The Dialog State Tracking Challenge

Jason Williams[1], **Antoine Raux**[2*], **Deepak Ramachandran**[3*] **and Alan Black**[4]
[1]Microsoft Research, Redmond, WA, USA [2]Lenovo Corporation, Santa Clara, CA, USA
[3]Nuance Communications, Mountain View, CA, USA [4]Carnegie Mellon University, Pittsburgh, PA, USA
jason.williams@microsoft.com araux@lenovo.com deepak.ramachandran@nuance.com awb@cmu.edu

Abstract

In a spoken dialog system, *dialog state tracking* deduces information about the user's goal as the dialog progresses, synthesizing evidence such as dialog acts over multiple turns with external data sources. Recent approaches have been shown to overcome ASR and SLU errors in some applications. However, there are currently no common testbeds or evaluation measures for this task, hampering progress. The *dialog state tracking challenge* seeks to address this by providing a heterogeneous corpus of 15K human-computer dialogs in a standard format, along with a suite of 11 evaluation metrics. The challenge received a total of 27 entries from 9 research groups. The results show that the suite of performance metrics cluster into 4 natural groups. Moreover, the dialog systems that benefit most from dialog state tracking are those with less discriminative speech recognition confidence scores. Finally, generalization is a key problem: in 2 of the 4 test sets, fewer than half of the entries out-performed simple baselines.

1 Overview and motivation

Spoken dialog systems interact with users via natural language to help them achieve a goal. As the interaction progresses, the dialog manager maintains a representation of the state of the dialog in a process called *dialog state tracking* (DST). For example, in a bus schedule information system, the dialog state might indicate the user's desired bus route, origin, and destination. Dialog state tracking is difficult because automatic speech recognition (ASR) and spoken language understanding (SLU) errors are common, and can cause the system to misunderstand the user's needs. At the same time, state tracking is crucial because the system relies on the estimated dialog state to choose actions – for example, which bus schedule information to present to the user.

Most commercial systems use hand-crafted heuristics for state tracking, selecting the SLU result with the highest confidence score, and discarding alternatives. In contrast, statistical approaches compute scores for many *hypotheses* for the dialog state (Figure 1). By exploiting correlations between turns and information from external data sources – such as maps, bus timetables, or models of past dialogs – statistical approaches can overcome some SLU errors.

Numerous techniques for dialog state tracking have been proposed, including heuristic scores (Higashinaka et al., 2003), Bayesian networks (Paek and Horvitz, 2000; Williams and Young, 2007), kernel density estimators (Ma et al., 2012), and discriminative models (Bohus and Rudnicky, 2006). Techniques have been fielded which scale to realistically sized dialog problems and operate in real time (Young et al., 2010; Thomson and Young, 2010; Williams, 2010; Mehta et al., 2010). In end-to-end dialog systems, dialog state tracking has been shown to improve overall system performance (Young et al., 2010; Thomson and Young, 2010).

Despite this progress, direct comparisons between methods have not been possible because past studies use different domains and system components, for speech recognition, spoken language understanding, dialog control, etc. Moreover, there is little agreement on how to evaluate dialog state tracking. Together these issues limit progress in this research area.

The Dialog State Tracking Challenge (DSTC) provides a first common testbed and evaluation

[*] Most of the work for the challenge was performed when the second and third authors were with Honda Research Institute, Mountain View, CA, USA

