Discovering Latent Structure in Task-Oriented Dialogues

Ke Zhai[*]
Computer Science, University of Maryland
College Park, MD 20740
zhaike@cs.umd.edu

Jason D. Williams
Microsoft Research
Redmond, WA 98052
jason.williams@microsoft.com

Abstract

A key challenge for computational conversation models is to discover latent structure in task-oriented dialogue, since it provides a basis for analysing, evaluating, and building conversational systems. We propose three new unsupervised models to discover latent structures in task-oriented dialogues. Our methods synthesize hidden Markov models (for underlying state) and topic models (to connect words to states). We apply them to two real, non-trivial datasets: human-computer spoken dialogues in bus query service, and human-human text-based chats from a live technical support service. We show that our models extract meaningful state representations and dialogue structures consistent with human annotations. Quantitatively, we show our models achieve superior performance on held-out log likelihood evaluation and an ordering task.

1 Introduction

Modeling human conversation is a fundamental scientific pursuit. In addition to yielding basic insights into human communication, computational models of conversation underpin a host of real-world applications, including interactive dialogue systems (Young, 2006), dialogue summarization (Murray et al., 2005; Daumé III and Marcu, 2006; Liu et al., 2010), and even medical applications such as diagnosis of psychological conditions (DeVault et al., 2013).

Computational models of conversation can be broadly divided into two genres: *modeling* and *control*. *Control* is concerned with choosing actions in interactive settings—for example to maximize task completion—using reinforcement learning (Levin et al., 2000), supervised learning (Hurtado et al., 2010), hand-crafted rules (Larsson and Traum, 2000), or mixtures of these (Henderson and Lemon, 2008). By contrast, *modeling*—the genre of this paper—is concerned with inferring a phenomena in an existing corpus, such as dialogue acts in two-party conversations (Stolcke et al., 2000) or topic shifts in multi-party dialogues (Galley et al., 2003; Purver et al., 2006; Hsueh et al., 2006; Banerjee and Rudnicky, 2006).

Many past works rely on supervised learning or human annotations, which usually requires manual labels and annotation guidelines (Jurafsky et al., 1997). It constrains scaling the size of training examples, and application domains. By contrast, unsupervised methods operate only on the observable signal (e.g. words) and are estimated without labels or their attendant limitations (Crook et al., 2009). They are particularly relevant because conversation is a *temporal process* where models are trained to infer a latent *state* which evolves as the dialogue progresses (Bangalore et al., 2006; Traum and Larsson, 2003).

Our basic approach is to assume that each utterance in the conversation is in a latent *state*, which has a causal effect on the words the conversants produce. Inferring this model yields basic insights into the structure of conversation and also has broad practical benefits, for example, speech recognition (Williams and Balakrishnan, 2009), natural language generation (Rieser and Lemon, 2010), and new features for dialogue policy optimization (Singh et al., 2002; Young, 2006).

There has been limited past work on unsupervised methods for conversation modeling. Chotimongkol (2008) studies task-oriented conversation and proposed a model based on a *hidden Markov model* (HMM). Ritter et al. (2010) extends it by introducing additional word sources, and applies to *non-task-oriented* conversations—social interactions on Twitter, where the subjects

[*]Work done at Microsoft Research.

discussed are very diffuse. The additional word sources capture the subjects, leaving the state-specific models to express common dialogue flows such as question/answer pairs.

In this paper, we retain the underlying HMM, but assume words are emitted using *topic models* (TM), exemplified by *latent Dirichlet allocation* (Blei et al., 2003, LDA). LDA assumes each word in an utterance is drawn from one of a set of latent topics, where each topic is a multinomial distribution over the vocabulary. The key idea is that the set of topics is *shared* across all states, and each state corresponds to a mixture of topics. We propose three model variants that link topics and states in different ways.

Sharing topics across states is an attractive property in task-oriented dialogue, where a single concept can be discussed at many points in a dialogue, yet different topics often appear in predictable sequences. Compared to past works, the *decoupling* of states and topics gives our models more expressive power and the potential to be more data efficient. Empirically, we find that our models outperform past approaches on two real-world corpora of task-oriented dialogues.

This paper is organized as follows: Section 2 introduces two task-oriented domains and corpora; Section 3 details three new unsupervised generative models which combine HMMs and LDA and efficient inference schemes; Section 4 evaluates our models qualitatively and quantitatively, and finally conclude in Section 5.

2 Data

To test the generality of our models, we study two very different datasets: a set of human-computer spoken dialogues in quering bus timetable (*BusTime*), and a set of human-human text-based dialogues in the technical support domain (*TechSupport*). In *BusTime*, the conversational structure is known because the computer followed a deterministic program (Williams, 2012), making it possible to directly compare an inferred model to ground truth on this corpus.[1] In *TechSupport*, there is no known flowchart,[2] making this a realistic application of unsupervised methods.

[1] Available for download at `http://research.microsoft.com/en-us/events/dstc/`

[2] Technical support human agents use many types of documentation—mainly checklists and guidelines, but in general, there are no flowcharts.

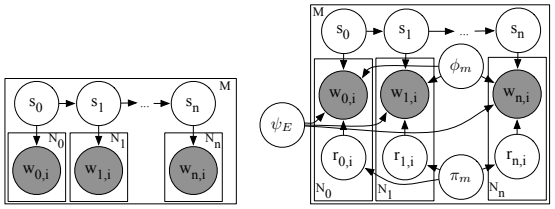
BusTime This corpus consists of logs of telephone calls between a spoken dialogue system and real bus users in Pittsburgh, USA (Black et al., 2010). For the user side, the words logged are the words recognized by the automatic speech recognizer. The vocabulary of the recognizer was constrained to the bus timetable task, so only words known to the recognizer in advance are output. Even so, the word error rate is approximately 30-40%, due to the challenging audio conditions of usage—with traffic noise and extraneous speech. The system asked users sequentially for a bus route, origin and destination, and optionally date and time. The system confirmed low-confidence speech recognition results. Due to the speech recognition channel, system and user turns always alternate. An example dialogue is given below:

System: Say a route like ⟨bus-route⟩, or say I'm not sure.
User: ⟨bus-route⟩.
System: I thought you said ⟨bus-route⟩, is that right?
User: Yes.
System: Say where're you leaving from, like ⟨location⟩.
User: ⟨location⟩.
System: Okay, ⟨location⟩, where are you going to?
...

