Using Automatically Transcribed Dialogs to Learn User Models in a Spoken Dialog System

Umar Syed
Department of Computer Science
Princeton University
Princeton, NJ 08540, USA
usyed@cs.princeton.edu

Jason D. Williams
Shannon Laboratory
AT&T Labs — Research
Florham Park, NJ 07932, USA
jdw@research.att.com

Abstract

We use an EM algorithm to learn user models in a spoken dialog system. Our method requires automatically transcribed (with ASR) dialog corpora, plus a model of transcription errors, but does not otherwise need any manual transcription effort. We tested our method on a voice-controlled telephone directory application, and show that our learned models better replicate the true distribution of user actions than those trained by simpler methods and are very similar to user models estimated from manually transcribed dialogs.

1 Introduction and Background

When designing a dialog manager for a spoken dialog system, we would ideally like to try different dialog management strategies on the actual user population that will be using the system, and select the one that works best. However, users are typically unwilling to endure this kind of experimentation. The next-best approach is to build a model of user behavior. That way we can experiment with the model as much as we like without troubling actual users.

Of course, for these experiments to be useful, a high-quality user model is needed. The usual method of building a user model is to estimate it from transcribed corpora of human-computer dialogs. However, manually transcribing dialogs is expensive, and consequently these corpora are usually small and sparse. In this work, we propose a method of building user models that does not operate on manually transcribed dialogs, but instead uses dialogs that have been transcribed by an automatic speech recognition (ASR) engine. Since this process is error-prone, we cannot assume that the transcripts will accurately reflect the users' true actions and internal states. To handle this uncertainty, we employ an EM algorithm that treats this information as unobserved data. Although this approach does not directly employ manually transcribed dialogs, it does require a confusion model for the ASR engine, which *is* estimated from manually transcribed dialogs. The key benefit is that the number of manually transcribed dialogs required to estimate an ASR confusion model is much smaller, and is fixed with respect to the complexity of the user model.

Many works have estimated user models from transcribed data (Georgila et al., 2006; Levin et al., 2000; Pietquin, 2004; Schatzmann et al., 2007). Our work is novel in that we do not assume we have access to the correct transcriptions at all, but rather have a model of how errors are made. EM has previously been applied to estimation of user models: (Schatzmann et al., 2007) cast the user's internal state as a complex hidden variable and estimate its transitions using the true user actions with EM. Our work employs EM to infer the model of user actions, not the model of user goal evolution.

2 Method

Before we can estimate a user model, we must define a larger model of human-computer dialogs, of which the user model is just one component. In this section we give a general description of our dialog model; in Section 3 we instantiate the model for a voice-controlled telephone directory.

We adopt a probabilistic dialog model (similar

Proceedings of ACL-08: HLT, Short Papers (Companion Volume), pages 121–124,
Columbus, Ohio, USA, June 2008.

to (Williams and Young, 2007)), depicted schematically as a graphical model in Figure 1. Following the convention for graphical models, we use directed edges to denote conditional dependencies among the variables. In our dialog model, a dialog transcript $\mathbf{x}$ consists of an alternating sequence of system actions and observed user actions: $\mathbf{x} = (S_0, \tilde{A}_0, S_1, \tilde{A}_1, \ldots)$. Here S_t denotes the system action, and $\tilde{A}_t$ the output of the ASR engine when applied to the true user action A_t.

A dialog transcript $\mathbf{x}$ is generated by our model as follows: At each time t, the system action is S_t and the unobserved user state is U_t. The user state indicates the user's hidden goal and relevant dialog history which, due to ASR confusions, is known with certainty only to the user. Conditioned on (S_t, U_t), the user draws an unobserved action A_t from a distribution $\Pr(A_t \mid S_t, U_t; \theta)$ parameterized by an unknown parameter θ. For each user action A_t, the ASR engine produces a hypothesis $\tilde{A}_t$ of what the user said, drawn from a distribution $\Pr(\tilde{A}_t \mid A_t)$, which is the ASR confusion model. The user state U_t is updated to U_{t+1} according to a deterministic distribution $\Pr(U_{t+1} \mid S_{t+1}, U_t, A_t, \tilde{A}_t)$. The system outputs the next system action S_{t+1} according to its dialog management policy. Concretely, the values of S_t, U_t, A_t and $\tilde{A}_t$ are all assumed to belong to finite sets, and so all the conditional distributions in our model are multinomials. Hence θ is a vector that parameterizes the user model according to $\Pr(A_t = a \mid S_t = s, U_t = u; \theta) = \theta_{asu}$.