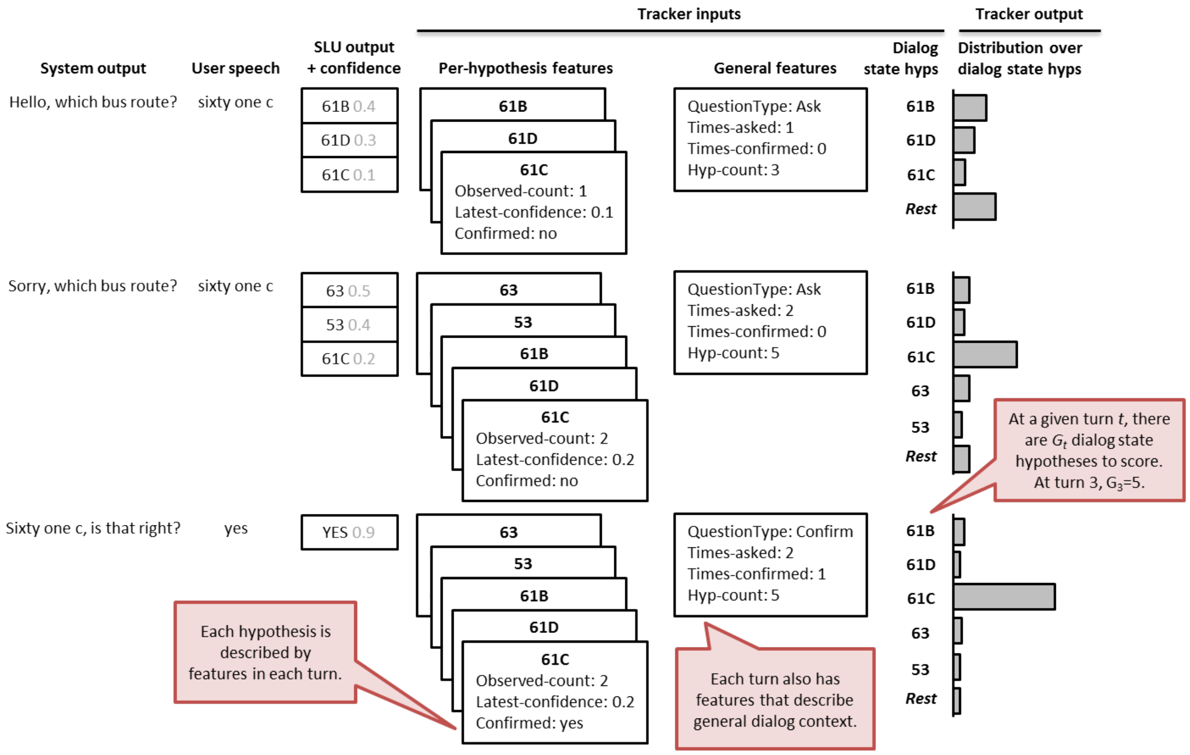


Figure 1: Overview of dialog state tracking. In this example, the dialog state contains the user's desired bus route. At each turn t, the system produces a spoken output. The user's spoken response is processed to extract a set of spoken language understanding (SLU) results, each with a *local* confidence score. A set of N_t dialog state hypotheses is formed by considering all SLU results observed so far, including the current turn and all previous turns. Here, $N_1 = 3$ and $N_2 = 5$. The dialog state tracker uses features of the dialog context to produce a distribution over all N_t hypotheses and the meta-hypothesis that none of them are correct.

suite for dialog state tracking. The DSTC organizers made available a public, heterogeneous corpus of over 15K transcribed and labeled human-computer dialogs. Nine teams entered the challenge, anonymously submitting a total of 27 dialog state trackers.

This paper serves two roles. First, sections 2 and 3 provide an overview of the challenge, data, and evaluation metrics, all of which will remain publicly available to the community (DST, 2013). Second, this paper summarizes the results of the challenge, with an emphasis on gaining new insights into the dialog state tracking problem, in Section 4. Section 5 briefly concludes.

2 Challenge overview

2.1 Problem statement

First, we define the dialog state tracking problem. A dialog state tracker takes as input all of the observable elements up to time t in a dialog, including all of the results from the automatic speech recognition (ASR) and spoken language understanding (SLU) components, and external knowledge sources such as bus timetable databases and models of past dialogs. It also takes as input a set of N_t possible dialog state hypotheses, where a hypothesis is an assignment of values to slots in the system. The tracker outputs a probability distribution over the set of N_t hypotheses, and the meta-hypothesis REST which indicates that none of them are correct. The goal is to assign probability 1.0 to the correct state, and 0.0 to other states. Note that the set of dialog states is *given*. Also note that N_t varies with t – typically as the dialog progresses and more concepts are discussed, the number of candidate hypotheses increases. An example is given in Figure 1.

In this challenge, dialog states are generated in the usual way, by enumerating all slots values that have appeared in the SLU N-best lists or system output up until the current turn. While this approach precludes a tracker assigning a score to an

SLU value that has *not* been observed, the cardinality of the slots is generally large, so the likelihood of a tracker correctly guessing a slot value which hasn't been observed anywhere in the input or output is vanishingly small.