We discard dialogues with fewer than 20 utterances. We also map all named entities (e.g., "downtown" and "28X") to their semantic types (resp. ⟨location⟩ and ⟨bus-route⟩) to reduce vocabulary size. The corpus we use consists of approximately 850 dialogue sessions or 30,000 utterances. It contains 370,000 tokens (words or semantic types) with vocabulary size 250.

TechSupport This corpus consists of logs of real web-based human-human text "chat" conversations between clients and technical support agents at a large corporation. Usually, clients and agents first exchange names and contact information; after that, dialogues are quite free-form, as agents ask questions and suggest fixes. Most dialogues ultimately end when the client's issue has been resolved; some clients are provided with a reference number for future follow-up. An example dialogue is given below:

Agent: Welcome to the answer desk! My name is ⟨agent-name⟩. How can I help you today?
Agent: May I have your name, email and phone no.?
Client: Hi, ⟨agent-name⟩. I recently installed new software but I kept getting error, can you help me?
Agent: Sorry to hear that. Let me help you with that.
Agent: May I have your name, email and phone no.?
Client: The error code is ⟨error-code⟩.
Client: It appears every time when I launch it.
Client: Sure. My name is ⟨client-name⟩.
Client: My email and phone are ⟨email⟩, ⟨phone⟩.
Agent: Thanks, ⟨client-name⟩, please give me a minute.



(a) *LM-HMM* (b) *LM-HMMS*

Figure 1: Plate diagrams of baseline models, from existing work (Chotimongkol, 2008; Ritter et al., 2010). Variable definitions are given in the text.

...

This data is less structured than *BusTime*; clients' issues span software, hardware, networking, and other topics. In addition, clients use common internet short-hand (e.g., "thx", "gtg", "ppl", "hv", etc), with mis-spellings (e.g., "ofice", "offfice", "erorr", etc). In addition, chats from the web interface are segmented into turns when a user hits "Enter" on a keyboard. Therefore, clients' input and agents' responses do **not** necessarily alternate consecutively, e.g., an agent's response may take multiple turns as in the above example. Also, it is unreasonable to group consecutive chats from the same party to form a "alternating" structure like *BusTime* dataset due to the asynchronism of different states. For instance, the second block of client inputs clearly comes from two different states which should *not* be merged together.

We discard dialogues with fewer than 30 utterances. We map named entities to their semantic types, apply stemming, and remove stop words.[3] The corpus we use contains approximately $2,000$ dialogue sessions or $80,000$ conversation utterances. It consists of $770,000$ tokens, with a a vocabulary size of $6,600$.

3 Latent Structure in Dialogues

In this work, our goal is to infer latent structure presented in task-oriented conversation. We assume that the structure can be encoded in a probabilistic state transition diagram, where the dialogue is in one state at each utterance, and states have a causal effect on the words observed. We assume the boundaries between utterances are given, which is trivial in many corpora.

The simplest formulation we consider is an HMM where each state contains a unigram *language model* (LM), proposed by Chotimongkol (2008) for task-oriented dialogue and originally developed for discourse analysis by Barzilay and Lee (2004). We call it *LM-HMM* as in Figure 1(a). For a corpus of M dialogues, the m-th dialogue contains n utterances, each of which contains N_n words (we omit index m from terms because it will be clear from context). At n-th utterance, we assume the dialogue is in some latent state s_n. Words in n-th utterance $w_{n,1}, \ldots, w_{n,N_n}$ are generated (independently) according to the *LM*. When an utterance is complete, the next state is drawn according to *HMM*, i.e., $P(s'|s)$.

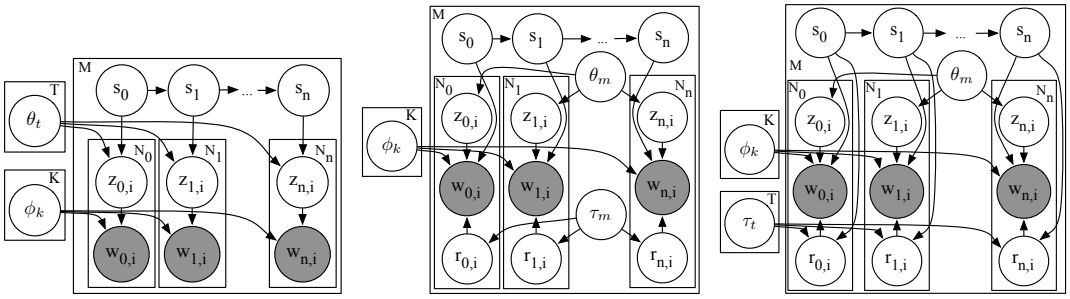
While *LM-HMM* captures the basic intuition of conversation structure, it assumes words are conditioned only on state. Ritter et al. (2010) extends *LM-HMM* to allow words to be emitted from two *additional* sources: the topic of current dialogue $\boldsymbol{\phi}$, or a background *LM* $\boldsymbol{\psi}$ shared across all dialogues. A multinomial $\boldsymbol{\pi}$ indicates the expected fraction of words from these three sources. For every word in an utterance, first draw a source indicator r from $\boldsymbol{\pi}$, and then generate the word from the corresponding source. We call it *LM-HMMS* (Figure 1(b)). Ritter et al. (2010) finds these alternate sources are important in non-task-oriented domains, where events are diffuse and fleeting. For example, Twitter exchanges often focus on a particular event (labeled X), and follow patterns like "saw X last night?", "X was amazing". Here X appears throughout the dialogue but does not help to distinguish conversational states in social media. We also explore similar variants.

In this paper, these two models form our baselines. For all models, we use *Markov chain Monte Carlo* (MCMC) inference (Neal, 2000) to find latent variables that best fit observed data. We also assume symmetric Dirichlet priors on all multinomial distributions and apply collapsed Gibbs sampling. In the rest of this section, we present our models and their inference algorithms in turn.

3.1 TM-HMM

Our approach is to modify the emission probabilities of states to be *distributions over topics* rather than distributions over words. In other words, instead of generating words via a *LM*, we generate words from a *topic model* (TM), where each state maps to a mixture of topics. The key benefit of this additional layer of abstraction is to enable states to express higher-level concepts through pooling of topics across states. For example, topics might be inferred for content like "bus-route" or "lo-

[3] We used regular expression to map named entities, and Porter stemmer in NLTK to stem all tokens.



