TR 539: *Condensed version submitted to NIPS'01*

Learning Human Interactions with the Influence Model

Sumit Basu[*] **Tanzeem Choudhury**[*] **Brian Clarkson**[*]
Media Laboratory
Massachusetts Institute of Technology
Cambridge, MA 02139
{sbasu,tanzeem,clarkson}@media.mit.edu

Alex (Sandy) Pentland
Media Laboratory
Massachusetts Institute of Technology
Cambridge, MA 02139
sandy@media.mit.edu

Abstract

We are interested in quantitatively modeling the interactions between humans in conversational settings. While a variety of models are potentially appropriate, such as the coupled HMM, all require a very large number of parameters to describe the interactions between chains. We propose as an alternative the generative model developed in [1], the Influence Model, which parametrizes the hidden state transition probabilities by taking a convex combination of the pairwise transitions with constant "influence" parameters. We develop a learning algorithm for this model and show its abilities to model chain dependencies in comparison to other standard models using synthetic data. We also show early results of applying this model to human interaction data.

1 Introduction

There is a long history of work in the social sciences aimed at understanding the interactions between individuals and influencing their behavior. In the psychology community, there are many instances of work studying these effects. For instance, Wells and Petty [2] show how a speaker's confidence could be significantly influenced by repeated head nodding from the audience. Studies of this kind give us interesting insights into the workings of human dynamics. In many cases, the

[*] The first three authors contributed equally to this paper and are listed alphabetically.

experimenters have been able to take quantitative measures of behavior changes by looking at task performance or questionnaire responses. However, there are important aspects of the behavior which can not be captured this way – how much one person is influencing another's behavior, the intensity of the interaction, and so on. Thus far, experimenters have relied on qualitative measures based on hand annotations ("Bill talked more to Susan during this session" or "Joe was very angry and always yelled at Jane") or anecdotal incidents. This has a number of problems – it is difficult to know exactly what the experimenters meant by their measures, and even more difficult to compare the results across different studies. Our goal is to provide a means for quantifying these effects. Our backgrounds are in computer vision, speech processing, and machine learning, and we believe that with the relevant tools in hand we can make some progress towards this goal. This paper is our first exploration into modeling these complex interactions.

The general scenario of our work involves a number of participants interacting with each other and a number of actuators that can potentially affect their behavior. Initially, we wish only to observe and model the effects of people and actuators on each other, which will allow us to analyze and predict their behavior. In the long term, we wish to use these models to build feedback systems that apply actuators based on the results of analysis. Closing the loop in this way would eventually result in a stochastic model for influencing human behavior. This would ideally allow us to enhance the interaction between the individuals (e.g., prevent fights, increase understanding, facilitate discussions). While we are still far from this goal, we hope these first steps make some progress towards understanding human interactions. To this end, we are currently working with a particular experimental setup we call the "Facilitator Room," which we describe in Section 2.

While there are many possible *ad hoc* approaches to computing features for each of these effects (e.g., interaction intensity), we wish to develop a common framework in which we can analyze all of the effects of subjects influencing each other and experimental variables influencing the subjects. The "Influence Model," developed as a generative model by Chalee Asivathiratham in his PhD disseration [1], is a possible means of representing the influences a number of Markov chains have on each other. As we will describe in Section 3, this model seems very appropriate to model the effects we are interested in. In this paper, we generalize the model with hidden states/observations and develop an algorithm for learning its parameters from data (Section 3.2). We then show the performance of this algorithm and model with respect to other models using synthetic data. In Section 4, we show early results of applying our model to interaction scenarios in the Facilitator Room. We close with our preliminary conclusions and plans for future work.

2 The Facilitator Room

In the interests of studying the interactions between humans and the influences of various experimental variables, we have developed an experimental setup we call the "Facilitator Room." This room is a 15 foot by 15 foot space with three couches and a table. The room is instrumented with six pan-tilt-zoom cameras and an array of microphones. From these sensors, we estimate features such as speaking rate, speech pitch, relative speech energy, region-based motion energy, and blob tracking. When using all sensors, this amounts to a data rate of 1.9 Gigabytes/hour. In the experiments described in Section 4, we use a subset of the sensors resulting a data rate of 1.1G/hr for two hours.