2.2 Challenge design

The dialog state tracking challenge studies this problem as a *corpus-based task* – i.e., dialog state trackers are trained and tested on a *static corpus of dialogs*, recorded from systems using a variety of state tracking models and dialog managers. The challenge task is to *re-run* state tracking on these dialogs – i.e., to take as input the runtime system logs including the SLU results and system output, and to output scores for dialog states formed from the runtime SLU results. This corpus-based design was chosen because it allows different trackers to be evaluated on the same data, and because a corpus-based task has a much lower barrier to entry for research groups than building an end-to-end dialog system.

In practice of course, a state tracker will be used in an end-to-end dialog system, and will drive action selection, thereby affecting the distribution of the dialog data the tracker experiences. In other words, it is known in advance that the distribution in the training data and live data will be mismatched, although the nature and extent of the mis-match are not known. Hence, unlike much of supervised learning research, drawing train and test data from the same distribution in offline experiments may overstate performance. So in the DSTC, train/test mis-match was explicitly created by choosing test data to be from different dialog systems.

2.3 Source data and challenge corpora

The DSTC uses data from the public deployment of several systems in the Spoken Dialog Challenge (SDC) (Black et al., 2010), provided by the Dialog Research Center at Carnegie Mellon University. In the SDC, telephone calls from real passengers of the Port Authority of Allegheny County, who runs city buses in Pittsburgh, were forwarded to dialog systems built by different research groups. The goal was to provide bus riders with bus timetable information. For example, a caller might want to find out the time of the next bus leaving from Downtown to the airport.

The SDC received dialog systems from three different research groups, here called Groups A, B, and C. Each group used its own ASR, SLU, and dialog manager. The dialog strategies across groups varied considerably: for example, Groups A and C used a mixed-initiative design, where the system could recognize any concept at any turn, but Group B used a directed design, where the system asked for concepts sequentially and could only recognize the concept being queried. Groups trialled different system variants over a period of almost 3 years. These variants differed in acoustic and language models, confidence scoring model, state tracking method and parameters, number of supported bus routes, user population, and presence of minor bugs. Example dialogs from each group are shown in the Appendix.

The dialog data was partitioned into 5 training corpora and 4 testing corpora (Table 1). The partioning was intended to explore different types of mis-match between the training and test data. Specifically, the dialog system in TRAIN1A, TRAIN1B, TRAIN1C, TRAIN2, and TEST1 are all very similar, so TEST1 tests the case where there is a large amount of similar data. TEST2 uses the same ASR and SLU but a different dialog controller, so tests the case where there is a large amount of somewhat similar data. TEST3 is very similar to TRAIN3 and tests the case where there is a small amount of similar data. TEST4 uses a completely different dialog system to any of the training data.

2.4 Data preparation

The dialog system log data from all three groups was converted to a common format, which described SLU results and system output using a uniform set of dialog acts. For example, the system speech *East Pittsburgh Bus Schedules. Say a bus route, like 28X, or say I'm not sure.* was represented as *hello(), request(route), example(route=28x), example(route=dontknow)*. The user ASR hypothesis *the next 61c from oakland to mckeesport transportation center* was represented as *inform(time.rel=next), inform(route=61c), inform(from.neighborhood=oakland), inform(to.desc="mckeesport transportation center")*. In this domain there were a total of 9 slots: the bus route, date, time, and three components each for the origin and destination, corresponding to streets, neighborhoods, and points-of-interest like universities. For complete details see (Williams et al., 2012).

