeting was efficient enough to make all the decisions within 14 minutes, then the team will also be efficient in wrapping up. Once the meeting has gone about 14 minutes without all decisions made, then people also tend to spend more time wrapping up. If the meetings runs very long without decisions being made, then once the decisions are made, the meeting tends to wrap up quickly.

These results are specific to meetings whose length is less than an hour as in the AMI corpus, which, according to Romano Jr and Nunamaker Jr [2001] is true 26% of the time; it would be interesting for future work to see if a piecewise linear model works well for meetings of other lengths, though this hypothesis cannot currently be verified by any database that we are aware of.

5 Question iv: Do persuasive words exist?

A "good" meeting might be one where we have contributed to the team effort. We always want to suggest good ideas and want the team members to accept those ideas. Numerous articles claim that how we package our ideas, and our choice of words, is sometimes as important as the idea itself [Olsen, 2009, Carlson, 2012]. We are interested in understanding this hidden factor in the team's decision making process. Are there patterns in suggestions that are accepted versus rejected? Can we use this to improve how we present ideas to the team?

To select data for this section, we chose a bag-of-words representation for each "suggestion" dialogue act. We gathered a set of all words that occurred in all suggestions, excluding stop words, leading to a 1,839 dimensional binary vector representation for each suggestion. The labels were determined by the annotations, where accepted suggestions received a $+1$ label, and rejected suggestions received a -1 label.

5.1 Are published sets of persuasive words really persuasive?

We would like to know whether the persuasive words of Olsen [2009], Carlson [2012] are truly persuasive. These words are not domain-specific, meaning that they do not depend on the topic of the meeting, and could thus be more generally useful. Note that the data cannot tell us (without a controlled experiment) whether there is a truly causal relationship between using persuasive words and having a proposal accepted; however, we can study correlations, which provide evidence. An important first question is whether the proportion of persuasive words in accepted suggestions differ significantly from the proportion of persuasive words in rejected suggestions. We marked each suggestion as to whether or not one of the words in Olsen [2009], Carlson [2012] appears. We also mark each suggestion as to whether it is an accepted or rejected suggestion. Of the 139 times that a suggestion contained persuasive words from Olsen [2009], Carlson

[2012], 134 of these appearances were within accepted suggestions (96% of appearances). Of the 2,185 times a suggestion did not contain any of the words from Olsen [2009], Carlson [2012], 1,981 of these appearances were within accepted suggestions (90% of appearances). Using Fisher's exact test, the difference in proportions is significant at the 0.01 level (pvalue 0.0037); it appears that persuasive words do appear more often in accepted suggestions than other words do.

On the other hand, we tested each persuasive word individually, to see whether the number of accepted suggestions it appears in is significantly different than the proportion of rejected suggestions it appears in. The difference between the two proportions was not significant at the 0.05 level for any of the persuasive words, again using Fisher's exact test for differences between proportions. Perhaps this is because each word individually is rare, while the collection is not. This begs the question as to whether there are words that themselves show a significant difference in the proportion of accepted suggestions they appear in. A related question is which words are the most important, in terms of predictive modeling, for distinguishing accepted and rejected suggestions. We will answer both of these questions below, the latter answered first.

5.2 Using SVM coefficients to tell us about persuasive words

In this section, we try to discover persuasive words from data alone, and afterwards compare to the published lists of persuasive words. Specifically, we want to know whether we can predict if a suggestion would be accepted or rejected solely based on words that are used, not based on the idea within it. A standard way to do this is to apply a linear SVM with cross-validation for the tradeoff parameter, and examine the values of its coefficients; this provides a ranking of features, and a rough measure of how important each word is to the overall classifier [e.g., see Guyon et al., 2002]. The set of 2,324 suggestions with 1,839 features (one feature per unique word) was separated into 5 folds using each in turn as the test set, and the SVM accuracy was 83% ± 2.1%. Note that the SVM has no prior knowledge of what persuasive words are. Persuasive words identified by the SVM are considered to be words with large absolute coefficient value (normalized) over all 5 folds. These include "things," "start," "meeting," "people," "yeah." Many of the persuasive words concerned the topic of marketing ("market," "presentation," "gimmick," "logo") and non-persuasive words included "buttons," "speech," "LCD," "recognition," and more often words specific to the topic of creating a remote control ("scroll," "green").

