Recipe For Building Robust Spoken Dialog State Trackers: Dialog State Tracking Challenge System Description

Sungjin Lee
Language Technologies Institute,
Carnegie Mellon University,
Pittsburgh, Pennsylvania, USA
sungjin.lee@cs.cmu.edu

Maxine Eskenazi
Language Technologies Institute,
Carnegie Mellon University,
Pittsburgh, Pennsylvania, USA
max@cs.cmu.edu

Abstract

For robust spoken conversational interaction, many dialog state tracking algorithms have been developed. Few studies, however, have reported the strengths and weaknesses of each method. The *Dialog State Tracking Challenge* (DSTC) is designed to address this issue by comparing various methods on the same domain. In this paper, we present a set of techniques that build a robust dialog state tracker with high performance: wide-coverage and well-calibrated data selection, feature-rich discriminative model design, generalization improvement techniques and unsupervised prior adaptation. The DSTC results show that the proposed method is superior to other systems on average on both the development and test datasets.

1 Introduction

Even though we have recently seen an explosive growth of interest in speech-enabled applications, there are still many problems to overcome in order to provide users with practical and profitable services. One of the long-standing problems which may often frustrate users is *Automatic Speech Recognition* (ASR) error. Due to ASR error, it is barely possible to directly observe what the user said and finally figure out the true user goal. The aim of dialog state tracking is, therefore, to accurately estimate the true dialog state from erroneous observations as a dialog unfolds.

In order to achieve this goal, many dialog state tracking algorithms have been developed. Few studies, however, have reported the strengths and weaknesses of each method. The *Dialog State Tracking Challenge*[1] (DSTC) was organized to advance state-of-the-art technologies for dialog state tracking by allowing for reliable comparisons between different approaches using the same datasets. Unlike other machine learning-based empirical tasks, DSTC is also carefully designed to take into consideration diverse realistic mismatches. For instance, there are test datasets that were collected by systems using different speech recognizers, spoken language understanding (SLU) modules, and dialog managers. Also there are test datasets that were produced by similar systems but deployed at a different time (1 year later) with extended coverage. Since such mismatches between training and test data may often happen in real deployment, it is important to build a tracker which constantly shows high performance across all test datasets despite various mismatches.

The aim of this paper is to describe a set of techniques used to build a robust tracker with high performance: wide-coverage and well-calibrated data selection, feature-rich discriminative model design, generalization improvement techniques and unsupervised prior adaptation. Our challenge systems are basically various combinations of those techniques. The DSTC results demonstrate the effectiveness of each technique.

This paper is structured as follows. Section 2 describes the challenge setup. Section 3 elaborates on our proposed approaches. Section 4 briefly describes previous research and other systems that participated in DSTC. Section 5 presents and discusses the results. Finally, Section 6 concludes with a brief summary and suggestions for future research.

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

Proceedings of the SIGDIAL 2013 Conference, pages 414–422,
Metz, France, 22-24 August 2013.

2 Dialog State Tracking Challenge

This section describes the task for DSTC and datasets provided for training and test. Most part of this section is borrowed from the DSTC manual[2].

2.1 Task Description

DSTC data is taken from several different spoken dialog systems which all provided bus schedule information for Pittsburgh, Pennsylvania, USA as part of the *Spoken Dialog Challenge* (Black et al., 2011). There are 9 slots which are evaluated: *route*, *from.desc*, *from.neighborhood*, *from.monument*, *to.desc*, *to.neighborhood*, *to.monument*, *date*, and *time*. Since both marginal and joint representations of dialog states are important for deciding dialog actions, the challenge takes into consideration both. Each joint representation is an assignment of values to all slots. Thus there are 9 marginal outputs and 1 joint output in total, which are all evaluated separately.

The dialog tracker receives SLU N-best hypotheses for each user turn, each with a confidence score. In general, there are a large number of values for each slot, and the coverage of N-best hypotheses is good, thus the challenge confines consideration of goals to slots and values that have been observed in an SLU output. By exploiting this aspect, the task of a dialog state tracker is to generate a set of observed slot and value pairs, with a score between 0 and 1. The sum of all scores is restricted to sum to 1.0. Thus 1.0 – total score is defined as the score of a special value *None* that indicates the user's goal has not yet been appeared on any SLU output.

2.2 Datasets

The data is divided into 2 training sets and 4 test sets (Table 1). For standardized development sets, each training set is split in half. Participants were asked to report results on the second half of each set. The data from group A in train2, and test1 was collected using essentially the same dialog system. Only a few updates were made to reflect changes to the bus schedule. The data in test2 was collected using a different version of group A's dialog manager. The data from group B in train3 and test3 were collected using essentially the same dialog system; the main difference is that test3 covers more bus routes. Test4 tests the condition when training and testing using totally different dialog systems, and when there is no same-system training data available.