The room is also outfitted with a number of actuators meant to influence the behavior of the participants. Currently we have speakers mounted behind each seat intended to mask sounds with white noise, whisper items to individuals, and so on. We have also installed lights focused on each seat whose colors and intensities are under computer control. These are meant to change overall room lighting conditions and also to spotlight individuals to affect others' response to them. There are five projectors in the room: three on the walls, one going to a main screen, and one on the table. These are intended to show relevant information at appropriate times in the hopes of changing the conversation pattern.

In the experiments in this paper, we have only begun to use the potential of this room – at this point we are not using the actuators. However, we cannot study the effects of actuators until we have modeled the baseline interactions among individuals, and thus in this study we focus on the latter.

3 The Influence Model

In seeking a model appropriate to our goals, we turned to the work on dynamic Bayes nets (DBN's) in the graphical models community (for a review of DBNs see [3]). The most straightforward approach would be to model each participant with an HMM and then take the outer product of all the state spaces. Unfortunately, the number of states would be exponential in the number of chains N where Q is the number of states per chain, i.e., Q^N, and the number of parameters in the transition matrix would be Q^{2N}. Furthermore, it would be difficult to interpret the parameters of the resulting model – it would not be easy to determine, for example, whether a given person had an effect on another.

The coupled HMM described by [4] and [5] for pairs of chains is a potential solution here, as it models the dependency of one chain's state on both chains' previous states, i.e., $P(S_t^i \mid S_{t-1}^i, S_{t-1}^j)$, as shown in Figure 1 (a). This keeps the state space at $2N$, with a QxQxQ transition table for each chain, for a total of $2Q^3$ transition parameters per chain.

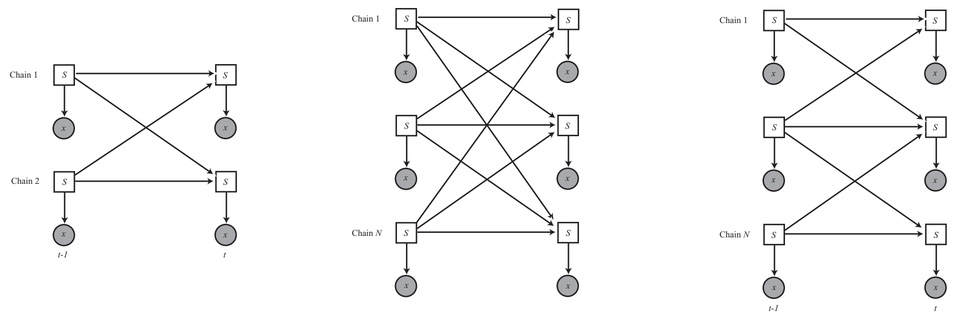

Figure 1: Possible DBN models for human interactions shown at times t-1 and t: (a) a coupled HMM, (b) a generalized version of the coupled HMM with N chains, and (c) pairwise chains of coupled HMMs.

The generalization of this model to N chains that makes sense here is for the next state to depend on the value of all of the previous states, i.e., we should estimate

$P(S_t^i \mid S_{t-1}^1, \ldots, S_{t-1}^N)$ (see Figure 1). However, this requires a Q^NxQ transition table for each chain, resulting in NQ^{N+1} transition matrix parameters, which is still very large. Another possibility is to model all possible pairs of chains with a standard coupled HMM. Such a scheme would still require $\binom{N}{2} Q^3$ parameters for the transition tables. Since we would like to estimate these parameters over fairly short segments such as a fraction of a conversation, we wish to keep the number of parameters to a minimum, and as a result continued to search for an appropriate model.

3.1 (Re)introducing the Influence Model

In his dissertation, Asavathiratham [1] introduced the "Influence Model," a generative model for describing the connections between many Markov chains with a simple parametrization in terms of the “influence” each chain has on the others. His work showed how complex phenomena involving interactions between large numbers of chains could be simulated through this simplified model, such as the up/down time for power stations across the US power grid. In his description, all states were observed, and he did not develop a mechanism for learning the parameters of the model. We thus nominally extend his model by adding the notion of hidden states and observations. We also develop a learning algorithm for the parameters in Section 3.2.