The next question is whether these learned persuasive words make sense. To answer this, we checked whether the persuasive words from Olsen [2009], Carlson [2012] had positive or negative SVM coefficients. The overlap between our set of 1,839 words and the persuasive words from Olsen [2009], Carlson [2012] is 29 words. Their average coefficient values over 5 folds are shown in Table 4. The finding is consistent with the Fisher's exact test result in Section 5.1 — not a single word has a large positive coefficient and all words have relatively large standard deviations, showing lack of their individual connection to persuasiveness.

Table 4. List of persuasive words on which SVM and the articles [Olsen, 2009, Carlson, 2012] agree/disagree

Agrees		**Disagrees**	
Words	λ ± **Std.**	**Words**	λ ± **Std.**
strength	0.006 ± 0.0026	inspiration	-0.0002 ± 0.003
free	0.0054 ± 0.0031	drive	-0.0002 ± 0.0011
good	0.0048 ± 0.0015	easy	-0.0004 ± 0.003
power	0.0043 ± 0.0032	health	-0.0005 ± 0.0022
avoid	0.0039 ± 0.0011	creativity	-0.0016 ± 0.0021
offer	0.0036 ± 0.0035	guarantee	-0.0017 ± 0.001
save	0.003 ± 0.0036	explore	-0.0017 ± 0.001
safe	0.0028 ± 0.0015	safety	-0.0017 ± 0.001
energy	0.0024 ± 0.0032	reinvent	-0.0017 ± 0.0009
imagine	0.0024 ± 0.0016	approach	-0.0019 ± 0.003
important	0.0023 ± 0.002	money	-0.0025 ± 0.0035
confidence	0.0019 ± 0.0027	purpose	-0.0027 ± 0.0008
wanted	0.0015 ± 0.0009		
quick	0.0015 ± 0.0009		
memory	0.0015 ± 0.0009		
life	0.0006 ± 0.0029		
hurry	0.0005 ± 0.0029		

In what follows we will explore a different method for feature ranking, where the space is reduced to only words that occur in significantly different proportions between accepted and rejected suggestions before applying an SVM.

5.3 Which words are individually persuasive?

We have 2,324 suggestions in an 1,839 dimensional space. Fisher's exact test is based on the hypergeometric distribution and has the benefit that it can be used for features that are rarely present. It considers whether the suggestions containing the feature have significantly different proportions of acceptance and rejection. The selected persuasive words are shown in Table 5 with their associated pvalues, where the most significant words are at the top of the list. We can also find non-persuasive words that are significant according to Fisher's exact test. Together with the persuasive words, this gives us a set of important features to use as a reduced feature space for machine learning. Some of the non-persuasive words include "recognition", "speech", "fair", "selecting", "flat", "animals", "middle", and "bottom". The total number of features we used is 244 (where persuasive words and non-persuasive words were selected under the same significance threshold). After this feature reduction, we applied SVM and achieved an accuracy of 87.2% ± 0.010%, which is higher than before. The SVM coefficients for all of the words that we identified as being persuasive are positive, as shown also in Table 5. These words are thus persuasive both individually and together: they are each individually significant in predicting accepted suggestions, and they have positive predictive coefficients.

Table 5. List of persuasive words from Fisher's exact test (higher ranking for smaller pvalues). The second column contains the pvalue from Fisher's exact test. The third column contains the ratio of accepted proposals when the word appeared. The fourth column contains the ratio of accepted proposals when the word did not appear. The last column contains the SVM coefficients.