Dataset	Source	Calls	Time period
train2	Group A	678	Summer 2010
train3	Group B	779	Summer 2010
test1	Group A	765	Winter 2011-12
test2	Group A	983	Winter 2011-12
test3	Group B	1037	Winter 2011-12
test4	Group C	451	Summer 2010

Table 1: Dataset description.

2.3 Metrics

There are a variety of aspects of tracker performance that were measured: accuracy, mean reciprocal rank (MRR), ROC curves, Average score [3], and Brier score [4]. There are three schedules for determining which turns to include in each evaluation.

- Schedule 1: Include all turns.
- Schedule 2: Include a turn for a given concept only if that concept either appears on the SLU N-Best list in that turn, or if the system's action references that concept in that turn.
- Schedule 3: Include only the turn before the system starts over from the beginning, and the last turn of the dialog.

3 Recipe for Building a Robust Tracker

In this section, we present several ingredients for building a robust state tracker that come into play at various levels of the development process: from data selection to model adaptation.

3.1 Wide-Coverage and Well-Calibrated Data Selection

The first step to create a robust dialog state tracker is the use of data which covers diverse system dialog actions and user inputs with well-calibrated confidence scores. Since dialog policies can be varying according to how a dialog proceeds, it is crucial to arrange a training dialog corpus with well-balanced dialog actions. For example, group A datasets barely have implicit confirmation and heavily rely on explicit confirmation, while group B datasets have both types of confirmation. Thus a model trained on group A datasets cannot exploit implicit

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

[3] the average score assigned to the correct item

[4] the L2 norm between the vector of scores output by dialog state tracker and a vector with 1 in the position of the correct item, and 0 elsewhere

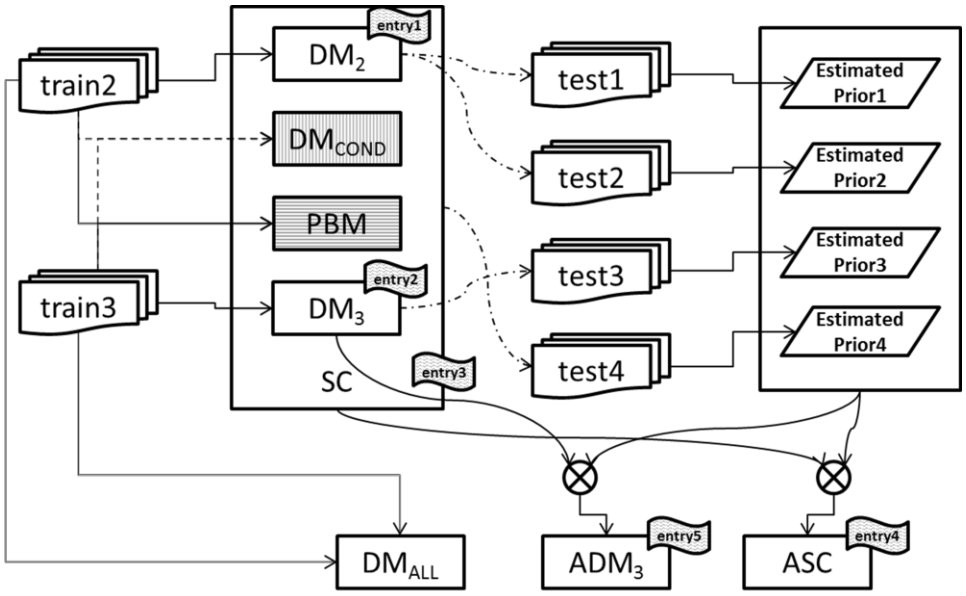


Figure 1: Diagram showing the relation between datasets and models. Each team could have up to five systems entered. Our challenge entries are tagged by their entry numbers. More detailed descriptions about each model are provided in Section 3.

confirmation when applied to group B datasets, whereas a model trained on group B datasets can be applied to group A datasets without much loss.

Another important aspect of the data is how well user inputs are calibrated. If the confidence score is well-calibrated, confirmation can be skipped in the case of a hypothesis with a high confidence. On the contrary, if the quality of the confidence score is very poor, a successful dialog will only be possible via heavy use of confirmation. Thus a model trained on a well-calibrated dataset is likely to perform well on the poorly-calibrated dataset because of backup confirmation. Whereas, a model trained on the poorly-calibrated dataset will not perform well on the well-calibrated dataset due to the mismatch of the confidence score as well as the scarceness of confirmation information. The group A datasets have been shown to be poorly calibrated (Lee and Eskenazi, 2012); this is also shown in Fig. 2. Group B datasets are relatively well-calibrated, however.