	TRAIN					TEST			
	1A	1B	1C	2	3	1	2	3	4
Group	A	A	A	A	B	A	A	B	C
Year(s)	2009	2009	2009	2010	2010	2011	2012	2011-2	2010
Dialogs	1013	1117	9502	643	688	715	750	1020	438
Turns/Dialog	14.7	13.3	14.5	14.5	12.6	14.1	14.5	13.0	10.9
Sys acts/turn	4.0	3.8	3.8	4.0	8.4	2.8	3.2	8.2	4.6
Av N-best len	21.7	22.3	21.9	22.4	2.9	21.2	20.5	5.0	3.2
Acts/N-best hyp	2.2	2.2	2.2	2.3	1.0	2.1	2.0	1.0	1.6
Slots/turn	44.0	46.5	45.6	49.0	2.1	41.4	36.9	4.3	3.5
Transcribed?	yes	yes	yes	yes	yes	yes	yes	yes	yes
Labelled?	yes	no	no	yes	yes	yes	yes	yes	yes
1-best WER	42.9%	41.1%	42.1%	58.2%	40.5%	57.9%	62.1%	48.1%	55.6%
1-best SLU Prec.	0.356	-	-	0.303	0.560	0.252	0.275	0.470	0.334
1-best SLU Recall	0.522	-	-	0.388	0.650	0.362	0.393	0.515	0.376
N-best SLU Recall	0.577	-	-	0.485	0.738	0.456	0.492	0.634	0.413

Table 1: Summary of the datasets. One turn includes a system output *and* a user response. *Slots* are named entity types such as bus route, origin neighborhood, date, time, etc. N-best SLU Recall indicates the fraction of concepts which appear anywhere on the SLU N-best list.

Group B and C systems produced N-best lists of ASR and SLU output, which were included in the log files. Group A systems produced only 1-best lists, so for Group A systems, recognition was *re-run* with the Pocketsphinx speech recognizer (Huggins-Daines et al., 2006) with N-best output enabled, and the results were included in the log files.

Some information in the raw system logs was specific to a group. For example, Group B's logs included information about word confusion networks, but other groups did not. All of this information was included in a "system specific" section of the log files. Group A logs contained about 40 system-specific name/value pairs per turn, and Group B about 600 system-specific name/value pairs per turn. Group C logs contained no system specific data.

3 Labeling and evaluation design

The output of a dialog state tracker is a probability distribution over a set of given dialog state hypotheses, plus the REST meta-hypothesis. To evaluate this output, a label is needed for each dialog state hypothesis indicating its *correctness*.

In this task-oriented domain, we note that the user enters the call with a specific goal in mind. Further, when goal changes do occur, they are usually explicitly marked: since all of the systems first collect slot values, and then provide bus timetables, if the user wishes to change their goal, they need to start over from the beginning. These "start over" transitions are obvious in the logs. This structure allows the correctness of each dialog state to be equated to the correctness of the SLU items it contains. As a result, in the DSTC we labeled the correctness of SLU hypotheses in each turn, and then assumed these labels remain valid until either the call ends, or until a "start over" event. Thus to produce the labels, the labeling task followed was to assign a correctness value to every SLU hypothesis on the N-best list, given a transcript of the words actually spoken in the dialog up to the current turn.

To accomplish this, first all user speech was transcribed. The TRAIN1 datasets had been transcribed using crowd-sourcing in a prior project (Parent and Eskenazi, 2010); the remainder were transcribed by professionals. Then each SLU hypothesis was labled as correct or incorrect. When a transcription exactly and unambiguously matched a recognized slot value, such as the bus route "sixty one c", labels were assigned automatically. The remainder were assigned using crowd-sourcing, where three workers were shown the true words spoken and the recognized concept, and asked to indicate if the recognized concept was correct – *even if it did not match the recognized words exactly*. Workers were also shown dialog history, which helps decipher the user's meaning when their speech was ambiguous. If the 3 workers were not unanimous in their labels (about 4% of all turns), the item was labeled manually by the organizers. The REST meta-hypothesis was not explicitly labeled; rather, it was deemed to be correct if none of the prior SLU results were labeled as correct.

In this challenge, state tracking performance was measured on each of the 9 slots separately, and also on a *joint* dialog state consisting of all the slots. So at each turn in the dialog, a tracker output 10 scored lists: one for each slot, plus a 10th list where each dialog state contains values from all slots. Scores were constrained to be in the range $[0, 1]$ and to sum to 1.

To evaluate tracker output, at each turn, each hypothesis (including REST) on each of the 10 lists was labeled as correct or incorrect by looking up its corresponding SLU label(s). The scores and labels over all of the dialogs were then compiled to compute 11 metrics. **Accuracy** measures the percent of turns where the top-ranked hypothesis is correct. This indicates the correctness of the item with the maximum score. **L2** measures the L^2 distance between the vector of scores, and a vector of zeros with 1 in the position of the correct hypothesis. This indicates the quality of all scores, when the scores as viewed as probabilities.

