Engineering Statistical Dialog State Trackers: A Case Study on DSTC

Daejoong Kim[1], Jaedeug Choi[1], Kee-Eung Kim[1], Jungsu Lee[2], and Jinho Sohn[2]

[1] Department of Computer Science, KAIST, South Korea
{djkim,jdchoi}@ai.kaist.ac.kr, {kekim}@cs.kaist.ac.kr
[2] LG Electronics, South Korea
{jungsu.lee,jinho.sohn}@lge.com

Abstract. We describe our experience with engineering the dialog state tracker for the first Dialog State Tracking Challenge (DSTC). Dialog trackers are one of the essential components of dialog systems which are used to infer the true user goal from the speech processing results. We explain the main parts of our tracker: the observation model, the belief refinement model, and the belief transformation model. We also report experimental results on a number of approaches to the models, and compare the overall performance of our tracker to other submitted trackers. This technical report is a companion to the shortened version presented at SIGDIAL 2013.

Keywords: Spoken dialog system, dialog state tracking, belief tracking, dialog manager, Dialog State Tracking Challenge, DSTC

1 Introduction

In spoken dialog systems (SDSs), one of the main challenges is to identify the user goal from her utterances. The significance of accurately identifying the user goal, referred to as *dialog state tracking* or *belief tracking*, has emerged from the need for SDSs to be robust to inevitable errors in the spoken language understanding (SLU), which is a key to improving the performance of SDSs.

A number of studies have been conducted to track the dialog state through multiple dialog turns using a probabilistic framework, treating SLU results as noisy observations and maintaining probability distribution (*i.e.*, belief) on user goals. Bohus and Rudnicky [1] proposed a framework which uses the compact representation of beliefs and a generalized linear regression model for belief updates. Mehta *et al.* [2] proposed a tree-structured Bayesian network to represent beliefs. The approach enables leveraging the dependency between domain concepts while avoiding intensive computation. Additionally, many approaches in SDSs [3–6] have proposed to use a partially observable Markov decision process (POMDP) [7]. The advantage of this approach is that POMDPs provide a unified decision theoretic model for both tracking beliefs probabilistically and optimizing dialog strategies.

In this paper, we share our experience and lessons learned from developing the dialog state tracker that participated in the first Dialog State Tracking Challenge (DSTC) [8][3].

[3] http://research.microsoft.com/en-us/events/dstc

Our tracker is based on the belief update in the the POMDP framework, particularly the hidden information state (HIS) model [9] and the partition recombination method [10]. Our main contribution lies in experimenting with a number of techniques to engineer the POMDP belief update, and providing analyses of experimental results on the DSTC datasets in terms of the various performance measures used in the challenge.

2 Dialog State Tracking

Our dialog state tracker mainly follows the belief update in HIS-POMDP [9]. The SDS executes system action a, and the user with goal g responds to the system with utterance u. The SLU unit processes the utterance and generates the result as an N-best list $\boldsymbol{o} = [\langle \tilde{u}_1, f_1 \rangle, \ldots, \langle \tilde{u}_N, f_N \rangle]$ of the hypothesized user utterance $\tilde{u}_i$ and its associated confidence score f_i. Because the SLU is not perfect, the system cannot exactly identify the user goal. The system thus maintains a probability distribution over user goals, called a *belief.* In addition, the system groups user goals into equivalence classes and assigns a single probability for each equivalence class since the number of user goals is often too large to perform individual belief updates for all possible user goals. The equivalence classes are called partitions and denoted as ψ. Hence, given the current belief b, system action a, and recognized N-best list $\boldsymbol{o}$, the dialog state tracker updates the belief b' over partitions as follows:

$$b'(\psi') \propto \sum_u \Pr(\boldsymbol{o}|u) \Pr(u|\psi', a) \Pr(\psi'|\psi) b(\psi) \tag{1}$$

where $\Pr(\boldsymbol{o}|u)$ is the observation model, $\Pr(u|\psi, a)$ is the user utterance model, $\Pr(\psi'|\psi)$ is the belief refinement model and ψ is the parent of ψ'. We describe each model in the following sections.

2.1 Observation Model

The observation model $\Pr(\boldsymbol{o}|u)$ is the probability that the SLU produces the N-best list $\boldsymbol{o}$ when the user utterance is u. In HIS-POMDP, a partition ψ of the user goal is iteratively split into two partitions ψ'_i and $\psi - \psi'_i$, $i = 1, \ldots, N$, where ψ'_i is obtained by treating the SLU result $\tilde{u}_i$ as the true user utterance u. Thus, the observation probabilities used in updating the beliefs of ψ'_i and $\psi - \psi'_i$ are $\Pr(\tilde{u}_i, f_i|u)$ and $1 - \Pr(\tilde{u}_i, f_i|u)$. We experimented with the following three models for the observation model.

Confidence score model: as in HIS-POMDP, this model assumes that the confidence score f_i obtained from the SLU is exactly the probability of generating the hypothesized user utterance $\tilde{u}_i$. Hence, $f_i = \Pr(\tilde{u}_i, f_i|u)$. One of the practical concerns using this model is that, when the SLU produces a confidence score $f_i = 1$ for any $\tilde{u}_i$, all the probability mass is concentrated on the partitions associated with $\tilde{u}_i$, and as a consequence the belief of any other user goal becomes zero. In the DSTC datasets, we observed a number of cases where the $\tilde{u}_i$ is incorrect even when $f_i = 1$. To mitigate this problem, we discount the confidence score so that $\Pr(\tilde{u}_i, f_i|u) = \gamma f_i$ with $0 < \gamma < 1$ whose best value is obtained from the training datasets via cross-validation.