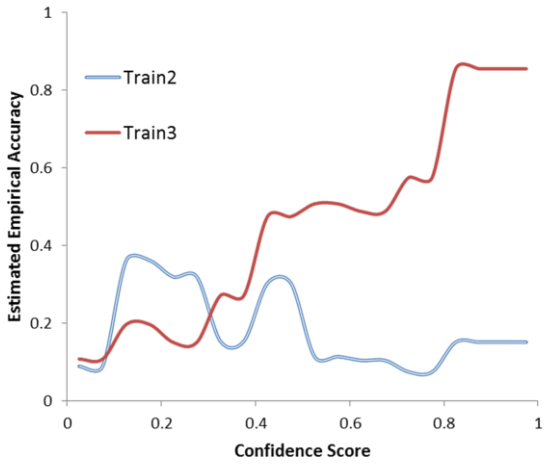


Figure 2: Estimated empirical accuracy of confidence score for *from* slot. Ideally calibrated confidence score should be directly proportional to empirical accuracy.

The importance of wide coverage and well-calibrated data can be observed by examining the results of entry1 and entry2 (Fig. 1) which are trained on group A and B datasets, respectively.

3.2 Feature-Rich Discriminative Model Design

Most previous approaches are based on generative temporal modeling where the current dialog state is estimated using a few features such as the current system action and N-best hypotheses with corresponding confidence scores given the estimated dialog state at the previous turn (Gasic and Young, 2011; Lee and Eskenazi, 2012; Raux and Ma, 2011; Thomson and Young, 2010; Williams, 2010; Young et al., 2010). However, several fundamental questions have been raised recently about the formulation of the dialog state update as a generative temporal model: limitation in modeling correlations between observations in different time slices; and the insensitive discrimination between true and false dialog states (Williams, 2012).

In fact, such limitations can be improved by adopting a discriminative approach, which enables the incorporation of a rich set of features without worrying about their interdependence (Sutton and McCallum, 2006). For example, a hypothesis that repeats with low confidence scores is likely to be a manifestation of ASR error correlations between observations in different time slices. Thus, the highest confidence score that a hypothesis has attained so far could be a useful feature in preventing repeated incorrect hypotheses from defeating the correct hypothesis (which had a higher score but was only seen once). Another useful feature could be the distribution of confidence scores that a hypothesis has attained thus far, since it may not have the same effect as having a single observation with the total score due to the potential nonlinearity of confidence scores. There are many other potentially useful features. The entire list of features used for the challenge system is found in Appendix A.

In addition to the role of rich features in performance enhancement, the incorporation of rich features is also important for robust state tracking. If the tracker estimates the true state by considering various aspects of observations and prior knowledge, then the influence of differences in certain factors between datasets can be mitigated by many other factors that are retained relatively unchanged between datasets.

For the challenge system, we employed a *Maximum Entropy* (MaxEnt) model which is one of most powerful undirected graphical models. Unlike previous work using MaxEnt (Bohus and Rudnicky, 2006) where the model is limited to maintain only the top K-best hypotheses, we amended MaxEnt to allow for the entire set of observed hypotheses to be incorporated; Several feature functions which differ only by output labels were aggregated into one common feature function so that they can share common parameters and gather their statistics together (Appendix A). This modification is also crucial for robust estimation of the model parameters since some slots such as *from* and *to* can have about 10^4 values but most of them are not seen in the training corpus.

The effectiveness of feature-rich discriminative modeling can be observed by comparing the results of DM_{ALL} and PBM (Fig. 1) which are discriminative and generative models, respectively.

Note that interesting relational constraints, e.g. whether or not departure and arrival places are valid on a route, can be incorporated by adopting a structured model such as *Conditional Random Field* (CRF). But CRF was not used for the challenge since the bus information that was provided is not compatible with every dataset. The effectiveness of a structured model has been investigated in a separate publication (Lee, 2013).

3.3 Generalization Improvement Techniques

Even though the incorporation of a set of rich features helps overcome the weaknesses of previous approaches, it also implies a risk of overfitting training datasets due to its increased capacity of function class. Overfitting is a serious hazard especially for test datasets that are severely dissimilar to training datasets. As noted above, since the test datasets of the challenge are intentionally arranged to have various mismatches, it is crucial that we prevent a model from overfitting training datasets. In the rest of this section, we describe various ways of controlling the capacity of a model.