(a) *TM-HMM* (b) *TM-HMMS* (c) *TM-HMMSS*

Figure 2: Plate diagrams of proposed models. *TM-HMM* is an HMM with state-wise topic distributions. *TM-HMMS* adds session-wise topic distribution and a source generator. *TM-HMMSS* adds a state-wise source generator. Variable definitions are given in the text.

cations"; and other topics for dialogue acts, like to "ask" or "confirm" information. States could then be *combinations* of these, e.g., a state might express "ask bus route" or "confirm location". This approach also decouples the number of topics from the number of states. Throughout this paper, we denote the number of topics as K and the number of states as T. We index words, turns and dialogues in the same ways as baseline models.

We develop three generative models. In the first variant (*TM-HMM*, Figure 2(a)), we assume every state s in HMM is associated with a distribution over topics $\boldsymbol{\theta}$, and topics generate words $\boldsymbol{w}$ at each utterance. The other two models allow words to be generated from different sources (in addition to states), akin to the *LM-HMMS* model.

TM-HMM generates a dialogue as following:

1: For each utterance n in that dialogue, sample a state s_n based on the previous state s_{n-1}.
2: For each word in utterance n, first draw a topic z from the state-specified distribution over topics $\boldsymbol{\theta}_{s_n}$ conditioned on s_n, then generate word w from the topic-specified distribution over vocabulary $\boldsymbol{\phi}_z$ based on z.

We assume $\boldsymbol{\theta}$'s and $\boldsymbol{\phi}$'s are drawn from corresponding Dirichlet priors, as in *LDA*.

The posterior distributions of state assignment s_n and topic assignment $z_{n,i}$ are

$$p(s_n|\boldsymbol{s}_{-n}, \boldsymbol{z}, \boldsymbol{\alpha}, \boldsymbol{\gamma}) \propto p(s_n|\boldsymbol{s}_{-n}, \boldsymbol{\gamma}) \cdot p(\boldsymbol{z}_n|\boldsymbol{s}, \boldsymbol{z}_{-n}, \boldsymbol{\alpha}), \quad (1)$$

$$p(z_{n,i}|\boldsymbol{s}, \boldsymbol{w}, \boldsymbol{z}_{-(n,i)}, \boldsymbol{\alpha}, \boldsymbol{\beta}) \propto p(z_{n,i}|\boldsymbol{s}, \boldsymbol{z}_{-(n,i)}, \boldsymbol{\alpha}) \cdot p(w_{n,i}|s_n, \boldsymbol{w}_{-(n,i)}, \boldsymbol{z}, \boldsymbol{\beta}),$$

where $\boldsymbol{\alpha}$, $\boldsymbol{\beta}$, $\boldsymbol{\gamma}$ are symmetric Dirichlet priors on state-wise topic distribution $\boldsymbol{\theta}_t$'s, topic-wise word distribution $\boldsymbol{\phi}_t$'s and state transition multinomials, respectively. All probabilities can be computed using collapsed Gibbs sampler for *LDA* (Griffiths and Steyvers, 2004) and *HMM* (Goldwater and Griffiths, 2007). We iteratively sample all parameters until convergence.

3.2 TM-HMMS

TM-HMMS (Figure 2(b)) extends *TM-HMM* to allow words to be generated either from state *LM* (as in *LM-HMM*), or a set of dialogue topics (akin to *LM-HMMS*). Because task-oriented dialogues usually focus on a specific domain, a set of words appears repeatedly throughout a given dialogue. Therefore, the topic distribution is often stable throughout the entire dialogue, and does not vary from turn to turn. For example, in the troubleshooting domain, dialogues about network connections, desktop productivity, and anti-virus software could each map to different session-wide topics. To express this, words in the *TM-HMMS* model are generated either from a dialogue-specific topic distribution, or from a state-specific language model.[4] A distribution over sources is sampled once at the beginning of each dialogue and selects the expected fraction of words generated from different sources.

The generative story for a dialogue session is:

1: At the beginning of each session, draw a distribution over topics $\boldsymbol{\theta}$ and a distribution over word sources $\boldsymbol{\tau}$.
2: For each utterance n in the conversation, draw a state s_n based on previous state s_{n-1}.
3: For each word in utterance n, first choose a word source r according to $\boldsymbol{\tau}$, and then depending on r, generate a word w either from the session-wide topic distribution $\boldsymbol{\theta}$ or the language model specified by the state s_n.

[4]Note that a *TM-HMMS* model with state-specific topic models (instead of state-specific language models) would be subsumed by *TM-HMM*, since one topic could be used as the background topic in *TM-HMMS*.

Again, we impose Dirichlet priors on distributions over topics $\boldsymbol{\theta}$'s and distributions over words $\boldsymbol{\phi}$'s as in *LDA*. We also assume the distributions over sources $\boldsymbol{\tau}$'s are governed by a Beta distribution.

The session-wide topics is slightly different from that used in *LM-HMMS*: *LM-HMMS* was developed for social chats on Twitter where topics are very diffuse and unlikely to repeat; hence often unique to each dialogue. By contrast, our models are designed for task-oriented dialogues which pertain to a given domain where topics are more tightly clustered; thus, in *TM-HMMS* session-wide topics are shared across the corpus.

The posterior distributions of state assignment s_n, word source $r_{n,i}$ and topic assignment $z_{n,i}$ are

$$\begin{aligned}
p(s_n|\boldsymbol{r}, \boldsymbol{s}_{-n}, \boldsymbol{w}, \boldsymbol{\gamma}, \boldsymbol{\pi}) &\propto p(s_n|\boldsymbol{s}_{-n}, \boldsymbol{\gamma}) \\
&\quad \cdot p(\boldsymbol{w}_n|\boldsymbol{r}, \boldsymbol{s}, \boldsymbol{\pi}), \\
p(r_{n,i}|\boldsymbol{r}_{-(n,i)}, \boldsymbol{s}, \boldsymbol{w}, \boldsymbol{\pi}) &\propto p(r_{n,i}|\boldsymbol{r}_{-(n,i)}, \boldsymbol{\pi}) \\
&\quad \cdot p(w_{n,i}|\boldsymbol{r}, \boldsymbol{s}, \boldsymbol{w}_{-(n,i)}, \boldsymbol{z}, \boldsymbol{\beta}), \qquad (2) \\
p(z_{n,i}|\boldsymbol{r}, \boldsymbol{w}, \boldsymbol{z}_{-(n,i)}, \boldsymbol{\alpha}, \boldsymbol{\beta}) &\propto p(z_{n,i}|\boldsymbol{r}, \boldsymbol{z}_{-(n,i)}, \boldsymbol{\alpha}) \\
&\quad \cdot p(w_{n,i}|\boldsymbol{r}, \boldsymbol{w}_{-(n,i)}, \boldsymbol{z}, \boldsymbol{\beta}),
\end{aligned}$$

where $\boldsymbol{\pi}$ is a symmetric Dirichlet prior on session-wise word source distribution $\boldsymbol{\tau}_m$'s, and other symbols are defined above. All these probabilities are Dirichlet-multinomial distributions and therefore can be computed efficiently.