Histogram model: this model estimates a function that maps the confidence score to the probability of correctness. We constructed a histogram of confidence scores from the training datasets to obtain the empirical probability $\Pr(cor(f_i))$ of whether the entry associated with confidence score f_i is a correct hypothesis or not. We then used $\Pr(\tilde{u}_i, f_i|u) = \Pr(cor(f_i))$ for the observation probability.

Generative model: Williams [11] proposed a generative model that uses various characteristics of the SLU. We adopted a simplified version of the model, which does not use other information than the confidence score: the observation probability is modeled as $\Pr(\tilde{u}_i, f_i|u) = \Pr(cor(i))\Pr(f_i|cor(i))$ where $\Pr(cor(i))$ is the probability of the i-th entry being a correct hypothesis and $\Pr(f_i|cor(i))$ is the probability of the i-th entry having confidence score f_i when it is a correct hypothesis. We used the empirical distribution from the training datasets for the probabilities.

2.2 User Utterance Model

The user utterance model $\Pr(u|\psi, a)$ indicates how the user responds to the system action a when the user goal is in ψ. We adopted the HIS-POMDP user utterance model, consisting of a bigram model and an item model. The bigram model predicts the probability of the user utterance given the system action in terms of their types. Specifically, $\Pr(u|a) = \Pr(\mathcal{T}(u)|\mathcal{T}(a))$ where $\mathcal{T}(u)$ denotes the type[4] of user utterance u and $\mathcal{T}(a)$ denotes the type of system action a. The item model checks the consistency of the user utterance, defined by $\mathcal{M}(u|\psi, a) = 1$ if the user utterance is consistent with the user goal and the system action, and $\mathcal{M}(u|\psi, a) = 0$ otherwise. As an example, the user should reply “yes” to the explicit confirmation asking whether the desired route number is 2 when the user wants to query the bus schedule for route number 2, and vice versa. In summary, we used $\Pr(u|\psi, a) = \Pr(\mathcal{T}(u)|\mathcal{T}(a))\mathcal{M}(u|\psi, a)$ for the user utterance model.

2.3 Belief Refinement Model

Given the SLU result $\tilde{u}_i$ and the system action a, the partition ψ is split into ψ'_i with probability $\Pr(\psi'_i|\psi)$ and $\psi - \psi'_i$ with probability $\Pr(\psi - \psi'_i|\psi)$. The belief refinement model $\Pr(\psi'_i|\psi)$ can be seen as the proportion of the belief that is carried from ψ to ψ'_i. This probability can be defined by the following models:

Empirical model: we count $n(\psi)$ from the training datasets, which is the number of user goals that are consistent with partition ψ. The probability is then modeled as $\Pr(\psi'_i|\psi) = \frac{n(\psi'_i)}{n(\psi)}$ if $n(\psi) > 0$ and $\Pr(\psi'_i|\psi) = 0$ otherwise.

Word-match model: this model extends the empirical model by using the domain knowledge when the SLU result $\tilde{u}_i$ does not appear in the training datasets. In other words, rather than simply setting $\Pr(\psi'_i|\psi) = 0$ when $n(\psi') = 0$, we try to have a better estimate of the probability using external information. In DSTC where the dialog

[4] Among total 13 user utterance types in the dialog corpus, meaningful 5 utterances (inform, affirm, negate, deny, and goback) are used for the dialog state tracker update. 4 system action types (open-request, request, impl-conf, and expl-conf) out of total 28 system actions are used for system action types in our tracker.

domain was bus schedule queries, the bus timetable database was made available to all participants. The database contained the valid bus stop names that can be used to track novel bus stop locations that never appeared in the training dataset. Using the database, we calculated how many words $w \in W$ in the user utterance $\tilde{u}_i$ were included in a bus timetable $\mathcal{D}$. For instance, if the user says $\tilde{u}_i$ = "Pittsburgh airport", then $W = \{$"Pittsburgh", "airport"$\}$. The model is thus defined as $\Pr(\psi'_i|\psi) = \frac{n(\psi'_i)}{n(\psi)}$ if $n(\psi'_i) > 0$ and $\Pr(\psi'_i|\psi) = \frac{\alpha}{|W|}\sum_{w \in W} \delta(w \in \mathcal{D})$ otherwise. δ is the indicator function ($\delta(x) = 1$ if x holds and $\delta(x) = 0$ otherwise) and α is the parameter estimated via cross-validation.

Mixture model: this model uses a different approach to estimating novel user goals not appearing in the training datasets. Since the empirical model will assign a zero probability to novel user goals, this model mixes the empirical model with a uniform probability, defined as $\Pr(\psi'_i|\psi) = \epsilon\frac{1}{n_G} + (1-\epsilon)\frac{n(\psi'_i)}{n(\psi)}$ if $n(\psi'_i) > 0$ and $\Pr(\psi'_i|\psi) = \frac{1}{n_G}$ otherwise. n_G is the number of all possible user goals. Since this number is not known a-priori, it is treated as the parameter of the model and found via cross-validation, together with the mixing parameter $\epsilon \in [0, 1]$.