The most obvious method to control the capacity is to penalize larger weights proportional to the squared values of the weights or the absolute values of the weights. We employ the *Orthant-wise Limited-memory Quasi Newton* optimizer (Andrew and Gao, 2007) for L1 regularization. The weights for L1 regularization were set to be 10 and 3 for the prior features and the other features, respectively. These values were chosen through cross-validation over several values rather than doing a thorough search.

A second method, which is often convenient, is to start with small weights and then stop the learning before it has time to overfit provided that it finds the true regularities before it finds the spurious regularities that are related to specific training datasets. It could be hard, however, to decide when to stop. A typical technique is to keep learning until the performance on the validation set gets worse and then stop training and go back to the best point. For the challenge systems, we applied a simpler method that is to stop the training if the average objective function change over the course of 10 previous iterations is less than 0.1, which is usually set to a much smaller number such as 10^{-4}.

In general, prediction errors can be decomposed into two main subcomponents, i.e., error due to bias and variance (Hastie et. al, 2009). It is also known that there is a tradeoff between bias and variance. If a model is flexible enough to fit the given data, errors due to bias

will decrease while errors due to variance will increase. The methods stated above try to achieve less error by decreasing errors due to variance. However we cannot avoid increasing errors due to bias in this way. Thus we need a method to alleviate the tradeoff between bias and variance.

System combination is one powerful way to reduce variance without raising bias. If we average models that have different forms and make different mistakes, the average will do better than the individual models. This effect is largest when the models make very different predictions from one another. We could make the models different by simply employing different machine learning algorithms as well as by training them on different subsets of the training data.

The challenge system, entry3, consists of three discriminative models and one generative model (Fig. 1). Entry1 and entry2 were trained on different training datasets to make them produce different predictions. DM_{COND} is a discriminative model trained on both train2 and train3. Also, DM_{COND} differs from other discriminative models in the way that it was trained: the parameters associated with the features which are computable without grounding action information (features (1), (5), (8), (9) and (10) in Appendix A) are trained first and then the other features are learned given the former parameters. The idea behind this training method is to encourage the model to put more weight on dialog policy invariant features. The final component PBM is the *AT&T Statistical Dialog Toolkit* [5] which is one of the state-of-the-art generative model-based systems. We modified it to process implicit confirmation and incorporate the prior distribution which was estimated on the training corpus. The prior distribution was smoothed by an approximate *Good-Turing* estimation on the fly when the system encounters an unseen value at run time. The improvement from system combination is verified by the results of entry3.

3.4 Unsupervised Prior Adaptation

While a prior is a highly effective type of information for dialog state tracking, it is also able to hamper the performance when incorrectly estimated. Thus it is worthwhile to investigate adapting the prior to the test datasets. Since a dialog state tracker is meant to estimate the posterior probabilities over hypotheses, we can extract estimated labels from test datasets by setting an appropriate threshold, taking the hypotheses with a greater probability than the threshold as labels. By combining the predictive prior from test datasets and the prior from training datasets, we adapted entry2 and entry3 in an unsupervised way to produce entry5 and entry4, respectively (Fig. 1). For each test dataset, we used different thresholds: 0.95 for test1, test2 and test3, and 0.85 for test4.