Words	**Pvalues**	**Ratio of accepted when appears**	**Ratio of accepted when not appears**	**SVM λ ± Std.**
yeah	0.012	1 (46/46)	0.90 (2069/2278)	0.011 ± 0.0021
give	0.020	1 (41/41)	0.90 (2074/2283)	0.010 ± 0.0015
menu	0.027	1 (38/38)	0.90 (2077/2286)	0.010 ± 0.0045
start	0.043	1 (33/33)	0.90 (2082/2291)	0.012 ± 0.0020
meeting	0.052	1 (31/31)	0.90 (2084/2293)	0.012 ± 0.0019
touch	0.085	1 (26/26)	0.90 (2089/2298)	0.0081 ± 0.00059
discuss	0.093	1 (25/25)	0.90 (2090/2299)	0.0092 ± 0.0024
find	0.093	1 (25/25)	0.90 (2090/2299)	0.0098 ± 0.0017
market	0.12	1 (22/22)	0.90 (2093/2302)	0.0094 ± 0.0023
yellow	0.12	1 (22/22)	0.90 (2093/2302)	0.0076 ± 0.0018
work	0.12	1 (22/22)	0.90 (2093/2302)	0.0097 ± 0.0031
good	0.13	0.97 (37/38)	0.90 (2078/2286)	0.0029 ± 0.0035
fruit	0.13	1 (21/21)	0.90 (2094/2303)	0.0065 ± 0.0032
logo	0.15	1 (20/20)	0.90 (2095/2304)	0.0073 ± 0.0019
people	0.15	0.97 (35/36)	0.90 (2080/2288)	0.012 ± 0.0024
side	0.16	0.97 (34/35)	0.90 (2081/2289)	0.0056 ± 0.0039
number	0.16	1 (19/19)	0.90 (2096/2305)	0.0059 ± 0.0027
presentation	0.18	1 (18/18)	0.90 (2097/2306)	0.0084 ± 0.0020
things	0.19	0.95 (45/47)	0.90 (2070/2277)	0.012 ± 0.0026
chip	0.20	1 (17/17)	0.90 (2098/2307)	0.0066 ± 0.0026
stick	0.22	1 (16/16)	0.90 (2099/2308)	0.0070 ± 0.0017
gonna	0.22	0.95 (42/44)	0.90 (2073/2280)	0.000063 ± 0.0057
information	0.24	1 (15/15)	0.90 (2100/2309)	0.0059 ± 0.0020
talk	0.24	1 (15/15)	0.90 (2100/2309)	0.0034 ± 0.0027

Studying the most persuasive words from Fisher's exact test, we find that many of these words are *not* specifically tied to the topic of the meeting (designing a remote control), but seem to be more generally persuasive. In fact, we can make informed hypotheses about *why* these words are persuasive. Some of these observations help us to understand more generally how language is used during meetings. Let us consider the most significant persuasive words as follows:

- *Yeah:* Dialogue segments where the word "yeah" is used include: "or yeah, maybe even just a limited multi-colour so it doesn't look too childish," "yeah, if you had one of those, just coming back to your other point about pressing the button and setting off the bleeper in the room," "Yeah if you are holding it in your hand you could do that." Judging from these and similar dialogue segments, our hypothesis is that framing a suggestion as an agreement with a previous suggestion increases its chances of being accepted. That is, if the idea comes across as if it were in line with previous thoughts by others, the suggestion has a higher chance of being accepted. This applies either when attributing the full idea to others, or just the line of thought. The case where one attributes their full idea to others in order to increase its chances of acceptance has been considered in popular books Carnegie [1936].
- *Give:* "Give" is used in at least three ways, the first one occurring the most often: (i) giving with respect to the topic of the meeting, which here is either the customer or the product ("so if you want to give the full freedom to the user," "You give it the full functions in here," "We can give them smooth keys"), (ii) giving to the meeting participants ("would give us a little bit of a marketing niche"), and (iii) to indicate that suggestions are based on previous data or knowledge ("given these parameters that we're just gonna sort of have this kind of uh non-remote remote", "given speech recognition I think you should go for the less fancy chip").
- *Menu and Touch:* These seems to be tied to the topic of remote controls without a general hypothesis for other types of meetings: "So we put a touch pad on it," "We can uh do a touch-pad on our remote," "Um and one other suggestions I'd make is to in is to include in a menu system."
- *Start:* Our hypothesis is that the word "start" gives group members the opportunity to agree, where agreement of basic suggestions provides an indication that group members want to be productive during the meeting; e.g., "Shouldn't we start with the most important parts?" "I will start by the basic one." This type of agreement may help with alliance building early on in the meeting.
- *Meeting:* The word "meeting" often appears in suggestions about what *not* to discuss: "Or maybe this is something for the next meeting," "I figure we could get back to it on the next meeting actually," "We take it to the other meeting." Our hypothesis is that suggesting that a topic belong to a later meeting may be a way to gently change the topic or move the current meeting along. It can be used instead of a negative assessment of a previous suggestion.
- *Discuss:* This word appears mainly in an organizational context for the meeting: "And then we can discuss some more closely," "I think we shouldn't discuss any points points that long," "Maybe we should centralise the discussion here." It seems that often, people tend to agree with organizational suggestions about the meeting.