3.3 TM-HMMSS

The *TM-HMMSS* (Figure 2(c)) model modifies *TM-HMMS* to re-sample the distribution over word sources $\boldsymbol{\tau}$ *at every utterance*, instead of once at the beginning of each session. This modification allows the fraction of words drawn from the session-wide topics to vary over the course of the dialogue. This is attractive in task-oriented dialogue, where some sections of the dialogue always follow a similar script, regardless of session topic—for example, the opening, closing, or asking the user if they will take a survey. To support these patterns, *TM-HMMSS* conditions the source generator distribution on the current state.

The generative story of *TM-HMMSS* is very similar to *TM-HMMS*, except the distribution over word sources $\boldsymbol{\tau}$'s are sampled at every state. A dialogue is generated as following:

1: For each session, draw a topic distribution $\boldsymbol{\theta}$.
2: For each utterance n in the conversation, draw a state s_n based on previous state s_{n-1}, and subsequently retrieve the state-specific distribution over word sources $\boldsymbol{\tau}_{s_n}$.
3: For each word in utterance n, first sample a word source r according to $\boldsymbol{\tau}_{s_n}$, and then depending on r, generate a word w either from the session-wide topic distribution $\boldsymbol{\theta}$ or the language model specified by the state s_n.

As in *TM-HMMS*, we assume multinomial distributions $\boldsymbol{\theta}$'s and $\boldsymbol{\phi}$'s are drawn from Dirichlet priors; and $\boldsymbol{\tau}$'s are governed by Beta distributions.

The inference for *TM-HMMSS* is exactly same as the inference for *TM-HMMS*, except the posterior distributions over word source $r_{n,i}$ is now

$$\begin{aligned}
p(r_{n,i}|\boldsymbol{r}_{-(n,i)}, \boldsymbol{s}, \boldsymbol{w}, \boldsymbol{\pi}) &\propto p(r_{n,i}|\boldsymbol{r}_{-(n,i)}, s_n, \boldsymbol{\pi}) \\
&\quad \cdot p(w_{n,i}|\boldsymbol{r}, \boldsymbol{s}, \boldsymbol{w}_{-(n,i)}, \boldsymbol{z}, \boldsymbol{\beta}), \qquad (3)
\end{aligned}$$

where the first term is integrated over all sessions and conditioned on the state assignment.

3.4 Supporting Multiple Parties

Since our primary focus is task-oriented dialogues between two parties, we assume every word source is associated with *two* sets of *LM*s—one for system/agent and another for user/client. This configuration is similar to *PolyLDA* (Mimno et al., 2009) or *LinkLDA* (Yano et al., 2009), such that utterances from different parties are treated as different languages or blog-post and comments pairs. In this work, we implement all models under this setting, but omit details in plate diagrams for the sake of simplicity.

In settings where the agent and client always alternate, each state emits both text before transitioning to the next state. This is the case in the *BusTime* dataset, where the spoken dialogue system enforces strict turn-taking. In settings where agents or client may produce more than one utterance in a row, each state emits either agent text *or* client text, then transitions to the next state. This is the case in the *TechSupport* corpus, where either conversant may send a message at any time.

3.5 Likelihood Estimation

To evaluate performance across different models, we compute the likelihood on held-out test set. For *TM-HMM* model, there are no local dependencies, and we therefore compute the marginal likelihood using the forward algorithm. However, for *TM-HMMS* and *TM-HMMSS* models, the latent topic distribution $\boldsymbol{\theta}$ creates local dependencies, rendering computation of marginal likeli-

hoods intractable. Hence, we use a Chib-style estimator (Wallach et al., 2009). Although it is computationally more expensive, it gives less biased approximation of marginal likelihood, even for finite samples. This ensures likelihood measurements are comparable across models.

4 Experiments

In this section, we examine the effectiveness of our models. We first evaluate our models qualitatively by exploring the inferred state diagram. We then perform quantitative analysis with log likelihood measurements and an ordering task on a held-out test set. We train all models with 80% of the entire dataset and use the rest for testing. We run the Gibbs samplers for 1000 iterations and update all hyper-parameters using slice sampling (Neal, 2003; Wallach, 2008) every 10 iterations. The training likelihood suggest all models converge within $500-800$ iterations. For all Chib-style estimators, we collect 100 samples along the Markov chain to approximate the marginal likelihood.

4.1 Qualitative Evaluation

Figure 3 shows the state diagram for *BusTime* corpus inferred by *TM-HMM* without any supervision.[5] Every dialogue is opened by asking the user to say a bus route, or to say "I'm not sure." It then transits to a state about location, e.g., origin and destination. Both these two states may continue to a confirmation step immediately after. After verifying all the necessary information, the system asks if the user wants "the next few buses".[6] Otherwise, the system follows up with the user on the particular date and time information. After system reads out bus times, the user has options to "repeat" or ask for subsequent schedules.

In addition, we also include the human-annotated dialogue flow in Figure 4 for reference (Williams, 2012). It only illustrates the most common design of system actions, without showing edge cases. Comparing these two figures, the dialogue flow inferred by our model along the most probable path (highlighted in bold red in Figure 3) is consistent with underlying design. Furthermore, our models are able to capture edge cases—omitted for space—through a more general and probabilistic fashion. In summary, our models yield a very similar flowchart to the underlying design in a completely unsupervised way.[7]

Figure 5 shows part of the flowchart for the *TechSupport* corpus, generated by the *TM-HMMSS* model.[8] A conversation usually starts with a welcome message from a customer support agent. Next, clients sometimes report a problem; otherwise, the agent gathers the client's identity. After these preliminaries, the agent usually checks the system version or platform settings. Then, information about the problem is exchanged, and a cycle ensues where agents propose solutions, and clients attempt them, reporting results. Usually, a conversation loops among these states until either the problem is resolved (as the case shown in the figure) or the client is left with a reference number for future follow-up (not shown due to space limit). Although technical support is task-oriented, the scope of possible issues is vast and not prescribed. The table in Figure 5 lists the top ranked words of selected topics—the categories clients often report problems in. It illustrates that, qualitatively, *TM-HMMSS* discovers both problem categories and conversation structures on our data.