We also applied the partition recombination method [10] to limit the number of partitions and to update beliefs efficiently.

2.4 Belief Transformation Model

The belief update described above produces the M-best hypotheses of user goals. Specifically, the belief update produces $\boldsymbol{x} = [\langle \tilde{g}_1, b(\tilde{g}_1)\rangle, \ldots, \langle \tilde{g}_M, b(\tilde{g}_M)\rangle]$ in each dialog turn, which consists of M most likely user goal hypotheses $\tilde{g}_i$ and their associated beliefs $b(\tilde{g}_i)$. The last hypothesis $\tilde{g}_M$ is reserved as the null hypothesis $\varnothing$ with the belief $b(\varnothing) = 1 - \sum_{i=1}^{M-1} b(\tilde{g}_i)$, which represents that the user goal is not known up to the current dialog turn.

One of the problems with the belief update is that the null hypothesis often remains as the most probable hypothesis even when the SLU result contains the correct user utterance with a high confidence score. Since an atomic hypothesis has a very small prior probability when there are so many possible user goals, the SLU result with a high confidence score is not enough to beat the null hypothesis under the standard belief update in Eqn. (1). We observed that this problem was prevalent when running the tracker on the training datasets, making the tracker incorrectly report the null hypothesis as most probable in most of the dialogs.

To overcome this problem, we added a post-processing step so that the tracker takes a two-step procedure for producing the final L-best list of user goals: we first perform belief update following Eqn. (1) and the models in the previous section, and then transform each belief $b(h_i)$ to the final confidence score s_i, generating the final L-best output list $\boldsymbol{y} = [\langle h_1, s_1\rangle, \ldots, \langle h_L, s_L\rangle]$ where $h_i \in \{\tilde{g}_1, \ldots, \tilde{g}_{M-1}\}$ for $i = 1, \ldots, L-1$. The last hypothesis h_L is reserved as the null hypothesis $\varnothing$. We experimented with the following four belief transformation models, described below.

Threshold model: this model produces the final output list $\boldsymbol{y} = [\langle h^*, s^* \rangle, \langle \varnothing, 1 - s^* \rangle]$ where $h^* = \text{argmax}_{h \in \{\tilde{g}_1, \ldots, \tilde{g}_{M-1}\}} b(h)$ and

$$s^* = \begin{cases} \theta, & \text{if } b(h^*) > \delta \\ b(h^*), & \text{otherwise.} \end{cases} \tag{2}$$

In other words, this model ensures that the top hypothesis has confidence score θ when a belief of the hypothesis is greater than a threshold δ. We find the best values for the parameters δ and θ via cross-validation. Note that this model produces at most two final hypotheses including the null hypothesis.

Full-list regression model: this model estimates the probability that each hypothesis is correct via casting the task as regression. The model uses two logistic regression functions $F_\varnothing$ and F_h. $F_\varnothing$ predicts the probability of correctness for the null hypothesis $\varnothing$ using the single input feature $\phi_\varnothing = b(\varnothing)$. Likewise, F_h predicts the probability of correctness for non-null hypotheses h_i using the input feature $\phi_i = b(h_i)$. We use the predicted probabilities as scores which are normalized so that they sum to 1. The model generates the final M-best output list $\boldsymbol{y} = [\langle h_1, s_1 \rangle, \ldots, \langle h_{M-1}, s_{M-1} \rangle, \langle \varnothing, s_M \rangle]$ where $h_i = \tilde{g}_i$ and

$$s_i = \begin{cases} \frac{F_\varnothing(\phi_i)}{\sum_{j=1}^{M-1} F_h(\phi_j) + F_\varnothing(\phi_\varnothing)}, & \text{if } i = M \\ \frac{F_h(\phi_i)}{\sum_{j=1}^{M-1} F_h(\phi_j) + F_\varnothing(\phi_\varnothing)}, & \text{otherwise.} \end{cases} \tag{3}$$

Rank regression model: this model works in a similar way as in the full-link regression model, except that it uses a single logistic regression function F_r for both the non-null and null hypotheses, and takes the rank value of the hypotheses as an additional input feature. We learn $F_r(\phi_i)$ with $\phi_i = \{b(h_i), i\}$ and produce the final M-best output list $\boldsymbol{y} = [\langle h_1, s_1 \rangle, \ldots, \langle h_{M-1}, s_{M-1} \rangle, \langle \varnothing, s_M \rangle]$ where $h_i = \tilde{g}_i$ and

$$s_i = \frac{F_r(\phi_i)}{\sum_{j=1}^{M} F_r(\phi_j)}. \tag{4}$$

Null regression model: this model uses a single logistic regression function $F_\varnothing$ in the full-list regression model, and predicts the probability of correctness for the null hypothesis only. The model produces the final M-best output list $\boldsymbol{y} = [\langle h_1, s_1 \rangle, \ldots, \langle h_{M-1}, s_{M-1} \rangle, \langle \varnothing, s_M \rangle]$ where $h_i = \tilde{g}_i$ and

$$s_i = \begin{cases} F_\varnothing(\phi_\varnothing) & \text{if } i = M, \\ (1 - s_M)\frac{b(h_i)}{\sum_{j=1}^{M-1} b(h_j)} & \text{otherwise.} \end{cases} \tag{5}$$

3 Experimental Setup

In the experiments, we used three labeled training datasets (train1a, train2, train3) and three test datasets (test1, test2, test3) used in DSTC. Some features of the datasets are provided in Tbl. 1. There was an additional test dataset (test4), which we decided not to include in the experiments since we found that a significant number of labels were missing or incorrect. We only used the SLU data for observations although the datasets contain rich side information such as the automatic speech recognition (ASR) data.