The problem we are interested in is estimating θ given the set of dialog transcripts $\mathcal{X}$, $\Pr(\tilde{A}_t \mid A_t)$ and $\Pr(U_{t+1} \mid S_{t+1}, U_t, A_t, \tilde{A}_t)$. Here, we assume that $\Pr(\tilde{A}_t \mid A_t)$ is relatively straightforward to estimate: for example, ASR models that rely a simple confusion rate and uniform substitutions (which can be estimated from small number of transcriptions) have been used to train dialog systems which outperform traditional systems (Thomson et al., 2007). Further, $\Pr(U_{t+1} \mid S_{t+1}, U_t, A_t, \tilde{A}_t)$ is often deterministic and tracks dialog history relevant to action selection — for example, whether the system correctly or incorrectly confirms a slot value. Here we assume that it can be easily hand-crafted.

Formally, given a set of dialog transcripts $\mathcal{X}$, our goal is find a set of parameters θ^* that maximizes the log-likelihood of the observed data, i.e.,

$$\theta^* = \arg\max_{\theta} \log \Pr(\mathcal{X} \mid \theta)$$

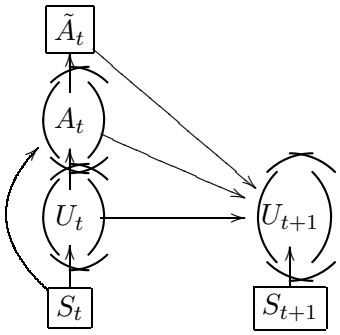

Figure 1: A probabilistic graphical model of a human-computer dialog. The boxed variables are observed; the circled variables are unobserved.

Unfortunately, directly computing θ^* in this equation is intractable. However, we can efficiently approximate θ^* via an expectation-maximization (EM) procedure (Dempster et al., 1977). For a dialog transcript $\mathbf{x}$, let $\mathbf{y}$ be the corresponding sequence of unobserved values: $\mathbf{y} = (U_0, A_0, U_1, A_1, \ldots)$. Let $\mathcal{Y}$ be the set of all sequences of unobserved values corresponding to the data set $\mathcal{X}$. Given an estimate $\theta^{(t-1)}$, a new estimate $\theta^{(t)}$ is produced by

$$\theta^{(t)} = \arg\max_{\theta} E_{\mathcal{Y}} \left[\log \Pr(\mathcal{X}, \mathcal{Y} \mid \theta) \;\middle|\; \mathcal{X}, \theta^{(t-1)} \right]$$

The expectation in this equation is taken over all possible values for $\mathcal{Y}$. Both the expectation and its maximization are easy to compute. This is because our dialog model has a chain-like structure that closely resembles an Hidden Markov Model, so a forward-backward procedure can be employed (Rabiner, 1990). Under fairly mild conditions, the sequence $\theta^{(0)}, \theta^{(1)}, \ldots$ converges to a stationary point estimate of θ^* that is usually a local maximum.

3 Target Application

To test the method, we applied it to a voice-controlled telephone directory. This system is currently in use in a large company with many thousands of employees. Users call the directory system and provide the name of a callee they wish to be connected to. The system then requests additional

information from the user, such as the callee's location and type of phone (office, cell). Here is a small fragment of a typical dialog with the system:

$S_0 =$ `First and last name?`
$A_0 =$ "John Doe" [$\tilde{A}_0 =$ *Jane Roe*]
$S_1 =$ `Jane Roe.  Office or cell?`
$A_1 =$ "No, no, John Doe" [$\tilde{A}_1 =$ *No*]
$S_2 =$ `First and last name?`
...