As one of the baseline model, we also include a part of flowchart generated by *LM-HMM* model with similar settings of $T = 20$ states. Illustrated by the highlighted states in 6, *LM-HMM* model conflates interactions that commonly occur at the beginning and end of a dialogue—i.e., "acknowledge agent" and "resolve problem", since their underlying language models are likely to produce similar probability distributions over words. By incorporating topic information, our proposed models (e.g., *TM-HMMSS* in Figure 5) are able to enforce the state transitions towards more frequent flow patterns, which further helps to overcome the weakness of language model.

4.2 Quantitative Evaluation

In this section, we evaluate our models using log likelihood and an ordering task on a held-out test set. Both evaluation metrics measure the predictive power of a conversation model.

[5] Recall in *BusTime*, state transitions occur after each *pair* of system/user utterances, so we display them synchronously.

[6] The system was designed this way because most users say "yes" to this question, obviating the date and time.

[7] We considered various ways of making a quantitative evaluation of the inferred state diagram, and proved difficult. Rather than attempt to justify a particular sub-division of each "design states", we instead give several straightforward quantitative evaluations in the next section.

[8] Recall in this corpus, state transitions occur after emitting each agent *or* client utterances, which does *not* necessarily alternate in a dialogue, so we display client request and agent response separately.

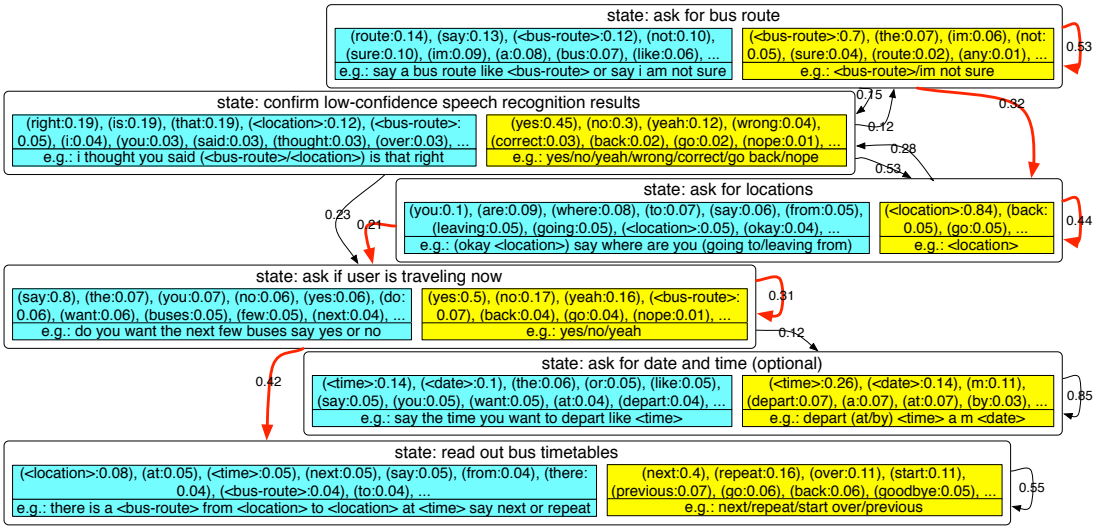


Figure 3: **(Upper)** Part of the flowchart inferred on *BusTime*, by *TM-HMM* model with $K = 10$ topics and $T = 10$ states. The most probable path is highlighted, which is consistent with the underlying design (Figure 4). Cyan blocks are system actions and yellow blocks are user responses. In every block, the upper cell shows the top ranked words marginalized over all topics and the lower cell shows some examples of that state. Transition probability cut-off is 0.1. States are labelled manually.

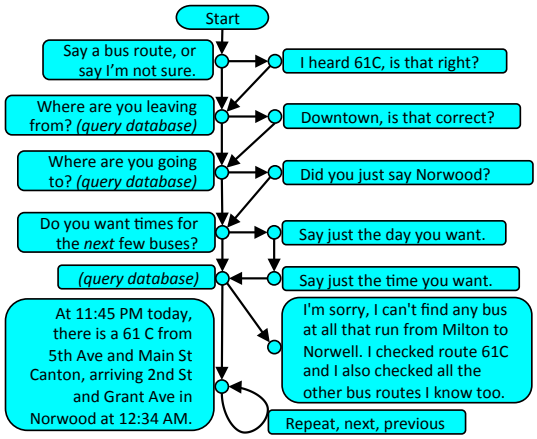


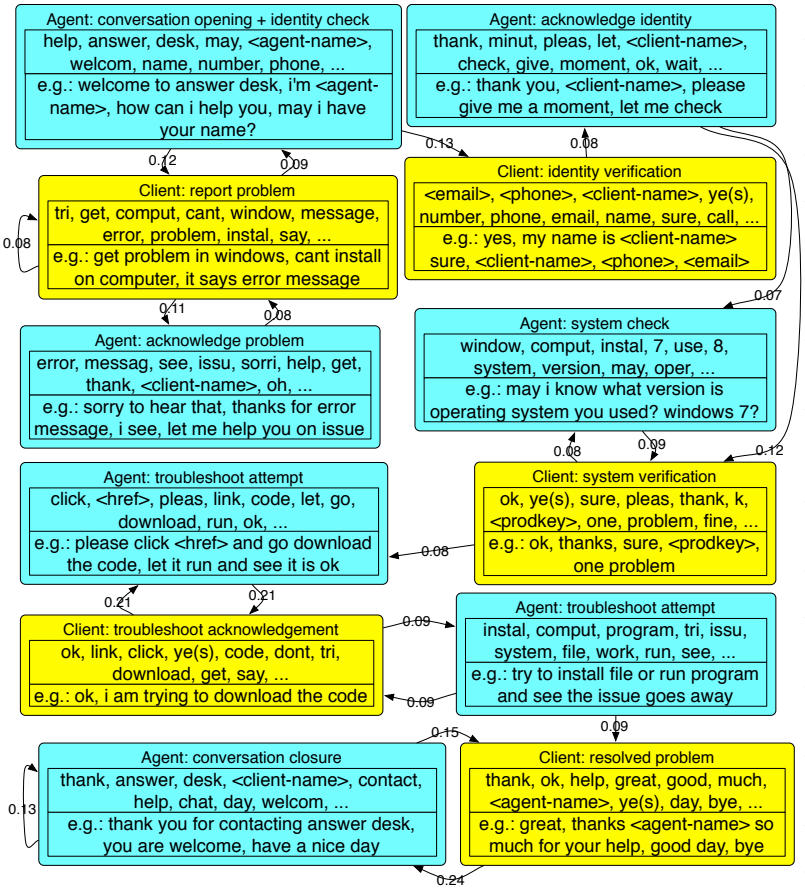
Figure 4: **(Left)** Hand-crafted reference flowchart for *BusTime* (Williams, 2012). Only the most common dialogue flows are displayed. System prompts shown are example paraphrases. Edge cases are not included.