Table 1. Some features of the DSTC datasets used for the tracker

Dataset	Calls	Similarity	Slot
train1a	1013	Similar to train2	9
train2	678	Similar to train1a	9
train3	779	Distinct from train1a and 2	5
test1	765	Very similar to train1a and 2	9
test2	983	Similar to train1a and 2	5
test3	1037	Very similar to train3	5
test4	451	Distinct from all training data	9

We conducted the experiments using datasets train1a, train2, and train3 to tune and select the best models. These training datasets were used separately for tuning the model parameters and evaluating the performance. We performed 10-fold cross-validation for all the experiments regarding the choice of the models.

Datasets train1a and train2 contained both the 1-best and the N-best SLU data. It would be ideal to train the models using the N-best data instead of 1-best data, but we were not able to achieve a significant performance improvement using the N-best data. We suspect that this is due to how the N-best data were collected: The DSTC organizer generated them by executing SLU on the recordings for the challenge, rather than the N-best data were generated when the dialogs were collected. We thus decided to use 1-best data for datasets train1a and train2.

We measured the tracker performance according to the following evaluation metrics used in DSTC[5]: **accuracy (acc)** measures the rate of the most likely hypothesis h_1 being correct, **average score (avgp)** measures the average of scores assigned to the correct hypotheses, **L2 norm** measures the Euclidean distance between the vector of scores from the tracker and the binary vector with 1 in the position of the correct hypotheses, and 0 elsewhere, **mean reciprocal rank (mrr)** measures the average of $1/R$, where R is the minimum rank of the correct hypothesis, **ROC equal error rate (eer)** is the sum of false accept (FA) and false reject (FR) rates when FA rate=FR rate, and **ROC.$\{$v1,v2$\}$.**P measures correct accept (CA) rate when there are at most $P\%$ false accept (FA) rate.

There are two types of ROC measured in DSTC depending on how CA and FA rates are calculated. Let $N(\text{FA})$, $N(\text{CR})$, $N(\text{CA})$ and $N(\text{FR})$ be the number of false accepts (FA), correct rejects (CR), correct accepts (CA), and false rejects (FR). Additionally, let N_D be the total number of data instances. The evaluation takes the most likely hypothesis h_1 and its score s_1 and compares them with the threshold θ and the ground truth user goal h^*. Each evaluation increments appropriate counters by

$$\begin{cases} N(\text{CA})\text{++} & \text{if } s_1 \geq \theta \text{ and } h_1 = h^* \\ N(\text{CR})\text{++} & \text{if } s_1 < \theta \text{ and } h_1 \neq h^* \\ N(\text{FA})\text{++} & \text{if } s_1 \geq \theta \text{ and } h_1 \neq h^* \\ N(\text{FR})\text{++} & \text{if } s_1 < \theta \text{ and } h_1 = h^* \end{cases}$$

[5] http://research.microsoft.com/apps/pubs/?id=169024

The CA rate in ROC.v1 is defined as $\frac{N(\mathrm{CA})}{N_D}$ while in ROC.v2, it is defined as $\frac{N(\mathrm{CA})}{N(\mathrm{CA})+N(\mathrm{FR})}$. Likewise the FA rate in ROC.v1 is defined as $\frac{N(\mathrm{FA})}{N_D}$ while in ROC.v2, it is defined as $\frac{N(\mathrm{FA})}{N(\mathrm{FA})+N(\mathrm{CR})}$. Although ROC.v2 uses the traditional definition, it raises concerns when evaluating the trackers. In the traditional sense, the denominators do not change depending on trackers (more generally, algorithms under evaluation) since they are the numbers of ground-truth positive and negative instances in the data. However, because of the way positive and negative instances are determined ($h_1 = h^*$ or $h_1 \neq h^*$), the denominators are dependent on the tracker. Hence, ROC.v1 is also measured as an alternative. In fact, there was a significant degree of discrepancy between the two versions of ROC, as we will show in the next section.

The trackers in DSTC are evaluated according to three different schedules. In **schedule1**, the metrics were evaluated in every dialog turn, whereas in **schedule2**, the dialog turns in which the user goal appears in the SLU result or in the system action were first identified and then the metrics were evaluated for the subsequent dialog turns. Finally, in **schedule3**, the metrics were evaluated only at the end of each dialog.

4 Results and Analyses

In this section, we present experimental results and provide analyses on models used for our tracker. Specifically, we compare the performances of the models for the observation, the belief refinement, and the belief transformation in the tracker via cross-validation on the training datasets. Using the best evaluated models for our tracker, we compare its performance to those of others participated in DSTC on test datasets. Since there are multiple slots to track in the dialog domain, we report the average performance over the "marginal" slots including the "joint" slot that assigns the values to all slots.