The graphical model for the influence model is identical to that of the generalized *N*-chain coupled HMM, but there is one very important simplification. Instead of keeping the entire $P(S_t^i \mid S_{t-1}^1, \ldots, S_{t-1}^N)$, we only keep $P(S_t^i \mid S_{t-1}^j)$ and approximate the former with:

$$P(S_t^i \mid S_{t-1}^1, \ldots, S_{t-1}^N) = \sum_j \alpha_{ij} P(S_t^i \mid S_{t-1}^j)$$

In other words, we form the distribution for a given chain’s next state by taking a convex combination of the pairwise conditional probabilities. As a result, we only have *N* QxQ tables and *N* α parameters per chain, resulting in a total of $NQ^2 + N^2$ transition parameters. This is far fewer parameters than any of the above models. The real question, of course, is whether we have retained enough modeling power to determine the interactions between the participants.

Asavathiratham refers to the α's as "influences," because they are constant factors that tell us how much the state transitions of a given chain depend on a given neighbor. It is important to realize the ramifications of these factors being constant: intuitively, it means that *how much* we are influenced by a neighbor is constant, but *how* we are influenced by it depends on its state. Another way to look at this is that we are only modeling the first-order effects of our neighbors' influences on us: if Joe yelling causes us to be quiet with certainty and Mark's yelling causes us to yell back with certainty and our α’s for both are equal, the combination of both yelling will result in a distribution of our next action that has its probability mass equally distributed over yelling and not yelling. This is what we are giving up in terms of modeling power – while the fully-connected coupled HMM would allow us to explicitly model the effect of the joint event of Joe and Mark yelling together, the influence model does not (note, however, that the set of pairwise coupled HMMs would also not be able to model this joint effect).

This simplification seems reasonable for the domain of human interactions and potentially for many other domains. Furthermore, it gives us a small set of interpretable parameters – the α values – which summarize the interactions between the chains. By estimating these parameters, we can gain an understanding of how much the chains influence each other.

In the sections below, we develop a learning algorithm for this model and show with both synthetic and real data the kinds of interaction structure this model can capture. Last, though we do not explore it in this paper, we can easily use this model to estimate the effects of actuators on the participants. We would simply create a new (observed) chain for each actuator and then observe its influences. In this case, to investigate *how* the actuator changed the participants' behavior, we would have to look at the probability tables as well as the α 's.

3.2 Learning for the Influence Model

The problem of estimating the Influence Model from data can be stated as follows. We are given sequences of observations, $\{x_t^i\}$, from each chain i. The goal is to estimate the amount of influence, α_{ij}, that chain *j* has on chain *i*, along with the pairwise conditional probability distributions that describe this inter-chain influence, $P(S_t^i \mid S_{t-1}^j)$. In this section we develop methods for doing this and illustrate them with synthetic data.

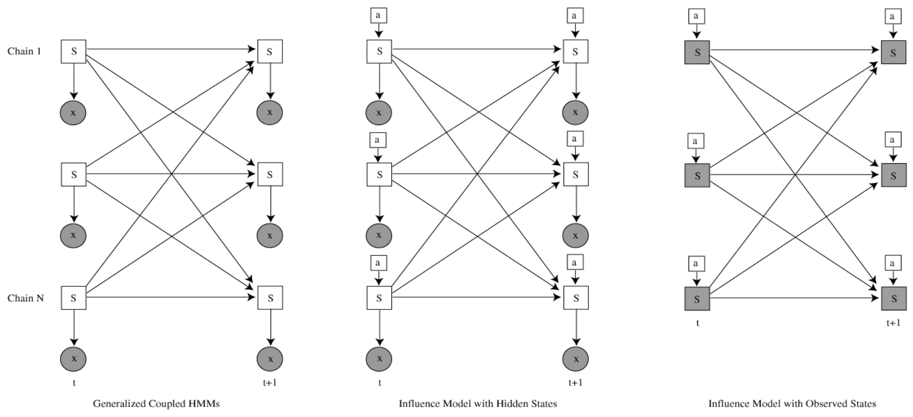
Chain 1
Chain N
S
x
a
t
t+1
Generalized Coupled HMMs
Influence Model with Hidden States
Influence Model with Observed States

Figure 2: Graphs for (a) a generalized coupled HMM, (b) an Influence Model with hidden states, (c) an Influence Model with observed states.