- *Find:* Our hypothesis is that "find" is often used in suggestions to gather more information or do more work, and these are suggestions that are often accepted: "we have to find out if it's possible," "um and I'm sure we can find more goals for the product we are going to develop," "but just try to find out what they're willing to pay for it."

Having a collection of persuasive words can be immediately useful, assuming that there is a causal relationship between persuasive words and accepted proposals, rather than only a correlation. The hypothesis tested in this work (that persuasive words truly exist) can also be tested for this causal relationship, and it would be very interesting to create a dataset for this purpose. If the hypothesis does hold, it has the potential to allow ideas to be communicated more clearly, and thus to make meetings more efficient overall. Already we have gained some insight for how specific words are used within meetings, and why suggestions containing these words are more likely to be accepted.

6 Related Work

The closest work to ours is that of the CALO (Cognitive Assistant that Learns and Organizes) project [SRI International, 2003-2009], which takes a *bottom-up* approach to designing a cognitive assistant that can reason and learn from users using machine learning techniques. Although the CALO project's focus (e.g., speech recognition, sentence segmentation, dialogue act tagging, topic segmentation, action item detection and decision extraction) does not intersect with our work, it is worthwhile to note that this multi-year project has made a definite step towards improving meetings using machine learning techniques. However, as pointed out by Tur et al. [2010], a number of challenges (e.g. extracting task descriptions) still exist before this tool can be integrated into daily life. The insights into meetings obtained through our work could be used as part of a *top-down* approach to design such a tool.

We believe our work is the first to take a truly data-driven approach to finding persuasive words; this is the first work to try to prove that a word is persuasive using data.

Qualitative Work: Research on group decision making processes traditionally appears in *qualitative* studies. Works along these lines attempt to understand and model the course of agreement [Black, 1948], design features to track the course of negotiation [Eugenio et al., 1999], develop theories for group decision making [Davis, 1973] and study different group decision-making schemes [Green and Taber, 1980].

Related Work for Question i: A few quantitative approaches have attempted to detect when decisions are made. These methods use maximum entropy [Hsueh and Moore, 2008], linear SVMs [Fernández et al., 2008] and directed graphical models [Bui et al., 2009]. A common factor of these works is that they require content information (audio and video data; prosodic features, position within the meeting, length in words and duration etc.) that can potentially contain sensitive information about what is being planned. Our approach requires only dialogue acts (i.e. the actual meeting content is not required) to achieve the same goal with relatively high accuracy. One benefit of using only dialogue acts is that the algorithm allows our results to hold independently of the specific type and purpose of the meeting being held and can be used in situations where the meeting contains potentially sensitive information. It is also much simpler logistically to require collecting only dialogue acts.