4.1 Observation Model

Tbl. 2 shows the cross-validation results of the three observation models on the training set under schedule1[6]. For the histogram and generative models, we discretized the confidence score values into 10 bins of width 0.1. In train1a and train2, no model had a clear advantage to others, whereas in train3, the confidence score model outperformed others by noticeable margins throughout almost all the metrics.

To gain a better understanding of the results, we generated an x-y plot of the most likely hypothesis being correct (accuracy) versus its SLU confidence score, shown in Fig. 1. The dataset train3 has a clear positive trend in the accuracy while the datasets train1a and train2 do not. Hence, the confidence score model, which simply takes the confidence score as the predicted probability of the SLU result being correct, was not able to perform well on the datasets train1 and train2. On the other hand, the histogram and the generative models should have performed better than confidence score model across the datasets since they can potentially represent a complex mapping between the confidence scores and accuracies. In other words, these two models are expected to perform at least as well as the confidence score model in train3, but they didn't in

[6] The comparative performance results were similar for schedule2 and schedule3.

Table 2. Evaluation of observation models (*Conf*: the confidence score model, *Hist*: the histogram model, *Gen*: the generative model). The results are obtained using the mixture model for belief refinement and the full-list regression model for belief transformation. The bold face denotes top scores for each metric.

	Train1a			Train2			Train3		
	Conf	Hist	Gen	Conf	Hist	Gen	Conf	Hist	Gen
accuracy	0.811	0.819	**0.822**	0.843	**0.855**	0.854	**0.902**	0.886	0.880
avgp	0.775	0.781	**0.781**	0.813	**0.819**	0.819	**0.808**	0.786	0.768
l2	0.313	0.305	**0.304**	0.263	**0.253**	0.254	**0.246**	0.273	0.296
mrr	0.869	0.874	**0.876**	0.887	**0.894**	0.894	**0.938**	0.927	0.921
roc.v1.05	0.692	**0.704**	0.704	0.728	0.736	**0.738**	**0.820**	0.795	0.790
roc.v1.10	0.742	0.750	**0.753**	0.783	0.800	**0.800**	**0.869**	0.847	0.832
roc.v1.20	0.781	0.791	**0.794**	0.828	0.839	**0.839**	**0.892**	0.866	0.855
roc.v1.eer	0.142	**0.138**	0.139	**0.124**	0.128	0.125	**0.097**	0.109	0.116
roc.v2.05	**0.343**	0.341	0.336	**0.237**	0.150	0.227	0.521	**0.536**	0.524
roc.v2.10	**0.542**	0.461	0.456	**0.330**	0.259	0.253	**0.708**	0.674	0.701
roc.v2.20	**0.700**	0.699	0.693	**0.431**	0.415	0.414	**0.834**	0.784	0.802

the experiments. We suspect that this is due to the naive binning strategy we used to model the probability distribution. Investigation of a more effective density estimation technique is left as a future work.

4.2 Belief Refinement Model

Tbl. 3 summarizes the results of the belief refinement models evaluated under schedule1[7]. We can first notice that the empirical model performed worst across the evaluation metrics. This is a natural result since the empirical model assigns a probability of 0 on novel user goals that do not appear in the training datasets. On the other hand, the word-match and the mixture models can generalize to novel user goals, but overall, the mixture model outperformed the word-match model. This is an unfortunate result given that the word-match model tries to leverage the domain knowledge to handle novel user goals, whereas the mixture model simply treats them using the uniform distribution. This implies that, unless the domain knowledge is used properly, simply taking the mixture with the uniform distribution yields a sufficient level of performance.

4.3 Belief Transformation Model

In this section, we compare the performances of the belief transformation models. As a reference, we also report the performance of pure belief update without any transformation. All the results are gathered under the evaluation schedule1[8]. Tbl. 4 summarizes the results. Without any belief transformation (None), the null hypothesis was often the most probable, hence it generally performed worse than other models. This highlights

[7] Again, the comparative performance results were similar for schedule2 and schedule3.

[8] Once again, the comparative performance results were similar for schedule2 and schedule3.

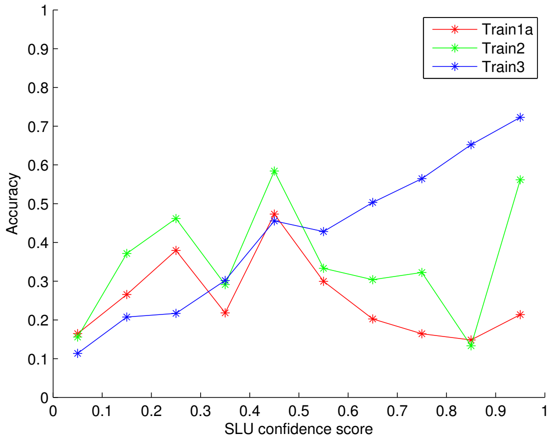

Fig. 1. SLU accuracy for training datasets

the limitation of the purely probabilistic approach to belief update - the Bayes update equation may be too rigid to compensate for the errors incurred by approximations and assumptions made in the observation and belief refinement models.

The threshold model was able to outperform the pure belief update in the accuracy since it successfully suppresses the null hypothesis whenever the most probable hypothesis has a score higher than a fixed threshold. It was also particularly effective on train1a and train2 for the metrics avgp and l2, but less so on train3. The result on train3 can be explained by recalling the fact that the threshold model returns at most two hypotheses and the observation probability graph in Fig. 1 - the pure belief update returns the full list of hypotheses, and thus even if the most probable hypothesis is incorrect, it is highly likely that the true user goal is somewhere down the list. In addition, the SLU confidence score is strongly correlated to the accuracy (*i.e.*, observation probability) in train3, hence the score of the correct hypothesis should be set to a meaningful value.