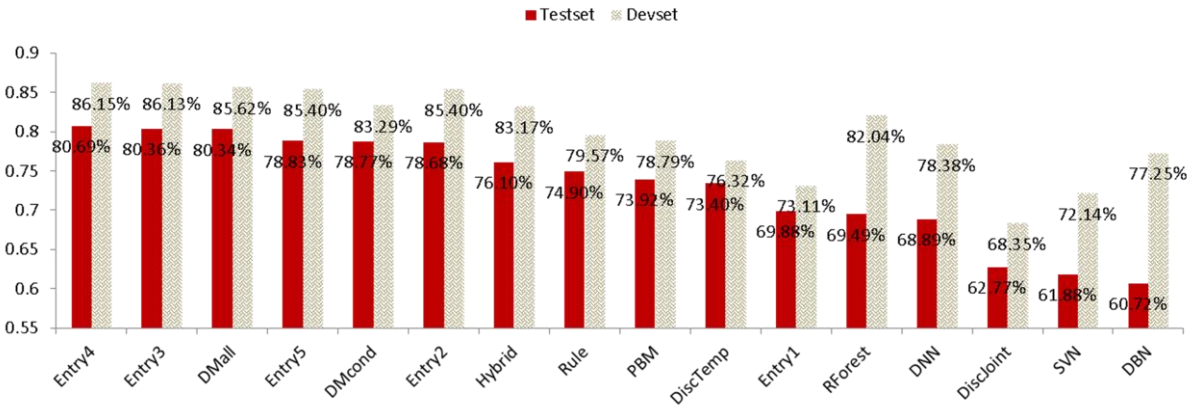
4 Related Work

Since the *Partially Observable Markov Decision Process* (POMDP) framework has offered a well-founded theory for both state tracking and decision making, most earlier studies adopted generative temporal models, the typical way to formulate belief state updates for POMDP-based systems (Williams and Young, 2007). Several approximate methods have also emerged to tackle the vast complexity of representing and maintaining belief states, e.g., partition-based approaches (Gasic and Young, 2011; Lee and Eskenazi, 2012; Williams, 2010; Young et al., 2010) and Bayesian network (BN)-based methods (Raux and Ma, 2011; Thomson and Young, 2010). A drawback of the previous generative models is that it is hard to incorporate a rich set of observation features, which are often partly dependent on one another. Moreover, the quality of the confidence score will be critical to all generative models proposed so far, since they do not usually try to handle potential nonlinearity in confidence scores.

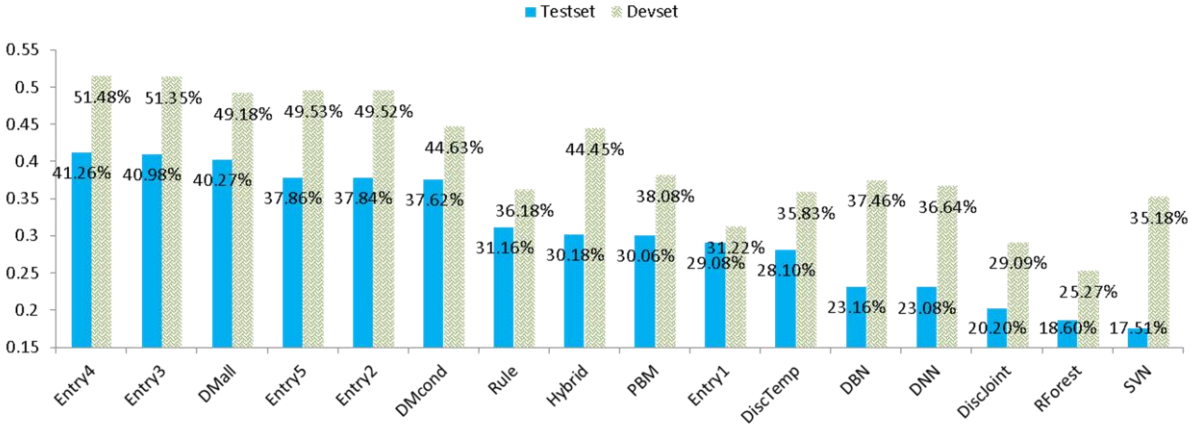
As far as discriminative models are concerned, the MaxEnt model has been applied (Bohus and Rudnicky, 2006). But the model is restricted to maintaining only the top K-best hypotheses, where K is a predefined parameter, resulting in potential degradation of performance and difficulties in extending it to structured models.

Finally, there is a wide range of systems that participated in *Dialog State Tracking Challenge* 2013: from rule-based systems to fairly complex statistical methods such as *Deep Neural Networks*. Since we have not only traditional generative models such as *Dynamic Bayesian Network* and partition-based approaches, but also newly-proposed discriminative approaches such as log-linear models, *Support Vector Machines* and *Deep Neural Networks*, the analysis of the challenge results is expected to reveal valuable lessons and future research directions.

[5] http://www2.research.att.com/sw/tools/asdt/



(a) *All* slot: a weighted average accuracy across all slots



(b) *Joint* slot

Figure 3: Accuracy measured at schedule 3 averaged over the test and development datasets. Models which do not appear in Fig. 1 are the best system of each team except for us. Rule denotes a rule-based system, Hybrid a hybrid system of discriminative and generative approaches, DiscTemp a discriminative temporal model, RForest a random forest model, DNN a deep neural network model, DiscJoint a discriminative model which deals with slots jointly, SVM a support vector machine model, and DBN a dynamic Bayesian network mode.

5 Results and Discussion

The official results of the challenge are publicly available and our team is team6. As mentioned in Section 2.3, there are a variety of aspects of tracker performance that were measured on different schedules. Since prediction accuracy at the end of a dialog directly translates to the success of the entire task, we first show the average accuracy across all test datasets measured at schedule 3 in Fig. 3. The average accuracy at schedule 3 also well represents how robust a state tracker is since the test datasets are widely distributed in the dimensions of dialog policies, dialog length and the quality of user input and confidence score.