3.2.1 Expectation-Maximization for the Influence Model

In Figure 2 we show the graphical model for the most general form of the Influence Model with hidden states and continuous observations. Fitting this model to data requires us to maximize the likelihood of Influence Model over its free parameters. The likelihood function can be readily written as:

$$P(S,X) = \left(\prod_i P(S_0^i) P(x_0^i \mid S_0^i) \right) \prod_i \prod_t P(x_t^i \mid S_t^i) \sum_j \alpha_{ij} P(S_t^i \mid S_{t-1}^j)$$

One possibility for estimating the parameters of this model is Expectation-Maximization. The E-step requires us to calculate $P(S \mid X, \theta)$, which in most cases amounts to applying the Junction Tree algorithm (for exact inference) or some approximate inference scheme (variational, etc.). We will discuss the possibilities for doing inference on this model later. The M-step is specific to this model and requires maximizing the lower bound obtained in the E-step. Examining this expression we can see that the M-step for all the parameters except the α_{ij}'s is only trivially different from the HMM. However, we can readily write down the update equations for the α_{ij}'s by noticing that they are mixture weights for *N* conditional probability tables analogous to a mixture of Gaussians. The α_{ij} update equations are obtained by following the derivation of the M-step for a Gaussian mixture (i.e., by introducing a hidden state to represent the "active" mixture component and then taking an expectation over its sufficient statistics):

$$\alpha_{ij}^{new} = \frac{\sum_t \sum_k \sum_l P(c_t^i = j, S_t^i = k, S_{t-1}^j = l \mid X)}{\sum_t \sum_k \sum_l P(S_t^i = k, S_{t-1}^j = l \mid X)}$$

The "$c_t^i = j$" event means that at time t chain i was influenced by chain j, and the "$S_t^i = k$" event means that chain i was in state k during time t.

3.2.2 The Observed Influence Model

Due to the difficulties involved in doing the inference required for E-step, we decided to simplify the estimation problem by allowing the states S_t^i to be observed for each chain (see Figure 2). We decided to obtain our state sequences by fitting an HMM to each chain's observations and performing a Viterbi decoding. The chain transition tables were then easily estimated (by frequency counts) directly from these state sequences. Since our goal is to estimate the inter-chain influences (via the α_{ij}'s) this "clamping" of the observation and chain transition parameters helps combat the overfitting problems of the full model.

We now have an unusual DBN where the observed nodes are strongly interconnected and the hidden states are not. This presents serious problems for inference because marginalizing out the observed state nodes causes all the hidden states to become fully connected across all time and all chains. Unless we apply an approximation that can successfully decouple these nodes, a maximization procedure such as EM will not be tractable. However, there is a far simpler way to estimate the α_{ij} values in our observed scenario. Let us first examine how the likelihood function simplifies for the observed Influence Model:

$$P(S \mid \{\alpha_{ij}\}) = \left(\prod_i P(S_0^i)\right) \prod_i \prod_t \sum_j \alpha_{ij} P(S_t^i \mid S_{t-1}^j)$$

Converting this expression to log likelihood and removing terms that are not relevant to a maximization over α_{ij} yields:

$$\alpha_{ij}^* = \arg\max_{\alpha_{ij}} \left[\sum_i \sum_t \log \sum_j \alpha_{ij} P(S_t^i \mid S_{t-1}^j) \right]$$

We can further simplify this expression by keeping terms relevant to chain i:

$$\alpha_{ij}^* = \arg\max_{\alpha_{ij}} \left[\sum_t \log \sum_j \alpha_{ij} P(S_t^i \mid S_{t-1}^j) \right]$$

This *per chain* likelihood is concave in α_{ij}, which can be easily shown as follows:

Let $\alpha = \begin{bmatrix} \alpha_{i0} \\ \vdots \\ \alpha_{iN} \end{bmatrix}$ $B_t^i = \begin{bmatrix} P(S_t^i \mid S_{t-1}^0) \\ \vdots \\ P(S_t^i \mid S_{t-1}^N) \end{bmatrix}$.