In general, the full-list and the rank regression models performed significantly better than other models. This is a naturally expected result since they use regression to convert the beliefs to final confidence scores, as an attempt to compensate for the errors incurred by approximations and assumptions made in the observation and belief refinement models.

4.4 DSTC Result

Each team participating in DSTC was allowed to submit up to 5 different trackers for the final results. A total of 9 teams participated in DSTC, submitting a total of 27 trackers. We submitted 5 trackers to take the full advantage. The trackers differ only in the belief transformation while they all use the confidence model for the observation and the mixture model for the belief refinement (see Tbl. 5). The difference between tracker1 and tracker2, also between tracker3 and tracker4, is in how the user goal labels are used

Table 3. Evaluation of belief refinement models (*Emp*: the empirical model, *Word*: the word-match model, *Mix*: the mixture model). The results are obtained using the confidence model for observations and the full-list regression model for belief transformation. The bold face denotes top scores for each metric.

	Train1a			Train2			Train3		
	Emp	Word	Mix	Emp	Word	Mix	Emp	Word	Mix
accuracy	0.754	0.773	**0.811**	0.795	0.843	**0.843**	0.712	0.884	**0.902**
avgp	0.753	0.756	**0.775**	0.781	0.804	**0.813**	0.682	0.798	**0.808**
l2	0.345	0.339	**0.313**	0.309	0.274	**0.263**	0.420	0.257	**0.246**
mrr	0.835	0.847	**0.869**	0.856	0.886	**0.887**	0.823	0.928	**0.938**
roc.v1.05	0.655	0.681	**0.692**	0.635	0.680	**0.728**	0.582	0.784	**0.820**
roc.v1.10	0.691	0.714	**0.742**	0.726	0.778	**0.783**	0.650	0.834	**0.869**
roc.v1.20	0.727	0.744	**0.781**	0.773	0.824	**0.828**	0.679	0.864	**0.892**
roc.v1.eer	0.219	**0.132**	0.142	0.128	0.131	**0.124**	0.131	0.112	**0.097**
roc.v2.05	0.343	0.242	**0.343**	**0.303**	0.236	0.237	**0.606**	0.509	0.521
roc.v2.10	0.471	0.382	**0.542**	**0.423**	0.259	0.330	0.635	0.673	**0.708**
roc.v2.20	**0.719**	0.604	0.700	**0.564**	0.367	0.431	0.721	0.774	**0.834**

for training: tracker1 and tracker3 use true user goals appearing at the end of dialogs, mostly for programming convenience since it requires reading the true user goal label only once for each dialog. On the other hand, tracker2 and tracker4 use true user goal labels in each dialog turn.

In order to compare our tracker with others participated in DSTC, we chose tracker4 as the most effective one among our 5 submitted trackers since it achieved the top scores in the largest number of evaluation metrics. In the same way, we selected tracker2 for team3, tracker3 for team6, tracker3 for team8, and tracker1 for the rest of the teams. The results of each team are presented in Tbl. 6. The baseline tracker is included as a reference, which simply outputs the hypothesis with the largest SLU confidence score in the N-best list. Our tracker (team9) achieved 15%, 29% and 8% improvements in accuracy compared to the baseline in datasets test1, test2, and test3, respectively. Our tracker also significantly improved the performance in evaluation metrics such as average score (avgp), mean reciprocal rank (mrr) and ROC.v1.

Compared to other teams, our tracker showed strong performance in accuracy, avgp, l2 and mrr. Recalling the definition of the metrics, a strong performance in accuracy implies that the tracker mostly selects the true user goal as the most probable hypothesis. A strong performance in avgp (higher is better) and l2 norm (lower is better) implies that the tracker assigns high confidence scores to the correct hypotheses, regardless of its position in the list. A strong performance in mrr (higher is better) implies that the tracker generally puts the correct hypotheses in the higher positions in the hypotheses lists.

Our tracker also showed a competitive level of performance in ROC.v1. On the other hand, ROC.v2 showed quite an independent trend compared to other metrics. Intrigued by the phenomenon, we gave a further thought on the ROC.v2 metric: reformulating the CA and FA rates in terms of conditional probabilities, we obtain $\Pr(s_1 \geq \theta | h_1 =$

Table 4. Evaluation of belief transformation models (*None*: the vanilla belief update without belief transformation, *Thre*: the threshold model, *Full*: the full-list regression model, *Rank*: the rank regression model, *Null*: the null regression model). The results are obtained using the confidence model for observations and the mixture model for belief refinement. Top scores for each metric are represented by the bold face.