First of all, we note that our 4 entries (entries2-5) took the top positions in both the *All* and *Joint* categories. Entry4, which showed the best performance, outperformed the best entry from other teams by 4.59% (entry2 of team9) and 10.1% (entry2 of team2). Specificially, the large improvement in *Joint* implies that our model performs evenly well for all slots and is more robust to the traits of each slot.

Furthermore, from the results we can verify the effectiveness of each technique for achieving robustness. Given the large gap between the performance of entry1 and of entry2, it is clearly shown that a model trained on a wide-coverage and well-calibrated dialog corpus can be applicable to a broad range of test datasets without much loss. Even though entry2 was trained on only 344 dialogs (the first half of train3), it already surpasses most of competing models.

The utility of a feature-rich discriminative model is demonstrated by the fact that DM_{ALL} greatly outperformed PBM. We also note that just using a discriminative model does not

guarantee improved performance since many discriminative systems that participated in the challenge underperformed some of the entries that were based on generative modeling or rules. This result implies that devising effective features is central to performance.

In addition, this result also points to the necessity of controlling the capacity of a model. While our models constantly show good performance both on development sets and test sets, the performance of the other models significantly dropped off. In fact, this explains why Hybrid and Rule systems switch their positions in the *Joint* slot. Moreover, many other systems in the graph tail seem to be severely overfitted, resulting in poor performance on test datasets despite relatively good performance on development datasets. As expected, system combination gives rise to better accuracy without loss of robustness; entry3 clearly outperforms each of its components, i.e. entry1, entry2, DM_{COND} and PBM, on both development and test datasets.

Finally, the improvement observed when using unsupervised prior adaptation is also shown to be positive but its effect size is not significant: entry5 vs. entry2 and entry4 vs. entry3. Given that the way in which we have adapted the model is fairly primitive, we believe that there is much room to refine the unsupervised adaptation method.

MRR measures the average of 1/R, where R is the rank of the first correct hypothesis. MRR at schedule 3 measures the quality of the final ranking which may be most important to a multi-modal interface that can display results to the user. Even though the results are not displayed due to space limitations, the results for MRR are very similar to those for accuracy. Our 4 entries (entries2-5) still take the top positions.

The ROC curves assess the discrimination of the top hypothesis' score. The better discrimination at schedule 2 may be helpful for reducing unnecessary confirmations for values with sufficiently high belief. Also, the better discrimination at schedule 3 may enable a model to adapt to test data in an unsupervised manner by allowing us to set a proper threshold to produce predictive labels. The ROC curves of our systems again showed the highest levels of discrimination.

6 Conclusion

In this paper, we presented a set of techniques to build a robust dialog state tracker without losing performance: wide-coverage and well-calibrated data selection, feature-rich discriminative model design, generalization improvement techniques and unsupervised prior adaptation. The results in terms of various metrics show that the proposed method is truly useful for building a tracker prominently robust not only to mismatches between training and test datasets but also to the traits of different slots. Since we used relatively simple features for this work, there is much room to boost performance through feature engineering. Also, more thorough search for regularization weights can give additional performance gain. Moreover, one can extend the present discriminative model presented here to a structured version which can improve performance further by allowing relational constraints to be incorporated (Lee, 2013). Finally, we believe that once a more detailed and thorough investigation of the challenge results has been carried out, we will be able to take the best of each system and combine them to generate a much better dialog state tracker.

Acknowledgments

This work was funded by NSF grant IIS0914927. The opinions expressed in this paper do not necessarily reflect those of NSF.

References

G. Andrew and J. Gao, 2007. Scalable training of L1-regularized log-linear models. In Proceedings of ICML.

A. Black et al., 2011. Spoken dialog challenge 2010: Comparison of live and control test results. In Proceedings of SIGDIAL.

D. Bohus and A. Rudnicky, 2006. A K hypotheses + other belief updating model. In Proceedings of AAAI Workshop on Statistical and Empirical Approaches for Spoken Dialogue Systems.

M. Gasic and S. Young, 2011. Effective handling of dialogue state in the hidden information state POMDP-based dialogue manager. ACM Transactions on Speech and Language Processing, 7(3).

T. Hastie, R. Tibshirani, and J. Friedman, 2009. The Elements of Statistical Learning: Data Mining, Inference, and Prediction (2nd edition). Springer.

S. Lee and M. Eskenazi, 2012. Exploiting Machine-Transcribed Dialog Corpus to Improve Multiple Dialog States Tracking Methods. In Proceedings of SIGDIAL, 2012.