Then the *per chain* likelihood becomes: $f_i(\alpha) = \sum_t \log \left\langle \alpha, B_t^i \right\rangle$. This is concave since for any $0 < w \le 1$ and α_0, α_1:

$$\begin{aligned} f((1-w)\alpha_0 + w\alpha_1) &= \sum_t \log \left\langle (1-w)\alpha_0 + w\alpha_1, B_t^i \right\rangle \\ &= \sum_t \log \left[(1-w)\left\langle \alpha_0, B_t^i \right\rangle + w\left\langle \alpha_1, B_t^i \right\rangle \right] \quad \text{(using Jensen)} \\ &\ge \sum_t (1-w)\log\left\langle \alpha_0, B_t^i \right\rangle + w\log\left\langle \alpha_1, B_t^i \right\rangle \\ &= (1-w)f(\alpha_0) + wf(\alpha_1) \end{aligned}$$

Now taking the derivative w.r.t. α_{ij}:

$$\frac{\partial}{\partial \alpha_{ij}}(.) = \sum_t \frac{P(S_t^i \mid S_{t-1}^j)}{\sum_k \alpha_{ik} P(S_t^i \mid S_{t-1}^k)} = \sum_t \frac{P(S_t^i \mid S_{t-1}^j)}{\left\langle \alpha_i, B_t^i \right\rangle}$$

Notice that the gradient and the *per chain* likelihood expression above are inexpensive to compute with appropriate rearranging of the conditional probability tables to form the B_t^i vectors. This along with the facts that the *per chain* likelihood is concave and the space of feasible α_{ij}'s is convex means that this optimization problem is a textbook case for constrained gradient ascent with full 1-D search (see p.29 of [6]). Furthermore, in all examples in this paper, 20 iterations were sufficient to ensure convergence, which amounted to less than 10 seconds of CPU time.

3.2.3 Evaluation of the Observed Influence Model on Synthetic Data

To evaluate the effectiveness of our learning algorithm we first show results on synthetic data. The data was generated by an Influence Model with 3 chains in lock step: one leader which was evolving randomly (i.e., flat transition tables) and 2 followers who meticulously followed the leader (i.e., an influence of 1 by chain 2 and a self-influence of 0). We sampled this model to obtain a training sequence of 50 timesteps for each chain. These state sequences were then used to train another randomly initialized Influence Model. As described above, the $P(S_t^i \mid S_{t-1}^j)$ were estimated by counting and the α_{ij}'s by gradient ascent. The resulting influence graph is shown along with a typical sample sequence in Figure 3. Note how the "following" behavior is learned exactly by this model – chains 1 and 3 follow chain 2 perfectly.

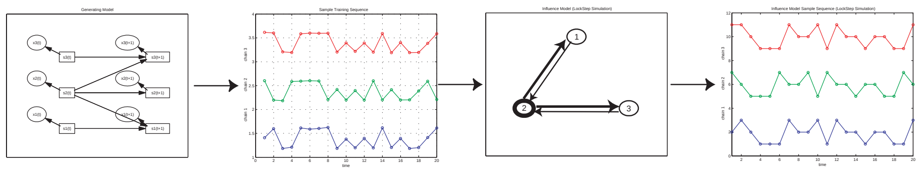

Figure 3: The evaluation pipeline for testing the Influence Model on the lockstep synthetic data: (a) the graph for the generating model at time t and t+1 (b) the training sequence (c) the learned influences (α's) – the thickness of the lines corresponds to the magnitude of the influence. Note that the strong influence of chain 2 on 1 and 3 was correctly learned. (d) Sample paths from the learned model. Note how chains 1 and 3 (the followers) follow chain 2 perfectly.

We also evaluated the Generalized Coupled HMM (i.e. full state transition tables instead of the mixtures of pairwise tables) on this data using EM, using the Junction Tree Algorithm for inference. Again we sampled from the lock step model and trained a randomly initialized model. A sample sequence from the resulting model is shown in Figure 4. In this case, the learned model performed reasonably well, but was unable to learn the "following" behavior perfectly due to the larger number of parameters it had to estimate ($P(S_t^i \mid S_{t-1}^1, \ldots, S_{t-1}^N)$ vs. $P(S_t^i \mid S_{t-1}^j)$).