AvgP measures the mean score of the first correct hypothesis. This indicates the quality of the score assigned to the correct hypothesis, ignoring the distribution of scores to incorrect hypotheses. **MRR** measures the mean reciprocal rank of the first correct hypothesis. This indicates the quality of the ordering the scores produces (without necessarily treating the scores as probabilities).

The remaining measures relate to receiver-operating characteristic (ROC) curves, which measure the discrimination of the score for the highest-ranked state hypothesis. Two versions of ROC are computed – V1 and V2. V1 computes correct-accepts (CA), false-accepts (FA), and false-rejects (FR) as fractions of *all* utterances, so for example

$$CA.V1(s) = \frac{\#CA(s)}{N} \qquad (1)$$

where $\#CA(s)$ indicates the number of correctly accepted states when only those states with score $\geq s$ are accepted, and N is the total number of states in the sample. The V1 metrics are a useful indication of overall performance because they combine discrimination and overall accuracy – i.e., the maximum $CA.V1(s)$ value is equal to accuracy computed above.

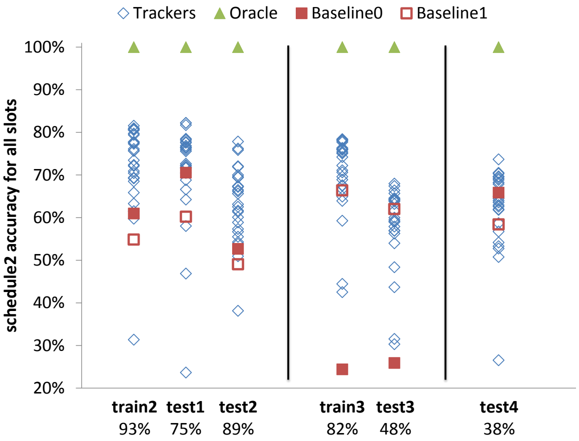


Figure 2: Schedule2 accuracy averaged over slots for every tracker on every dataset. Percentages under the datasets indicate the percent of the trackers which exceeded the performance of both baselines.

V2 considers fractions of *correctly classified utterances*, so for example

$$CA.V2(s) = \frac{\#CA(s)}{\#CA(0)}. \qquad (2)$$

The V2 metrics are useful because they measure the discrimination of the scoring independently of accuracy – i.e., the maximum value of $CA.V2(s)$ is always 1, regardless of accuracy.

From these ROC statistics, several metrics are computed. **ROC.V1.EER** computes $FA.V1(s)$ where $FA.V1(s) = FR.V1(s)$. The metrics **ROC.V1.CA05**, **ROC.V1.CA10**, and **ROC.V1.CA20** compute $CA.V1(s)$ when $FA.V1(s) = 0.05, 0.10$, and 0.20 respectively. **ROC.V2.CA05**, **ROC.V2.CA10**, and **ROC.V2.CA20** do the same using the V2 versions.

Apart from *what* to measure, there is currently no standard that specifies *when* to measure – i.e., which turns to include when computing each metric. So for this challenge, a set of 3 *schedules* were used. **schedule1** includes every turn. **schedule2** include turns where the target slot is either present on the SLU N-best list, or where the target slot is included in a system confirmation action – i.e., where there is some observable new information

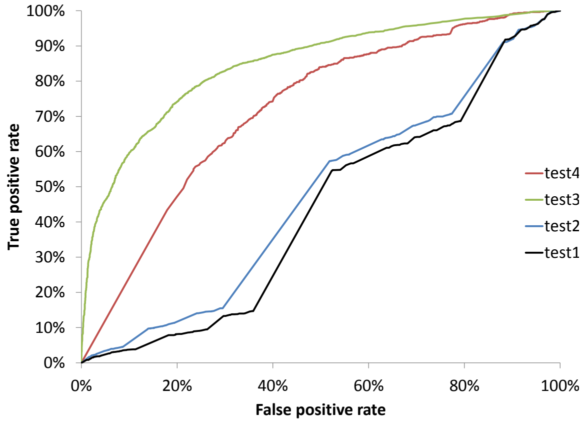


Figure 3: Receiver operating characteristc (ROC) curve for SLU confidence scores of the 1-best hypothesis in the test datasets. The SLU confidence score in TEST3 is most discriminative; TEST1 and TEST2 are the least discriminative.