Log Likelihood The likelihood metric measures the probability of generating the test set under a specified model. As shown in Figure 7, our models yield as good or better likelihood than *LM-HMM* and *LM-HMMS* models on both datasets under all settings. For our proposed models, *TM-HMMS* and *TM-HMMSS* perform better than *TM-HMM* on *TechSupport*, but not necessarily on *BusTime*. In addition, we notice that the marginal benefit of *TM-HMMSS* over *TM-HMM* is greater on *TechSupport* dataset, where each dialogue focuses on one of many possible tasks. This coincides with our belief that topics are more conversation dependent and shared across the entire corpus in customer support data—i.e., different clients in different sessions might ask about similar issues.

Ordering Test Ritter et al. (2010) proposes an evaluation based on rank correlation coefficient, which measures the degree of similarity between any two orderings over sequential data. They use Kendall's τ as evaluation metric, which is based on the agreement between pairwise orderings of two sequences (Kendall, 1938). It ranges from -1 to $+1$, where $+1$ indicates an identical ordering and -1 indicates a reverse ordering. The idea is to generate all permutations of the utterances in a dialogue (including true ordering), and compute the log likelihood for each under the model. Then, Kendall's τ is computed between the most probable permutation and true ordering. The result is the average of τ values for all dialogues in test corpus.

Ritter et al. (2010) limits their dataset by choosing Twitter dialogues containing 3 to 6 posts (utterances), making it tractable to enumerate all permutations. However, our datasets are much larger, and enumerating all possible permutations of dialogues with more than 20 or 30 utterances is infeasible. Instead, we incrementally build up the permutation set by adding one random permutation at a time, and taking the most probable permutation after each addition. If this process were continued (intractably!) until all permutations are enumerated, the true value of Kendall's τ test would be reached. In practice, the value appears to plateau after a few dozen measurements.

We present our results in Figure 8. Our models consistently perform as good or better than



Topic	Top Ranked Words
purchase	microsoft, store, purchas, able, get, sir, order, site, mr, contact, mac, ...
browser	internet, explor, browser, ie, open, websit, googl, download, click, chrome, ...
backup	file, restor, system, comput, back, folder, creat, option, dont, delet, ...
boot	comput, boot, mode, option, disc, safe, recoveri, repair, back, clean, cd, disk, ...
update	updat, window, servic, instal, pack, run, comput, download, check, restart, inform, system, error, fix, ...
network	connect, internet, printer, comput, network, pc, print, access, wireless, hp, cable, adapt, router, speed, ...
anti-virus	viru, scan, comput, remov, secur, run, system, anti, essenti, infect, defend, softwar, program, protect, antiviru, malwar, ...
hardware	driver, devic, drive, dvd, cd, hardwar, issu, model, laptop, plug, software, usb, ...
windows	window, upgrad, 8, download, 7, instal, bit, vista, pro, system, ...
office	offic, 2010, word, microsoft, home, excel, version, 2007, student, document, trial, 2013, ...
outlook	outlook, account, email, mail, microsoft, com, live, password, profil, contact, creat, server, access, ...
license	key, product, activ, purchas, licens, valid, verifi, id, disc, pro, grenuin, ...
facility	window, 8, comput, instal, manufactur, system, oem, 7, pc, hp, ...

Figure 5: Part of flowchart (left) and topic table (right) on *TechSupport* dataset, generated by *TM-HMMSS* model under settings of $K = 20$ topics and $T = 20$ states. The topic table lists top ranked words in issues discussed in the chats. Cyan blocks are system actions and yellow blocks are user responses. In every block, the upper cell shows top ranked words, and the lower cell shows example string patterns of that state. Transition probability cut-off is 0.05. States and topics are labelled manually.

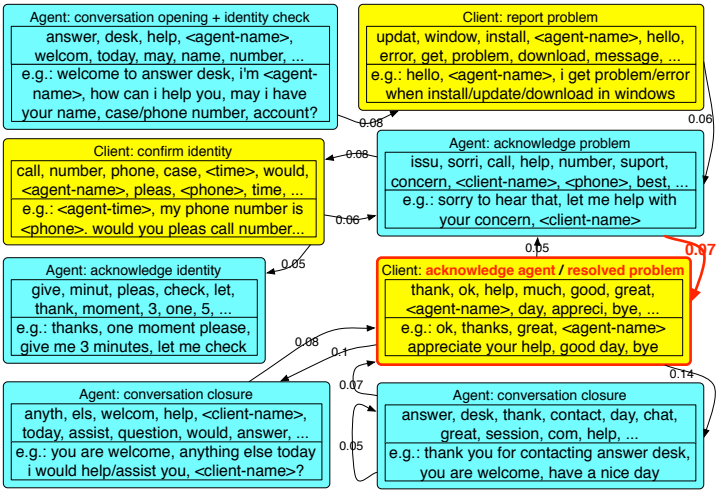


Figure 6: Part of flowchart on *TechSupport* dataset, generated by *LM-HMM* model with $T = 20$ states. Cyan blocks are system actions and yellow blocks are user responses. In every block, the upper cell shows the top ranked words, and the lower cell shows example word sequences or string patterns of that state. Transition probability cut-off is 0.05. States are labelled manually. A poorly-inferred state is highlighted, which seems to conflate the "acknowledge agent" and "resolve problem" states, and *TM-HMMSS* model has properly disentangled (Figure 5).

the baseline models. For *BusTime* data, all models perform relatively well except *LM-HMM* which only indicates weak correlations. *TM-HMM* out-performs all other models under all settings. This is also true for *TechSupport* dataset. *LM-HMMS*, *TM-HMMS* and *TM-HMMSS* models perform considerably well on *BusTime*, but not on *TechSupport* data. These three models allow words to be generated from additional sources other than states. Although this improves log likelihood, it is possible these models encode less information about the state sequences, at least in the more diffuse *TechSupport* data. In summary, under both quantitative evaluation measures, our models advance state-of-the-art, however *which* of our models is best depends on the application.