Related Work for Question ii: To the best of our knowledge, our work is the first to apply learning techniques to learn dynamical interactions between social and work aspects in meetings. As discussed earlier, one could consider the method we developed generally for learning a template from a sequence of data. Our method has the following characteristics: 1) The loss function considers meetings instantiated from the same template, but possibly with different lengths, to be equally good. For instance: if a template is 'A' 'B' 'C' with a back arrow from 'C' to 'A', then 'ABCABC', 'BCABCA', 'CAB' are equally good instantiations from this template. 2) An instantiation of the template can start at any position, as demonstrated by the 'BCABCA' and 'CAB' instances in the template above. 3) By optimizing the loss function we provide in Section 3, the algorithm is able to uncover a template of exactly the desired form, despite the existence of noise. Let us compare this method to other graph learning techniques within related subfields.

Petri nets are used (and were designed) to model concurrency and synchronization in distributed systems. In learning Petri nets from data, and generally in learning workflow graphs and workflow nets (see van der Aalst [1998]), all possible transitions in the data must be accounted for within the learned net (e.g., Agrawal et al. [1998]), which is the opposite of what we would like to do, which is to create instead a more concise representation that is not necessarily all-inclusive.

One major difference between our approach to learning macro-patterns and learning probabilistic graphical models such as Markov models is that our approach is deterministic (not probabilistic). We do not model the probabilities of transitions between states, as these probabilities are not of fundamental interest for building the template (but could be calculated afterwards). As in other deterministic methods (e.g., SVM), we aim to directly optimize the quality measure that the macro-pattern will be judged by on the data, in our case involving edit distance. This allows us to handle noisy data, that is, meetings that do not fit precisely into the pattern, without having to include additional nodes that complicate the template. We do not then need a graph that handles all possible transitions and their probabilities (as is required in Markov models or Petri nets). It is certainly possible to model a macro-pattern as a Markov chain, as we demonstrated, but the resulting transition matrix would most likely provide little insight. The full Markov transition matrix would be of size $|\Lambda| \times |\Lambda|$, but distilling this to a compact one dimensional graph with mostly forward arrows would then be an additional task.

Hidden Markov Models aim to infer unobserved states, where in our case, there are no natural unobserved states. Although we could artificially create hidden states, it is more natural to directly model the observed sequence. One exception to this is Profile HMM's, where hidden states are either "match" states, "insert" states, or "delete" states [Eddy, 1998]. Methods for fitting Profile HMM's generally do not create backwards edges, and thus cannot easily accommodate substrings being repeated arbitrarily within the template. Profile HMM's are designed for accuracy in aligning sequences, but they are not generally designed for conciseness or to minimize, for instance, a count of backwards edges. Methods for learning Profile HMMs generally require the user to specify the length of the HMM, whereas we purposely do not do this.

Profile HMM's generally have a fixed initial state, whereas our method can start anywhere within the template. The price of a fixed initial state and fixed length with no backwards edges is high; for instance for a template 'A' 'B' 'C' with a backwards arrow from 'C' to 'A,' we can equally well accommodate patterns 'ABCABCABC' and 'BCABCA,' as well as 'BC', all of them being perfect matches to our template. Profile HMM's, with any fixed initial state, and with no backwards arrows to allow repeats, would require insertions and deletions for each of these patterns, and have difficulty viewing all of these patterns equally. Note that in general, left-to-right HMM's cannot have backwards loops as our templates do.

Probabilistic Suffix Automata (PSA) [Ron et al., 1996] are probabilistic automata with variable memory length, that aim to learn an order-K Markov model with K varying in different parts of state space. As with order-1 Markov chains discussed above, is not clear how to construct a concise template of the form we are considering. On the other hand, it would be an interesting extension of our work to keep track of higher order patterns within the template like PSA does.

One could think of our goal in Section 3 as solving a type of case of the consensus string problem [Sim and Park, 2003, Lyngsø and Pedersen, 2002] with consensus error, but with two major changes: (i) using edit distance between meetings and template instantiations as a metric, rather than between pairs of strings, which allows backwards loops, (ii) encouraging conciseness in the template. Problems encountered when not including these aspects were provided for the 'ABC' example above.