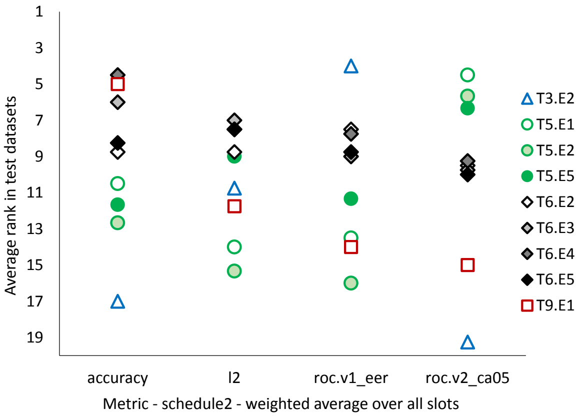


Figure 4: Average rank of top-performing trackers for the four metrics identified in Figure 6. Ranking was done using the given metric, schedule2, and the weighted average of all slots. $Tn.Em$ indicates team n, entry m.

about the target slot. **schedule3** includes only the last turn of a dialog.

In sum, for each tracker, one measurement is reported for each test set (4), schedule (3), and metric (11) for each of the 9 slots, the "joint" slot, and a weighted average of the individual slots (11), for a total of $4 \cdot 3 \cdot 11 \cdot 11 = 1452$ measurements per tracker. In addition, each tracker reported average latency per turn – this ranged from 10ms to 1s.

3.1 Baseline trackers

For comparisons, two simple baselines were implemented. The first (Baseline0) is a majority class baseline that always guesses REST with score 1. The second (Baseline1) follows simple rules which are commonly used in spoken dialog systems. It maintains a single hypothesis for each slot. Its value is the SLU 1-best with the highest confidence score observed so far, with score equal to that SLU item's confidence score.

4 Results and discussion

Logistically, the training data and labels, bus timetable database, scoring scripts, and baseline system were publicly released in late December 2012. The test data (without labels) was released on 22 March 2013, and teams were given a week to run their trackers and send results back to the organizers for evaluation. After the evaluation, the test labels were published. Each team could enter up to 5 trackers. For the evaluation, teams were asked to process the test dialogs online – i.e., to make a single pass over the data, as if the tracker were being run in deployment. Participation was open to researchers at any institution, including the organizers and advisory board. To encourage participation, the organizers agreed not to identify participants in publications, and there was no requirement to disclose how trackers were implemented.

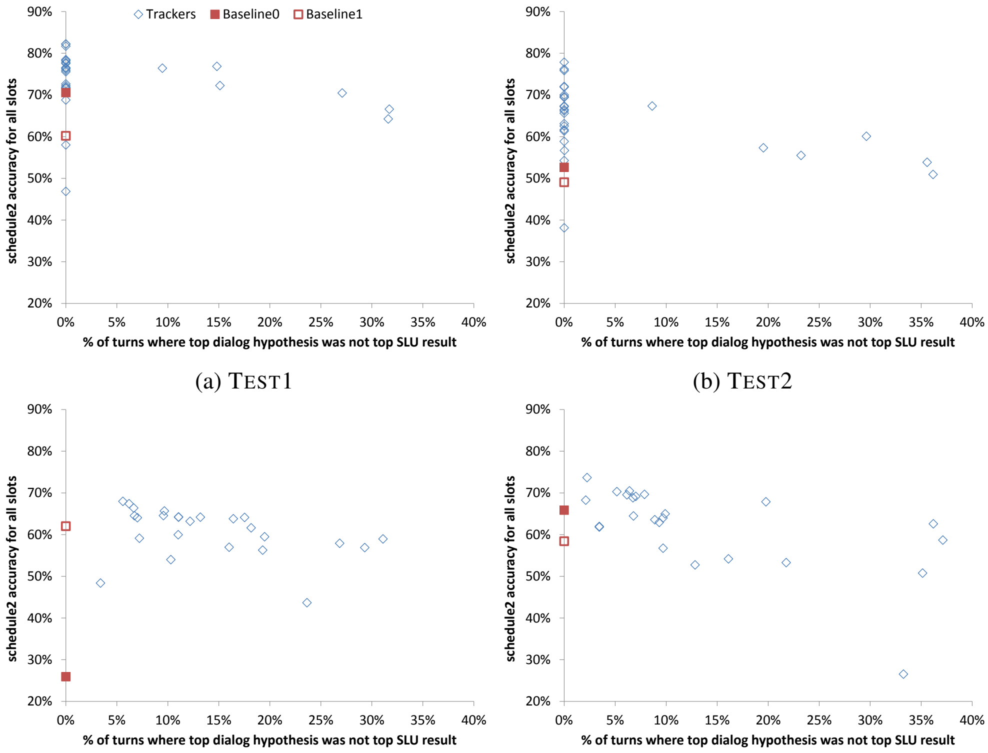
9 teams entered the DSTC, submitting a total of 27 trackers. The raw output and all 1452 measurements for each tracker (and the 2 baselines) are available from the DSTC homepage (DST, 2013).