S. Lee, 2013. Structured Discriminative Model For Dialog State Tracking. Submitted to SIGDIAL, 2013.

A. Raux, B. Langner, D. Bohus, A. W Black, and M. Eskenazi, 2005. Let's Go Public! Taking a Spoken Dialog System to the Real World. In Proceedings of Interspeech.

A. Raux and Y. Ma, 2011. Efficient Probabilistic Tracking of User Goal and Dialog History for Spoken Dialog Systems. In Proceedings of Interspeech.

C. Sutton and A. McCallum, 2006. An Introduction to Conditional Random Fields for Relational Learning. Introduction to Statistical Relational Learning. Cambridge: MIT Press.

B. Thomson and S. Young, 2010. Bayesian update of dialogue state: A POMDP framework for spoken dialogue systems. Computer Speech & Language, 24(4):562-588.

B. Thomson, F. Jurccek, M. Gasic, S. Keizer, F. Mairesse, K. Yu, S. Young, 2010a. Parameter learning for POMDP spoken dialogue models. In Proceedings of SLT.

J. Williams and S. Young, 2007. Partially observable Markov decision processes for spoken dialog systems. Computer Speech & Language, 21(2):393-422.

J. Williams, 2010. Incremental partition recombination for efficient tracking of multiple dialog states. In Proceedings of ICASSP.

J. Williams, 2011. An Empirical Evaluation of a Statistical Dialog System in Public Use, In Proceedings of SIGDIAL.

J. Williams, 2012. A Critical Analysis of Two Statistical Spoken Dialog Systems in Public Use. In Proceedings of SLT.

S. Young, M. Gasic, S. Keizer, F. Mairesse, J. Schatz-mann, B. Thomson and K. Yu, 2010. The Hidden Information State Model: a practical framework for POMDP-based spoken dialogue management. Computer Speech and Language, 24(2):150–174.

Appendix A. Feature Functions

Feature functions are playing a central role to the performance of discriminative models. We describe the feature functions that we used for the challenge system in the following. To facilitate readers' understanding an example of feature extraction is illustrated in Fig. 4.

One of the most fundamental features for dialog state tracking should exploit the confidence scores assigned to an informed hypothesis. The simplest form could be direct use of confidence scores. But often pre-trained confidence measures fail to match the empirical distribution of a given dialog domain (Lee and Eskenazi, 2012; Thomson et al. 2010). Also the distribution of confidence scores that a hypothesis has attained so far may not have the same effect as the total score of the confidence scores (e.g., in Fig. 4, two observations for 61C with confidence score 0.3 vs. 0.6 which is the sum of the scores). Thus we create a feature function that divides the range of confidence scores into bins and returns the frequency of observations that fall into the corresponding bin:

$$inform_k(y, \boldsymbol{x}_1^t) = \begin{cases} y \neq 'None', \ bin\_freq(k, CS_{inf}(y, \boldsymbol{x}_1^t)) \\ otherwise, \ 0 \end{cases} \quad (1)$$

where $CS_{inf}(\cdot)$ returns the set of confidence scores whose action informs y in the sequence of observations $\boldsymbol{x}_1^t$. $bin\_freq(k,\cdot)$ computes the frequency of observations that fall into the k^{th} bin.

There are two types of grounding actions which are popular in spoken dialog systems, i.e., implicit and explicit confirmation. To leverage affirmative or negative responses, the following feature functions are introduced in a similar fashion as the $inform$ feature function:

$$affirm_k(y, \boldsymbol{x}_1^t) = \begin{cases} y \neq 'None', \ bin\_freq(k, CS_{aff}(y, \boldsymbol{x}_1^t)) \\ otherwise, \ 0 \end{cases} \quad (2)$$

$$negate_k(y, \boldsymbol{x}_1^t) = \begin{cases} y \neq 'None', \ bin\_freq(k, CS_{neg}(y, \boldsymbol{x}_1^t)) \\ otherwise, \ 0 \end{cases} \quad (3)$$

where $CS_{aff}(\cdot)$ / $CS_{neg}(\cdot)$ returns the set of confidence scores whose associated action affirms / negates y in the sequence of observations $\boldsymbol{x}_1^t$.

$$impl\_affirm(y, \boldsymbol{x}_1^t) = \begin{cases} y \neq 'None', \ I_{impl\_aff}(y, \boldsymbol{x}_1^t) \\ otherwise, \ 0 \end{cases} \quad (4)$$