Related Work for Question iv: Identifying characteristics of persuasive speech and discourse has been of great interest in the qualitative research community [Sternthal et al., 1978, Scheidel, 1967]. However, there has not been much quantitative research done on this topic. Guerini et al. [2008] studied the relationship between the choice of words and the reaction they elicit in political speeches based on numerical statistics, including counts of words in each document and how common the words are across all documents (tf-idf). To the best of our knowledge, our work is the first work to apply machine learning techniques to learn persuasive words from free-form conversational meeting data, to rank persuasive words, and to compare resulting persuasive words with qualitative studies to gain insights.

Other meeting related work: Other studies that apply machine learning techniques for meeting related topics (but not specifically related to any of the work in this paper) include meeting summarization [Purver et al., 2007], topic segmentation [Galley et al., 2003], agreement/disagreement detection [Hahn et al., 2006, Hillard and Ostendorf, 2003, Bousmalis et al., 2009], detection of the state of a meeting (discussion, presentation and briefing) [Banerjee and Rudnicky, 2004, Reiter and Rigoll, 2005], and prediction of the roles of meeting participants [Banerjee and Rudnicky, 2004]. In all of this work, machine learning techniques are used as tools to classify a particular aspect of a meeting with a specific applications in mind. In addition to addressing different aspects of meetings, our work uses machine learning techniques as a way to to learn and study meetings scientifically in an attempt to bridge the gap between qualitative studies on understanding meetings and quantitative application-focused studies.

7 Conclusion

The scientific field of meeting analysis is still in beginning stages; meetings have not yet been well characterized, and this is one of the first works in this new scientific arena. Several hypotheses made in this work *cannot* be tested further with any current available dataset. We hope that by illuminating the potential of fully solving these problems, it will inspire the creation of new meetings corpora. Elaborating further, if we are able to automatically detect when key decisions are made, this could translate directly into a software tool that managers could use to determine when they should join an ongoing meeting of their staff without attending the full meeting. If we know common patterns of dialogue, this might help us to understand social cues better for business settings, and could potentially help reconstruct parts of dialogue that might not have been recorded properly. If we can automatically detect when a meeting's key decisions are made, and can accurately gauge the meeting wrap-up time, it can give us something immediately valuable, namely an estimated time for the end of the current meeting, which staff can use to plan ahead for the start of the next meeting, or to plan transportation, in a fast-paced corporate culture. If we truly knew which words were persuasive, we could use these words to help convey our ideas in the most favorable light. Before we can do all of this, however, we need to understand the science behind meetings, and make hypotheses that can be tested, which is the goal of this work.

Bibliography

R. Agrawal, D. Gunopulos, and F. Leymann. *Mining process models from workflow logs.* Springer, 1998.

S. Banerjee and A.I. Rudnicky. Using simple speech based features to detect the state of a meeting and the roles of the meeting participants. In *Proceedings of International Conference on Spoken Language Processing*, pages 221–231, 2004.

D. Black. On the rationale of group decision-making. *Journal of Political Economy*, 56(1):23–34, 1948.

K. Bousmalis, M. Mehu, and M. Pantic. Spotting agreement and disagreement: A survey of nonverbal audiovisual cues and tools. In *Proceedings of 3rd International Conference on Affective Computing and Intelligent Interaction and Workshops*, pages 1–9, 2009.

T. H. Bui, M. Frampton, J. Dowding, and S. Peters. Extracting decisions from multi-party dialogue using directed graphical models and semantic similarity. In *Proceedings of the 10th Special Interest Group on Discourse and Dialogue Workshop on Discourse and Dialogue*, pages 235–243, 2009.

G. Carlson. Uplifting word list: `http://www.freelance-copy-writing.com/Uplifting-Word-List.html`, 2012.

D. Carnegie. *How to win friends and influence people.* Simon & Schuster, 1936.

J. H. Davis. Group decision and social interaction: A theory of social decision schemes. *Psychological Review*, 80(2):97–125, 1973.