Because the telephone directory has many names, the number of possible values for A_t, $\tilde{A}_t$, and S_t is potentially very large. To control the size of the model, we first assumed that the user's intended callee does not change during the call, which allows us to group many user actions together into generic placeholders e.g. $A_t =$ `FirstNameLastName`. After doing this, there were a total of 13 possible values for A_t and $\tilde{A}_t$, and 14 values for S_t.

The user state consists of three bits: one bit indicating whether the system has correctly recognized the callee's name, one bit indicating whether the system has correctly recognized the callee's "phone type" (office or cell), and one bit indicating whether the user has said the callee's geographic location (needed for disambiguation when several different people share the same name). The deterministic distribution $\Pr(U_{t+1} \mid S_{t+1}, U_t, A_t, \tilde{A}_t)$ simply updates the user state after each dialog turn in the obvious way. For example, the "name is correct" bit of U_{t+1} is set to 0 when S_{t+1} is a confirmation of a name which doesn't match A_t.

Recall that the user model is a multinomial distribution $\Pr(A_t \mid S_t, U_t; \theta)$ parameterized by a vector θ. Based on the number user actions, system actions, and user states, θ is a vector of $(13-1) \times 14 \times 8 = 1344$ unknown parameters for our target application.

4 Experiments

We conducted two sets of experiments on the telephone directory application, one using simulated data, and the other using dialogs collected from actual users. Both sets of experiments assumed that all the distributions in Figure 1, except the user model, are known. The ASR confusion model was estimated by transcribing 50 randomly chosen dialogs from the training set in Section 4.2 and calculating the frequency with which the ASR engine recognized $\tilde{A}_t$ such that $\tilde{A}_t \neq A_t$. The probabilities $\Pr(\tilde{A}_t \mid A_t)$ were then constructed by assuming that, when the ASR engine makes an error recognizing a user action, it substitutes another randomly chosen action.

4.1 Simulated Data

Recall that, in our parameterization, the user model is $\Pr(A_t = a \mid S_t = s, U_t = u; \theta) = \theta_{asu}$. So in this set of experiments, we chose a reasonable, hand-crafted value for θ, and then generated synthetic dialogs by following the probabilistic process depicted in Figure 1. In this way, we were able to create synthetic training sets of varying sizes, as well as a test set of 1000 dialogs. Each generated dialog $\mathbf{d}$ in each training/test set consisted of a sequence of values for all the observed and unobserved variables: $\mathbf{d} = (S_0, U_0, A_0, \tilde{A}_0, \ldots)$.

For a training/test set $\mathcal{D}$, let $K^{\mathcal{D}}_{asu}$ be the number of times t, in all the dialogs in $\mathcal{D}$, that $A_t = a$, $S_t = s$, and $U_t = u$. Similarly, let $\widetilde{K}^{\mathcal{D}}_{as}$ be the number of times t that $\tilde{A}_t = a$ and $S_t = s$.

For each training set $\mathcal{D}$, we estimated θ using the following three methods:

1. *Manual*: Let θ be the maximum likelihood estimate using manually transcribed data, i.e., $\theta_{asu} = \frac{K^{\mathcal{D}}_{asu}}{\sum_a K^{\mathcal{D}}_{asu}}$.

2. *Automatic*: Let θ be the maximum likelihood estimate using automatically transcribed data, i.e., $\theta_{asu} = \frac{\widetilde{K}^{\mathcal{D}}_{as}}{\sum_a \widetilde{K}^{\mathcal{D}}_{as}}$. This approach ignores transcription errors and assumes that user behavior depends only on the observed data.

3. *EM*: Let θ be the estimate produced by the EM algorithm described in Section 2, which uses the automatically transcribed data and the ASR confusion model.

Now let $\mathcal{D}$ be the test set. We evaluated each user model by calculating the normalized log-likelihood of the model with respect to the *true* user actions in $\mathcal{D}$:

$$\ell(\theta) = \frac{\sum_{a,s,u} K^{\mathcal{D}}_{asu} \log \theta_{asu}}{|\mathcal{D}|}$$

$\ell(\theta)$ is essentially a measure of how well the user model parameterized by θ replicates the distribution

of user actions in the test set. The normalization is to allow for easier comparison across data sets of differing sizes.