4.1 Analysis of trackers and datasets

We begin by looking at one illustrative metric, schedule2 accuracy averaged over slots, which measures the accuracy of the top dialog hypothesis for every slot when it either appears on the SLU N-best list or is confirmed by the system.[1] Results in Figure 2 show two key trends. First, relative to the baselines, performance on the test data is markedly lower than the training data. Comparing TRAIN2 to TEST1/TEST2 and TRAIN3 to TEST3, the relative gain over the baselines is much lower on test data. Moreover, only 38% of trackers performed better than a simple majority-class baseline on TEST4, for which there was no matched training data. These findings suggests that generalization is an important open issues for dialog state trackers.

Second, Figure 2 indicates that the gains made

[1] Results using the joint dialog state are broadly similar, and are omitted for space.



(a) TEST1

(b) TEST2

(c) TEST3

(d) TEST4

Figure 5: Percent of highest-scored dialog state hypotheses which did not appear in the top-ranked SLU position vs. schedule2 accuracy over all slots. Trackers – including those with the highest accuracy – for TEST1 and TEST2 rarely assigned the highest score to an SLU hypothesis other than the top. All trackers for TEST3 and TEST4 assigned the highest score to an SLU hypothesis other than the top in a non-trivial percent of turns.

by the trackers over the baselines are larger for Group A systems (TEST1 and TEST2) than for Group B (TEST3) and C (TEST4) systems. Whereas the baselines consider only the top SLU hypothesis, statistical trackers can make use of the entire N-best list, increasing recall – compare the 1-best and N-best SLU recall rates in Table 1. However, Group A trackers almost never assigned the highest score to an item below the top position in the SLU N-best list. Rather, the larger gains for Group A systems seem due to the relatively poor discrimination of Group A's SLU confidence score (Figure 3): whereas the trackers use a multitude of features to assign scores, the baselines rely entirely on the SLU confidence for their scores, so undiscriminative SLU confidence measures hamper baseline performance.

4.2 Analysis of metrics

This challenge makes it possible to study the empirical differences among the evaluation metrics. Intuitively, if the purpose of a metric is to *order* a set of trackers from best to worst, then 2 metrics are similar if they yield a similar ordering over trackers. Specifically, for every metric m, we have a value $x(m, d, s, t)$ where d is the dataset, and s is the evaluation schedule, and t is the tracker. We define $r(m, d, s, t)$ as the *rank* of tracker t when ordered using metric m, dataset d and evaluation schedule s. Using these ranks, we compute Kendall's Tau for every d, s, and *pair* of metrics m_1 and m_2 (Kendall, 1938). We then compute the average Kendall's Tau for m_1 and m_2 by averaging over all d and s.[2]

Results are in Figure 6. Here we see 4 natural clusters emerge: a cluster for **correctness** with Accuracy, MRR, and the ROC.V1.CA measures; a cluster for **probability quality** with L2 and Average score; and two clusters for **score discrimination** – one with ROC.V1.EER and the other with the three ROC.V2 metrics. This finding suggest

[2] A similar analysis over schedules showed that the differences in ranking for different schedules were smaller than for metrics.