S. R. Eddy. Profile hidden markov models. *Bioinformatics*, 14(9):755–763, 1998.

S. Ertekin and C. Rudin. On equivalence relationships between classification and ranking algorithms. *Journal of Machine Learning Research*, 12:2905–2929, 2011.

B. D. Eugenio, P. W. Jordan, J. H. Thomason, and J. D. Moore. The agreement process: An empirical investigation of human-human computer-mediated collaborative dialogues. *International Journal of Human Computer Studies*, 53 (6):1017–1076, 1999.

R. Fernández, M. Frampton, P. Ehlen, M. Purver, and S. Peters. Modeling and detecting decisions in multi-party dialogue. In *Proceedings of the 9th Special Interest Group on Discourse and Dialogue Workshop on Discourse and Dialogue*, pages 156–163, 2008.

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

S. G. Green and T. D. Taber. The effects of three social decision schemes on decision group process. *Organizational Behavior and Human Performance*, 25 (1):97–106, 1980.

M. Guerini, C. Strapparava, and O. Stock. Trusting politicians' words (for persuasive NLP). *Computational Linguistics and Intelligent Text Processing*, pages 263–274, 2008.

I. Guyon, J. Weston, S. Barnhill, and V. Vapnik. Gene selection for cancer classification using support vector machines. *Machine Learning*, 46(1-3):389–422, March 2002.

S. Hahn, R. Ladner, and M. Ostendorf. Agreement/disagreement classification: Exploiting unlabeled data using contrast classifiers. In *Proceedings of the Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics*, pages 53–56, 2006.

J. Hall. Americans know how to be productive if managers will let them. *Organizational Dynamics*, 1994.

D. Hillard and M. Ostendorf. Detection of agreement vs. disagreement in meetings: Training with unlabeled data. In *Proceedings of the Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics*, 2003.

P. Y. Hsueh and J. Moore. Automatic decision detection in meeting speech. *Machine Learning for Multimodal Interaction*, pages 168–179, 2008.

G. Ji and J. Bilmes. Dialog act tagging using graphical models. In *Proceedings of International Conference on Acoustics, Speech, and Signal Processing*, volume 1, pages 33–36, 2005.

T. A. Kayser. *Mining group gold: How to cash in on the collaborative brain power of a group.* Serif Publishing, 1990.

R. B. Lyngsø and C. N. S. Pedersen. The consensus string problem and the complexity of comparing hidden markov models. *Journal of Computer and System Sciences*, 65(3):545–569, 2002.

A. Mackenzie and P. Nickerson. *The time trap: The classic book on time management.* Amacom Books, 2009.

I. McCowan, J. Carletta, W. Kraaij, S. Ashby, S. Bourban, M. Flynn, M. Guillemot, T. Hain, J. Kadlec, V. Karaiskos, et al. The AMI meeting corpus. In *Proceedings of Methods and Techniques in Behavioral Research*, 2005.

M. Nagata and T. Morimoto. First steps towards statistical modeling of dialogue to predict the speech act type of the next utterance. *Speech Communication*, 15(3-4):193–203, 1994.

D. Newlund. Make your meetings worth everyone's time: `http://www.usatoday.com/USCP/PNI/Business/2012-06-20-PNI0620biz-career-getting-aheadPNIBrd_ST_U.htm`, 2012.

H. Olsen. 108 persuasive words: `http://www.high-output.com/uncategorized/108-persuasive-words/`, 2009.

R. R. Panko and S. T. Kinney. Meeting profiles: Size, duration, and location. In *Proceedings of the Twenty-Eighth Hawaii International Conference on System Sciences*, volume 4, pages 1002–1011, 1995.

M. Purver, J. Dowding, J. Niekrasz, P. Ehlen, S. Noorbaloochi, and S. Peters. Detecting and summarizing action items in multi-party dialogue. In *Proceedings of the 8th Special Interest Group on Discourse and Dialogue Workshop on Discourse and Dialogue*, 2007.