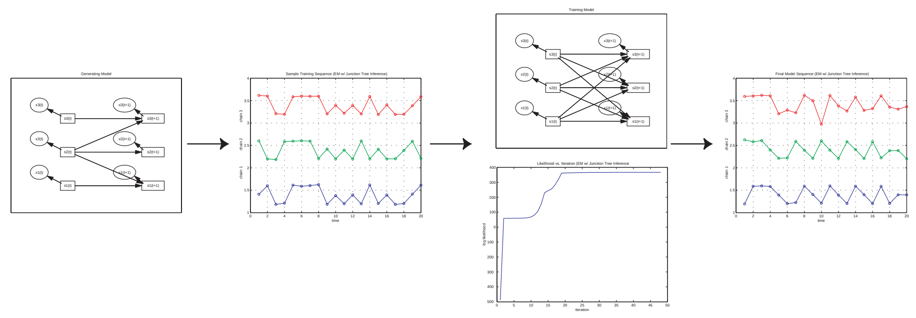

Figure 4: The evaluation pipeline for testing the Generalized Coupled HMM on the lockstep synthetic data: (a) the graph for the generating model (b) the training sequence (c) the graph for the learned model and the likelihood values for the EM iterations (d) a sample path from the learned model. Note that this model was unable to capture the lockstep behavior perfectly, as can be seen in the errors of chain 1 in following chain 2.

4 Experiments and Results

After verifying the performance of our algorithm on synthetic data, we tested our models on data of natural human interactions in the facilitator room. We recorded two hours of data of five participants playing an interactive debating game. The game, Opinions, comes with stack of cards that has different controversial debate topics. We recorded ten games (debate sessions) for our experiment. In order to ensure that we saw a debate session between all possible pairs of players, we listed all pairs and chose pairs from the list without replacement. The first participant in the list entry rolled a die to pick a side (proponent or opponent). Each debater spoke for one minute after which the stage was open for discussion between all participants. No restrictions were imposed on the participants' interaction style during the game. The features calculated automatically from the data were per person motion energy (30 Hz), speech energy (30 Hz), and voicing state (60 Hz). Also, the speaker turns (i.e., who was speaking when) were hand labeled for all the games.

In the first experiment, we used the hand-labeled speaker turns only. Each player had two states – speaking and silent. When multiple players were speaking at the same time, all of them were considered to be in the speaking state. The full set of features for the game was the binary state vector for all of the players, which was afterwards non-uniformly resampled in order to remove consecutively repeating states. Therefore, if all the players were in the same state for t timesteps, those t identical observations were effectively replaced with one time step. This effectively broke up the data such that there would be one feature vector per conversational turn. If the features were not resampled in this way, the self-transitions would overwhelm the effects of any inter-person influences.

We estimated the influence matrix α for the entire dataset (all ten games) and also for each game separately. In this dataset, player Tammy was asked always to respond/react to player Bob and thus we expect Bob to have a strong influence on Tammy. We see this influence in the full game influence matrix shown in Figure 5. Furthermore, Tammy and Anne were observed to be the dominating speakers in all of the datasets. This appears in the learned graphs as the strong connections to the other participants. Last, for each game we compared the learned influence graph with the influence graph generated from a hand-labeled interaction matrix. This latter graph was formed by creating a directed link from participant A to participant B if participant B responded to participant A; the strength of the link is proportional to the number of times B responded to A (normalized by all of A's influencers). Figure 6 shows the results for a subset of games from our dataset.

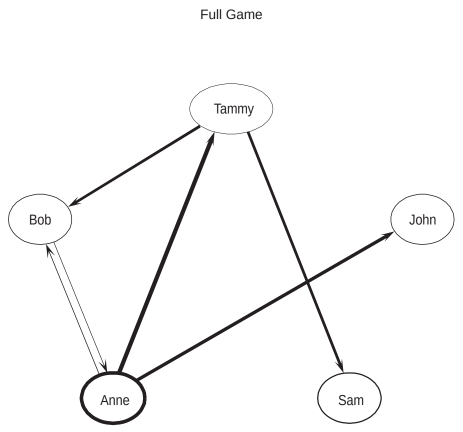

Figure 5: Influence graph for the full game showing strong links for the dominating speakers Tammy and Anne.