	Train1a					Train2					Train3				
	None	Thre	Full	Rank	Null	None	Thre	Full	Rank	Null	None	Thre	Full	Rank	Null
accuracy	0.779	0.806	**0.811**	0.810	0.773	0.787	0.832	0.843	**0.847**	0.781	0.876	0.892	**0.902**	0.897	0.847
avgp	0.775	**0.799**	0.775	0.770	0.773	0.786	**0.821**	0.813	0.813	0.782	**0.850**	0.846	0.808	0.779	0.809
l2	0.317	**0.284**	0.313	0.318	0.321	0.302	**0.254**	0.263	0.262	0.308	**0.205**	0.217	0.246	0.280	0.265
mrr	0.850	0.844	**0.869**	0.868	0.847	0.854	0.862	0.887	**0.889**	0.851	0.925	0.907	**0.938**	0.924	0.908
roc.v1.05	0.656	0.664	0.692	**0.694**	0.597	0.651	0.648	**0.728**	0.719	0.647	0.777	0.448	**0.820**	0.803	0.684
roc.v1.10	0.705	0.705	0.742	**0.746**	0.701	0.700	0.689	0.783	**0.793**	0.670	0.827	0.684	**0.869**	0.864	0.762
roc.v1.20	0.737	0.714	0.781	**0.784**	0.733	0.750	0.739	0.828	**0.831**	0.741	0.858	0.793	**0.892**	0.890	0.823
roc.v1.eer	0.240	0.176	0.142	**0.138**	0.410	0.143	0.206	0.124	**0.123**	0.435	0.119	0.491	0.097	**0.094**	0.147
roc.v2.05	0.185	0.222	**0.343**	0.336	0.000	0.101	0.203	0.237	**0.238**	0.083	**0.556**	0.416	0.521	0.482	0.498
roc.v2.10	0.381	0.407	**0.542**	0.523	0.190	0.175	0.218	**0.330**	0.325	0.122	0.607	0.419	**0.708**	0.564	0.620
roc.v2.20	0.618	0.636	0.700	**0.705**	0.318	0.477	0.301	0.431	**0.493**	0.381	0.817	0.431	**0.834**	0.754	0.737

Table 5. Entries submitted to DSTC

Tracker	Belief Transformation model
tracker1, tracker2	Full-list regression model
tracker3, tracker4	Rank regression model
tracker5	Null regression model

$h^*)$ for the CA rate and $\Pr(s_1 \geq \theta | h_1 \neq h^*)$ for the FA rate. Suppose an imaginary tracker with a very high accuracy $\Pr(h_1 = h^*) = 0.99$. This tracker misses for only 0.01 of the dialogs. However, if the tracker assigns large scores on those incorrectly tracked hypotheses, it will show very bad performance in ROC.v2 regardless of the high accuracy. Hence, it can be seen as measuring how well the scores are assigned, independent from (or normalized by) the accuracy of the tracker. Since we desire an incorrect tracking result to be assigned with a low confidence score, we believe that ROC.v2 is a useful metric to complement others.

As a final note, the models used in our tracker were prepared for marginal slots only. For the confidence score of the joint slot, the tracker simply used the multiplication of confidence scores from marginal slots. We expect that we can further improve the performance by preparing and training models for the joint slot as well, which should have been done before submitting our tracker.

5 Conclusion

In this paper, we described our experience with engineering a statistical dialog state tracker while participating in DSTC. Our engineering effort was focused on improving

Table 6. Results of the trackers participated in DSTC according to schedule1. The bold face denotes top 3 scores in each evaluation metric. T9 is our tracker.