S. Reiter and G. Rigoll. Multimodal meeting analysis by segmentation and classification of meeting events based on a higher level semantic approach. In *Proceedings of International Conference on Acoustics, Speech, and Signal Processing*, volume 2, pages 161–164, 2005.

N. C. Romano Jr and J. F. Nunamaker Jr. Meeting analysis: Findings from research and practice. In *Proceedings of the 34th Annual Hawaii International Conference on System Sciences*, 2001.

D. Ron, Y. Singer, and N. Tishby. The power of amnesia: Learning probabilistic automata with variable memory length. *Machine learning*, 25(2-3):117–149, 1996.

T. M. Scheidel. *Persuasive Speaking.* Scott, Foresman and Company, 1967.

J. S. Sim and K. Park. The consensus string problem for a metric is NP-complete. *Journal of Discrete Algorithms*, 1(1):111–117, 2003.

SRI International. CALO: Cognitive assistant that learns and organizes. `http://caloproject.sri.com`, 2003-2009.

B. Sternthal, L.W. Phillips, and R. Dholakia. The persuasive effect of scarce credibility: a situational analysis. *Public Opinion Quarterly*, 42(3):285–314, 1978.

A. Stolcke, K. Ries, N. Coccaro, E. Shriberg, R. Bates, D. Jurafsky, P. Taylor, R. Martin, C.V. Ess-Dykema, and M. Meteer. Dialogue act modeling for automatic tagging and recognition of conversational speech. *Computational Linguistics*, 26(3):339–373, 2000.

G. Tur, A. Stolcke, L. Voss, S. Peters, D. Hakkani-Tur, J. Dowding, B. Favre, R. Fernandez, M. Frampton, M. Frandsen, C. Frederickson, M. Graciarena, D. Kintzing, K. Leveque, S. Mason, J. Niekrasz, M. Purver, K. Riedhammer, E. Shriberg, Jing Tien, D. Vergyri, and Fan Yang. The CALO meeting assistant system. *IEEE Transactions on Audio, Speech, and Language Processing*, 18(6):1601–1611, 2010.

W. M. P. van der Aalst. The application of petri nets to workflow management. *Journal of Circuits, Systems, and Computers*, 8(01):21–66, 1998.

Appendix

Proof of Theorem 1.

We will use Hoeffding's inequality combined with the union bound to create a uniform generalization bound over all viable templates. The main step in doing this is to count the number of possible viable templates. Let us do this now.

Let Λ be the set of possible dialogue acts, and denote $|\Lambda|$ as the number of elements in the set. We will calculate the number of templates that are of size less than or equal to L, which is the size of our function class in statistical learning theory.

For a template of exactly length n, there are $|\Lambda|^n$ possible assignments of dialogue acts for the templates. Also, for a template of length n, there are at most $\binom{n(n-1)/2}{B}$ possible assignments of B backward arrows, where $B \leq n$. To see this, consider the set of backwards arrows as represented by an $n \times n$ adjacency matrix, where only the part below the diagonal could be 1. There are n^2 total elements in the matrix, n on the diagonal, so $n(n-1)$ off diagonal elements, and $n(n-1)/2$ elements in the lower triangle. If exactly b of these can be 1, the total number of possibilities is at most $\binom{n(n-1)/2}{b}$. There can be up to B backward arrows, so the total number of possibilities is at most

$$\sum_{b=0}^{\min(B,n)} \binom{n(n-1)/2}{b}.$$

Note that this number is an upper bound, as usually we cannot have more than one backwards arrow leaving or entering a node. Finally we could have n anywhere between 0 and L so the final number of possible templates has upper bound:

$$\sum_{n=0}^{L} |\Lambda|^n \sum_{b=0}^{\min(B,n)} \binom{n(n-1)/2}{b}.$$

Hoeffding's inequality applies to arbitrary bounded loss functions. This, combined with a union bound over all viable templates, yields the statement of the theorem.