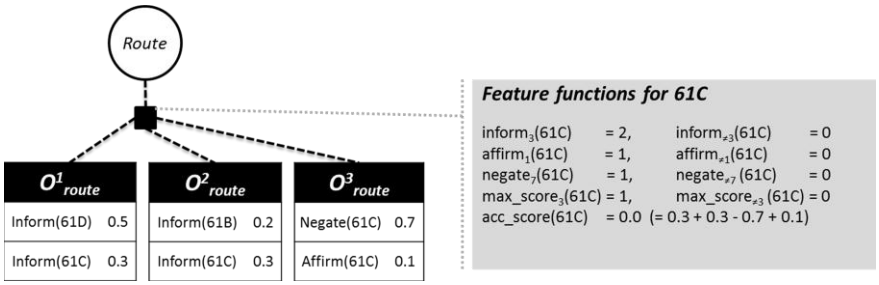


Figure 4: A simplified example of feature extraction for the route concept. It shows the values that each feature will have when three consecutive user inputs are given.

where $I_{impl\_aff}(\cdot)$ indicates whether or not the user has negated the system's implicit confirmation in the sequence of observations $\boldsymbol{x}_1^t$.

One of interesting feature functions is the so-called baseline feature which exploits the output of a baseline system. The following feature function emulates the output of the baseline system which always selects the top ASR hypothesis for the entire dialog:

$$max\_score_k(y, \boldsymbol{x}_1^t) = \begin{cases} y \neq 'None', & bin(k, MAX\_CS_{inf}(y, \boldsymbol{x}_1^t)) \\ otherwise, & 0 \end{cases} \quad (5)$$

where $MAX\_CS_{inf}(\cdot)$ returns the maximum confidence score whose action informs y in the sequence of observations $\boldsymbol{x}_1^t$. $bin(k,\cdot)$ indicates whether or not the maximum score falls into the k^{th} bin.

Yet another feature function of this kind is the accumulated score which adds up all confidence scores associated with *inform* and *affirm* and subtracts the ones with *negation*:

$$acc\_score(y, \boldsymbol{x}_1^t) = \begin{cases} y \neq 'None', & \sum CS_{inf}(y, \boldsymbol{x}_1^t) \\ & +\sum CS_{aff}(y, \boldsymbol{x}_1^t) \\ & -\sum CS_{neg}(y, \boldsymbol{x}_1^t) \\ otherwise, & 0 \end{cases} \quad (6)$$

Since we have a partition-based tracker, it is also possible to take advantage of its output:

$$pbm\_score(y, \boldsymbol{x}_1^t) = \begin{cases} y \neq 'None', & PBM(y, \boldsymbol{x}_1^t)) \\ otherwise, & 0 \end{cases} \quad (7)$$

where $PBM(\cdot)$ returns the posterior probability of a hypothesis estimated by the partition-based tracker. Note that such feature functions as $ax\_score(\cdot), acc\_score(\cdot)$ and $PBM(\cdot)$ are not independent of the others defined previously, which may cause generative models to produce deficient probability distributions.

It is known that prior information can boost the performance (Williams, 2012) if the prior is well-estimated. One of advantages of generative models is that they provide a natural mechanism to incorporate a prior. Discriminative models also can exploit a prior by introducing additional feature functions:

$$prior_k(y, \boldsymbol{x}_1^t) = \begin{cases} y \neq 'None', & bin(k, prior\_frac(y)) \\ otherwise, & 0 \end{cases} \quad (8)$$

where $prior\_frac(y)$ returns the fraction of occurrences of y in the set of true labels.

If the system cannot process a certain user request, it is highly likely that the user change his/her goal. The following feature function is designed to take care of such cases:

$$canthelp(y, \boldsymbol{x}_1^t) = \begin{cases} y \neq 'None', & I_{ooc}(y) \\ otherwise, & 0 \end{cases} \quad (9)$$

where $I_{ooc}(\cdot)$ indicates whether or not y is out-of-coverage.

As with other log-linear models, we also have feature functions for bias:

$$bias(y, \boldsymbol{x}_1^t) = 1$$
$$bias_{none}(y, \boldsymbol{x}_1^t) = \begin{cases} y = 'None', & 1 \\ otherwise, & 0 \end{cases} \quad (10)$$

Note that we have an additional bias term for *None* to estimate an appropriate weight for it. Here, *None* is a special value to indicate that the true hypothesis has not yet appeared in the ASR N-best lists. Since there are generally a large number of values for each concept, the probability of the true hypothesis will be very small unless the true hypothesis appears on the N-best lists. Thus we can make inferences on the model very quickly by focusing only on the observed hypotheses at the cost of little performance degradation.