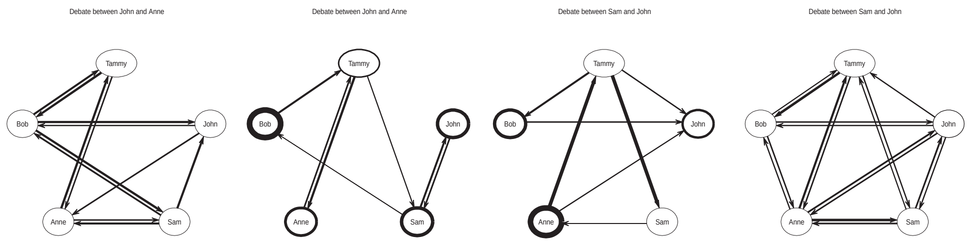
Debate between John and Anne
Debate between John and Anne
Debate between Sam and John
Debate between Sam and John
Tammy
Bob
John
Anne
Sam

Figure 6: Results for two sub-games (a) debate between John and Anne - interaction graph learned by our model (b) graph of hand-coded influence matrix used as ground truth (c) debate between Sam and John – learned model (d) ground truth.

Finally, we ran our algorithms using the motion energy, speech energy and voicing state for each person to generate the influence graphs, now learning the states in an unsupervised manner. Figure 7 shows one example of the influence structure extracted, which shows a strong link between the proponent and the opponent, as was observed in the audio-visual record for that game.

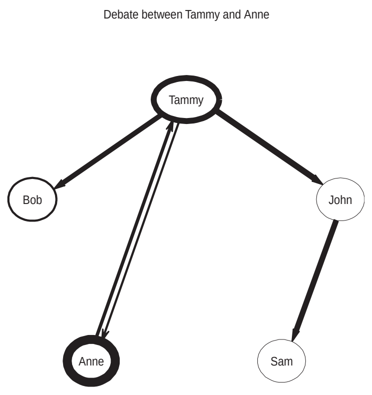

Figure 7: Influence graph for one debate session learned using the automatically generated features - motion energy, speech energy and voicing state. The model learned the strong link between the two debaters.

5 Discussion and Future Work

Human interactions are quite complex and we cannot hope to capture all of their subtleties with a simple graphical model. However, we have shown that the Influence Model is capable of describing some of the phenomena we expect to see, and also that the learning algorithms we have proposed are capable of reliably estimating these parameters. Furthermore, we have applied our algorithms to real data of human interactions and shown that we can recover some of the structure observed in the data. While these first forays do not yet constitute a comprehensive framework for analyzing human interactions, we believe they are an important step towards characterizing the influences people have on each other during conversations.

Currently, we are running further experiments using the automatically generated features (as in Figure 7) to see how well they correlate with our measures of ground truth. We are also considering a number of different methods for doing full learning on the model, including variational methods and other approximate inference techniques. We also plan to perform more detailed evaluations of the performance of the Influence Model with respect to the generalized (N-chain) Coupled HMM.

Acknowledgments

We used Kevin Murphy's MATLAB Bayes Net Toolbox for performing EM with the junction tree algorithm.

References

1. Asavathiratham, C., *The Influence Model: A Tractable Representation for the Dynamics of Networked Markov Chains*. In Dept. of EECS, MIT. Cambridge, 2000.
2. Wells, G., Petty, R., "The Effects of Overt Head Movements on Persuasion." *Basic and Applied Social Psychology*, 1980. **1**(3): pp. 219-230.
3. Ghahramani Z, "Learning Dynamic Bayesian Networks." *Adaptive Processing of Sequences and Data Structures. International Summer School on Neural Networks 'E.R. Caianiello'. Tutorial Lectures. Springer Verlag*, 1998.

4. Brand, M., N. Oliver, and A. Pentland. "Coupled hidden Markov models for complex action recognition." In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*. 1997.

5. Saul, L. and M. Jordan, "Boltzmann Chains and Hidden Markov Models." *Advances in Neural Information Processing Systems*, 1995. **7**: pp. 435-42.

6. Bertsekas, D.P., *Nonlinear Programming*. Athena Scientific: Belmont, 1995.