	Base	T1	T2	T3	T4	T5	T6	T7	T8	T9
Test 1										
accuracy	0.712	**0.832**	0.807	0.808	0.737	0.795	**0.867**	0.783	0.508	**0.822**
avgp	0.733	0.774	0.771	**0.807**	0.737	0.787	**0.823**	0.762	0.494	**0.794**
l2	0.377	0.319	0.322	**0.273**	0.372	0.300	**0.246**	0.335	0.715	**0.290**
mrr	0.797	**0.875**	0.858	0.846	0.813	0.852	**0.900**	0.843	0.593	**0.878**
roc.v1.05	0.622	**0.723**	0.672	0.601	0.196	0.710	**0.763**	0.650	0.202	**0.720**
roc.v1.10	0.634	**0.781**	0.747	**0.768**	0.290	0.751	**0.820**	0.704	0.329	0.756
roc.v1.20	0.665	**0.818**	0.790	0.789	0.534	0.777	**0.851**	0.764	0.349	**0.790**
roc.v1.eer	0.244	**0.126**	0.246	0.243	0.737	**0.122**	**0.118**	0.147	0.521	0.143
roc.v2.05	**0.487**	**0.642**	0.010	0.021	0.000	**0.548**	0.162	0.193	0.039	0.263
roc.v2.10	**0.689**	**0.715**	0.144	0.025	0.000	**0.677**	0.392	0.347	0.050	0.473
roc.v2.20	**0.707**	**0.803**	0.478	0.291	0.000	**0.735**	0.594	0.578	0.275	0.621
Test 2										
accuracy	0.546	0.646	**0.707**	0.683	0.635	0.622	**0.790**	0.652	0.344	**0.705**
avgp	0.573	0.550	0.629	**0.684**	0.634	0.615	**0.714**	0.649	0.290	**0.651**
l2	0.603	0.633	0.503	**0.446**	0.517	0.535	**0.386**	0.492	0.997	**0.476**
mrr	0.650	0.717	**0.792**	0.756	0.713	0.722	**0.843**	0.744	0.465	**0.797**
roc.v1.05	0.431	0.487	**0.516**	0.452	0.164	0.480	**0.660**	0.479	0.042	**0.490**
roc.v1.10	0.446	0.544	**0.568**	**0.632**	0.164	0.506	**0.710**	0.538	0.106	0.565
roc.v1.20	0.478	0.589	**0.637**	**0.642**	0.273	0.539	**0.761**	0.601	0.260	0.630
roc.v1.eer	0.192	0.197	0.394	**0.144**	0.635	0.212	**0.159**	**0.189**	0.358	0.219
roc.v2.05	**0.430**	**0.523**	0.244	0.269	0.000	0.401	**0.458**	0.409	0.045	0.376
roc.v2.10	0.472	**0.596**	0.398	0.370	0.000	**0.620**	**0.526**	0.469	0.171	0.412
roc.v2.20	0.499	**0.695**	0.479	0.558	0.000	**0.698**	**0.620**	0.553	0.440	0.473
Test 3										
accuracy	0.789	0.793	**0.843**	0.819	0.819	0.779	**0.835**	0.790	0.787	**0.847**
avgp	0.751	0.725	0.757	**0.787**	**0.784**	0.701	0.752	0.755	**0.764**	0.740
l2	0.352	0.369	0.323	**0.291**	**0.295**	0.395	0.334	0.337	**0.315**	0.343
mrr	0.835	0.851	**0.883**	0.853	0.853	0.828	**0.890**	0.841	0.804	**0.887**
roc.v1.05	0.565	0.647	0.681	**0.724**	**0.702**	0.623	0.687	0.701	0.330	**0.738**
roc.v1.10	0.660	0.704	**0.765**	**0.769**	0.758	0.686	0.757	0.740	0.468	**0.780**
roc.v1.20	0.743	0.761	**0.817**	0.803	0.802	0.738	**0.809**	0.768	0.606	**0.816**
roc.v1.eer	0.189	0.164	0.154	0.273	**0.124**	0.171	0.148	**0.119**	0.344	**0.129**
roc.v2.05	0.559	**0.624**	0.343	0.279	0.208	**0.624**	**0.609**	0.136	0.000	0.561
roc.v2.10	0.594	**0.705**	0.478	0.366	0.521	0.657	**0.664**	0.420	0.000	**0.667**
roc.v2.20	0.656	0.781	0.726	0.521	**0.821**	0.710	0.776	**0.866**	0.000	**0.788**

three important models in the tracker: the observation, the belief refinement, and the belief transformation models. Using standard statistical techniques, we were able to produce a tracker that performed competitively among the participants.

As for the future work, we plan to refine the user utterance model for improving the performance of the tracker since there are a number of user utterances that are not handled by the current model. We also plan to re-evaluate our tracker with properly handling the joint slot, since the current tracker constructs models independently for each marginal slot and then combines the results by simply multiplying the predicted scores.

Acknowledgement

This work was supported by NRF of Korea (Grant# 2012-007881), and MKE/KEIT of Korea (IT R&D Program Contract# 10041678)

References

1. Bohus, D., Rudnicky, A.: A ”k hypotheses + other” belief updating model. In: Proceedings of the AAAI Workshop on Statistical and Empirical Approaches for Spoken Dialogue Systems. (2006)
2. Mehta, N., Gupta, R., Raux, A., Ramachandran, D., Krawczyk, S.: Probabilistic ontology trees for belief tracking in dialog systems. In: Proceedings of the 11th Annual Meeting of the Special Interest Group on Discourse and Dialogue (SIGDIAL). (2010) 37–46
3. Roy, N., Pineau, J., Thrun, S.: Spoken dialogue management using probabilistic reasoning. In: Proceedings of the 38th Annual Meeting on Association for Computational Linguistics (ACL). (2000) 93–100
4. Williams, J.D., Young, S.: Partially observable Markov decision processes for spoken dialog systems. Computer Speech and Language **21**(2) (2007) 393–422
5. Thomson, B., Young, S.: Bayesian update of dialogue state: A POMDP framework for spoken dialogue systems. Computer Speech and Language **24**(4) (2010) 562–588
6. Kim, D., Kim, J.H., Kim, K.E.: Robust performance evaluation of POMDP-based dialogue systems. IEEE Transactions on Audio, Speech, and Language Processing **19**(4) (2011) 1029–1040
7. Kaelbling, L.P., Littman, M.L., Cassandra, A.R.: Planning and acting in partially observable stochastic domains. Artificial Intelligence **101**(1–2) (1998) 99–134
8. Williams, J., Raux, A., Ramachandran, D., Black, A.: The dialog state tracking challenge. In: Proceedings of the 14th Annual Meeting of the Special Interest Group on Discourse and Dialogue (SIGDIAL). (2013)
9. Young, S., Gašić, M., Keizer, S., Mairesse, F., Schatzmann, J., Thomson, B., Yu, K.: The hidden information state model: A practical framework for POMDP-based spoken dialogue management. Computer Speech and Language **24**(2) (2010) 150–174
10. Williams, J.D.: Incremental partition recombination for efficient tracking of multiple dialog states. In: Proceedings of the IEEE International Conference on Acoustics Speech and Signal Processing (ICASSP). (2010) 5382–5385
11. Williams, J.D.: Exploiting the ASR N-best by tracking multiple dialog state hypotheses. In: Proceedings of the 9th Annual Conference of the International Speech Communication Association (INTERSPEECH). (2008) 191–194