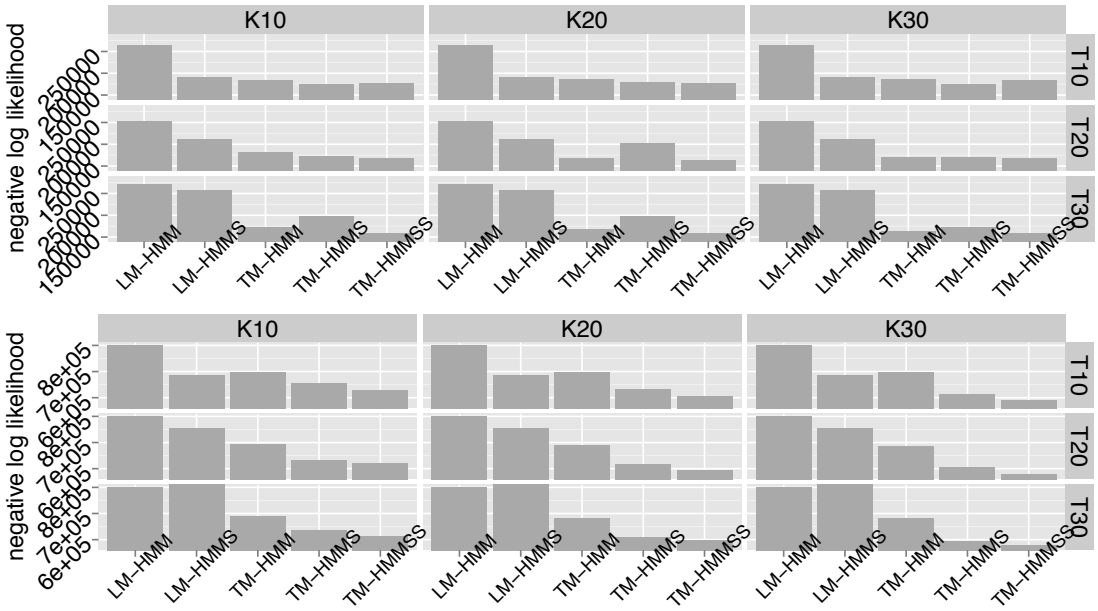


Figure 7: Negative log likelihood on *BusTime* (upper) and *TechSupport* (lower) datasets (smaller is better) under different settings of topics K and states T.

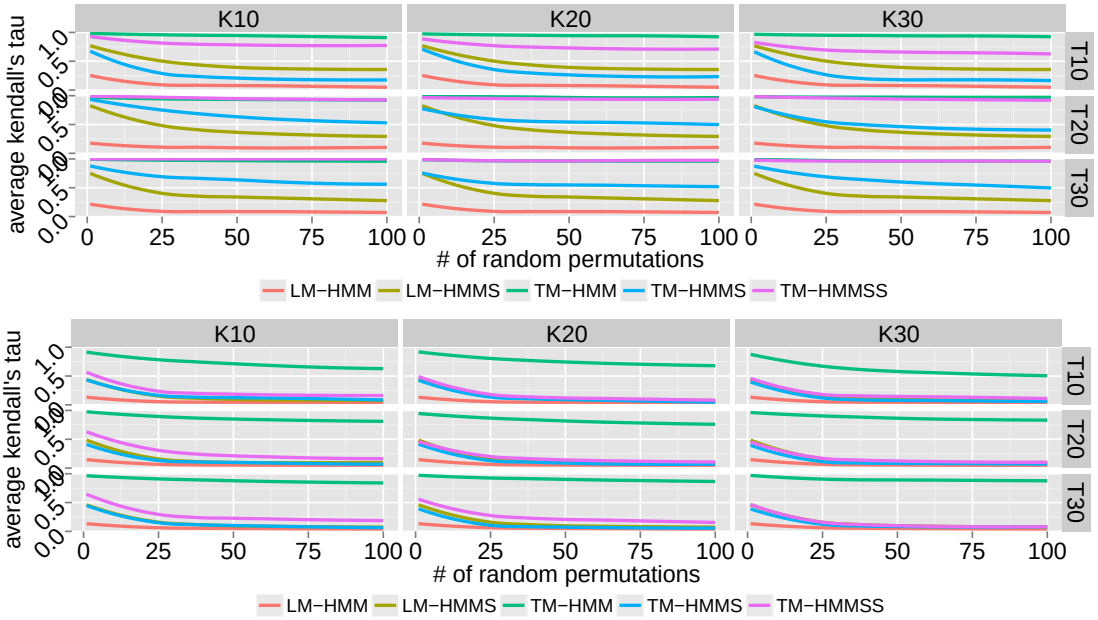


Figure 8: Average Kendall's τ measure on *BusTime* (upper) and *TechSupport* (lower) datasets (larger is better) against number of random permutations, under various settings of topics K and states T.

5 Conclusion and Future Work

We have presented three new unsupervised models to discover latent structures in task-oriented dialogues. We evaluated on two very different corpora—logs from spoken, human-computer dialogues about bus time, and logs of textual, human-human dialogues about technical support. We have shown our models yield superior performance both qualitatively and quantitatively.

One possible avenue for future work is scalability. Parallelization (Asuncion et al., 2012) or online learning (Doucet et al., 2001) could significantly speed up inference. In addition to MCMC, another class of inference method is variational Bayesian analysis (Blei et al., 2003; Beal, 2003), which is inherently easier to distribute (Zhai et al., 2012) and online update (Hoffman et al., 2010).

Acknowledgments

We would like to thank anonymous reviewers and Jordan Boyd-Graber for their valuable comments. We are also grateful to Alan Ritter and Bill Dolan for their helpful discussions; and Kai (Anthony) Lui for providing *TechSupport* dataset.

References

Arthur Asuncion, Padhraic Smyth, Max Welling, David Newman, Ian Porteous, and Scott Triglia, 2012. *Distributed Gibbs sampling for latent variable models*.

Satanjeev Banerjee and Alexander I Rudnicky. 2006. A texttiling based approach to topic boundary detection in meetings. In *INTERSPEECH*.

Srinivas Bangalore, Giuseppe Di Fabbrizio, and Amanda Stent. 2006. Learning the structure of task-driven human-human dialogs. In *ACL*, Stroudsburg, PA, USA.

Regina Barzilay and Lillian Lee. 2004. Catching the drift: Probabilistic content models, with applications to generation and summarization. In *NAACL*, pages 113–120.

Matthew J. Beal. 2003. *Variational Algorithms for Approximate Bayesian Inference*. Ph.D. thesis.