We repeated this entire process (generating training and test sets, estimating and evaluating user models) 50 times. The results presented in Figure 2 are the average of those 50 runs. They are also compared to the normalized log-likelihood of the "Truth", which is the actual parameter θ used to generated the data.

The EM method has to estimate a larger number of parameters than the Automatic method (1344 vs. 168). But as Figure 2 shows, after observing enough dialogs, the EM method is able to leverage the hidden user state to learn a better model of user behavior, with an average normalized log-likelihood that falls about halfway between that of the models produced by the Automatic and Manual methods.

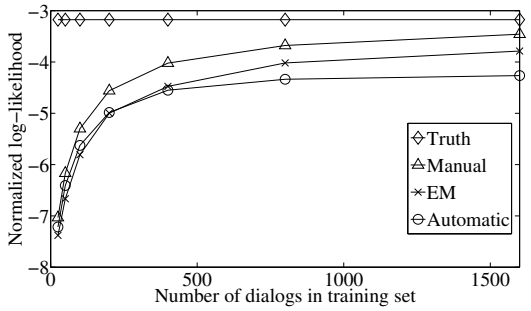

Figure 2: **Normalized log-likelihood of each model type with respect to the test set vs. size of training set.** Each data point is the average of 50 runs. For the largest training set, the EM models had higher normalized log-likelihood than the Automatic models in 48 out of 50 runs.

4.2 Real Data

We tested the three estimation methods from the previous section on a data set of 461 real dialogs, which we split into a training set of 315 dialogs and a test set of 146 dialogs. All the dialogs were both manually and automatically transcribed, so that each of the three methods was applicable. The normalized log-likelihood of each user model, with respect to both the training and test set, is given in Table 1. Since the output of the EM method depends on a random choice of starting point $\theta^{(0)}$, those results were averaged over 50 runs.

	Training Set $\ell(\theta)$	Test Set $\ell(\theta)$
Manual	-2.87	-3.73
EM	-3.90	-4.33
Automatic	-4.60	-5.80

Table 1: **Normalized log-likelihood of each model type with respect to the training set and the test set.** The EM values are the average of 50 runs. The EM models had higher normalized log-likelihood than the Automatic model in 50 out of 50 runs.

5 Conclusion

We have shown that user models can be estimated from automatically transcribed dialog corpora by modeling dialogs within a probabilistic framework that accounts for transcription errors in a principled way. This method may lead to many interesting future applications, such as continuous learning of a user model while the dialog system is on-line, enabling automatic adaptation.

References

AP Dempster, NM Laird, and DB Rubin. 1977. Maximum likelihood from incomplete data via the em algorithm. *J. Royal Stat. Soc.*, 39:1–38.

K Georgila, J Henderson, and O Lemon. 2006. User simulation for spoken dialogue systems: Learning and evaluation. In *Proc ICSLP, Pittsburgh, USA*.

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

O Pietquin. 2004. *A framework for unsupervised learning of dialogue strategies*. Ph.D. thesis, Faculty of Engineering, Mons (TCTS Lab), Belgium.

LR Rabiner, 1990. *A tutorial on hidden Markov models and selected applications in speech recognition*, pages 267–296. Morgan Kaufmann Publishers, Inc.

J Schatzmann, B Thomson, and SJ Young. 2007. Statistical user simulation with a hidden agenda. In *Proc SIGDial, Antwerp*, pages 273–282.

B Thomson, J Schatzmann, K Welhammer, H Ye, and SJ Young. 2007. Training a real-world POMDP-based dialog system. In *Proc NAACL-HLT Workshop Bridging the Gap: Academic and Industrial Research in Dialog Technologies, Rochester, New York, USA*, pages 9–17.

JD Williams and SJ Young. 2007. Partially observable Markov decision processes for spoken dialog systems. *Computer Speech and Language*, 21(2):393–422.