Alan W Black, Susanne Burger, Alistair Conkie, Helen Hastie, Simon Keizer, Nicolas Merigaud, Gabriel Parent, Gabriel Schubiner, Blaise Thomson, D. Williams, Kai Yu, Steve Young, and Maxine Eskenazi. 2010. Spoken dialog challenge 2010: Comparison of live and control test results. In *SIGDIAL*.

David M. Blei, Andrew Ng, and Michael Jordan. 2003. Latent Dirichlet allocation. *JMLR*.

Ananlada Chotimongkol. 2008. *Learning the Structure of Task-oriented Conversations from the Corpus of In-domain Dialogs*. Ph.D. thesis.

Nigel Crook, Ramn Granell, and Stephen G. Pulman. 2009. Unsupervised classification of dialogue acts using a dirichlet process mixture model. In *SIGDIAL*.

Hal Daumé III and Daniel Marcu. 2006. Bayesian query-focused summarization. In *ACL-44: Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics*, pages 305–312, Morristown, NJ, USA. Association for Computational Linguistics.

David DeVault, Kallirroi Georgila, Ron Artstein, Fabrizio Morbini, David Traum, Stefan Scherer, Albert Rizzo, and Louis-Philippe Morency. 2013. Verbal indicators of psychological distress in interactive dialogue with a virtual human. In *SIGDIAL*.

Arnaud Doucet, Nando De Freitas, and Neil Gordon, editors. 2001. *Sequential Monte Carlo methods in practice*. Springer Texts in Statistics.

Michel Galley, Kathleen McKeown, Eric Fosler-Lussier, and Hongyan Jing. 2003. Discourse segmentation of multi-party conversation. In *ACL*.

Sharon Goldwater and Thomas L. Griffiths. 2007. A fully Bayesian approach to unsupervised part-of-speech tagging. In *ACL*.

Thomas L. Griffiths and Mark Steyvers. 2004. Finding scientific topics. *PNAS*, 101(Suppl 1):5228–5235.

James Henderson and Oliver Lemon. 2008. Mixture model POMDPs for efficient handling of uncertainty in dialogue management. In *ACL*.

Matthew Hoffman, David M. Blei, and Francis Bach. 2010. Online learning for latent Dirichlet allocation. In *NIPS*.

Pei-yun Hsueh, Johanna D. Moore, and Steve Renals. 2006. Automatic segmentation of multiparty dialogue. In *EACL*.

Lluís F. Hurtado, Joaquin Planells, Encarna Segarra, Emilio Sanchis, and David Griol. 2010. A stochastic finite-state transducer approach to spoken dialog management. In *INTERSPEECH*.

Dan Jurafsky, Elizabeth Shriberg, and Debra Biasca. 1997. Switchboard SWBD-DAMSL shallow-discourse-function annotation coders manual. *Institute of Cognitive Science Technical Report*, pages 97–02.

Maurice G. Kendall. 1938. A new measure of rank correlation. Biometrika Trust.

Staffan Larsson and David R. Traum. 2000. Information state and dialogue management in the TRINDI dialogue move engine toolkit. *Natural Language Engineering*, 5(3/4):323–340.

Esther Levin, Roberto Pieraccini, and Wieland Eckert. 2000. A stochastic model of human-machine interaction for learning dialogue strategies. *IEEE Trans on Speech and Audio Processing*, 8(1):11–23.

Jingjing Liu, Stephanie Seneff, and Victor Zue. 2010. Dialogue-oriented review summary generation for spoken dialogue recommendation systems. In *NAACL*.

David Mimno, Hanna Wallach, Jason Naradowsky, David Smith, and Andrew McCallum. 2009. Polylingual topic models. In *EMNLP*.

Gabriel Murray, Steve Renals, and Jean Carletta. 2005. Extractive summarization of meeting recordings. In *European Conference on Speech Communication and Technology*.

Radford M. Neal. 2000. Markov chain sampling methods for Dirichlet process mixture models. *Journal of Computational and Graphical Statistics*, 9(2):249–265.

Radford M. Neal. 2003. Slice sampling. *Annals of Statistics*, 31:705–767.

Matthew Purver, Konrad Körding, Thomas L. Griffiths, and Joshua Tenenbaum. 2006. Unsupervised topic modelling for multi-party spoken discourse. In *ACL*.

Verena Rieser and Oliver Lemon. 2010. Natural language generation as planning under uncertainty for spoken dialogue systems. In *EMNLP*.

Alan Ritter, Colin Cherry, and Bill Dolan. 2010. Unsupervised modeling of twitter conversations. In *NAACL*.

Satinder Singh, Diane Litman, Michael Kearns, and Marilyn Walker. 2002. Optimizing dialogue management with reinforcement learning: Experiments with the NJFun system. *Journal of Artificial Intelligence Research*.

Andreas Stolcke, Noah Coccaro, Rebecca Bates, Paul Taylor, Carol Van Ess-Dykema, Klaus Ries, Elizabeth Shriberg, Daniel Jurafsky, Rachel Martin, and Marie Meteer. 2000. Dialogue act modeling for automatic tagging and recognition of conversational speech. *Computational Linguistics*, September.

David R Traum and Staffan Larsson. 2003. The information state approach to dialogue management. In *Current and new directions in discourse and dialogue*, pages 325–353.

Hanna M. Wallach, Iain Murray, Ruslan Salakhutdinov, and David Mimno. 2009. Evaluation methods for topic models. In *ICML*.

Hanna M. Wallach. 2008. *Structured Topic Models for Language*. Ph.D. thesis, University of Cambridge.

Jason D. Williams and Suhrid Balakrishnan. 2009. Estimating probability of correctness for ASR N-best lists. In *SIGDIAL*.

Jason D. Williams. 2012. Challenges and opportunities for state tracking in statistical spoken dialog systems: Results from two public deployments. *Journal of Selected Topics in Signal Processing*.

Tae Yano, William W. Cohen, and Noah A. Smith. 2009. Predicting response to political blog posts with topic models. In *NAACL*, pages 477–485, Stroudsburg, PA, USA. ACL.

Steve Young. 2006. Using POMDPs for dialog management. In *Proceedings of the 1st IEEE/ACL Workshop on Spoken Language Technologies (SLT06)*.

Ke Zhai, Jordan Boyd-Graber, Nima Asadi, and Mohamad Alkhouja. 2012. Mr. LDA: A flexible large scale topic modeling package using variational inference in mapreduce. In *WWW*.