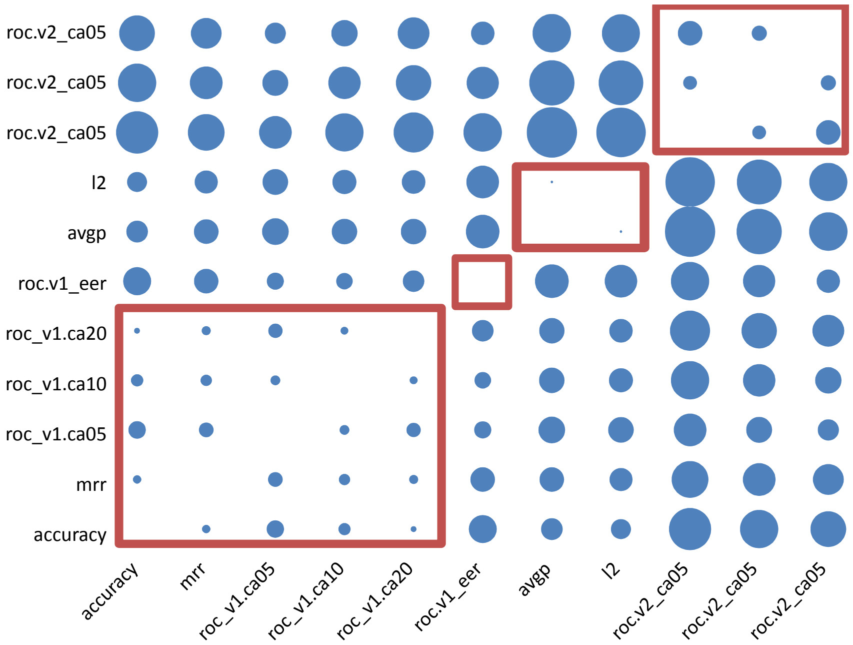


Figure 6: Average divergence between rank orderings produced by different metrics. The size of a circle at (x, y) is given by $1-\tau$, where τ is the average Kendall's Tau computed on the rank orderings produced by methods x and y. Larger circles indicate dissimilar rankings; smaller circles indicate similar rankings; missing circles indicate identical rankings. The red boxes indicate groups of metrics that yield similar rankings.

that measuring one metric from each cluster will contain nearly the same information as all 9 metrics. For example, one might report only Accuracy, L2, ROC.V1.EER, and ROC.V2.CA5.

Using these 4 metrics, we rank-ordered each tracker, using schedule2 and a weighted average of all slots. We then computed the average rank across the 4 test sets. Finally we selected the set of trackers with the top three average ranks for each metric. Results in Figure 4 emphasize that different trackers are tuned for different performance measures, and the optimal tracking algorithm depends crucially on the target performance measure.

5 Conclusion

The dialog state tracking challenge has provided the first common testbed for this task. The data, evaluation tools, and baselines will continue to be freely available to the research community (DST, 2013). The details of the trackers themselves will be published at SIGDIAL 2013.

The results of the challenge show that the suite of performance metrics cluster into 4 natural groups. We also find that larger gains over conventional rule-based baselines are present in dialog systems where the speech recognition confidence score has poor discrimination. Finally, we observe substantial limitations on generalization: in mismatched conditions, around half of the trackers entered did not exceed the performance of two simple baselines.

In future work, it should be verified that improvements in dialog state tracking lead to improvements in end-to-end dialog performance (e.g., task completion, user satisfaction, etc.). In addition, it would be interesting to study dialogs where goal changes are more common.

Acknowledgements

The organizers thank the advisory board for their valuable input on the design of the challenge: Daniel Boies, Paul Crook, Maxine Eskenazi, Milica Gasic, Dilek Hakkani-Tur, Helen Hastie, Kee-Eung Kim, Ian Lane, Sungjin Lee, Teruhisa Misu, Olivier Pietquin, Joelle Pineau, Blaise Thomson, David Traum, and Luke Zettlemoyer. The organizers also thank Ian Lane for his support for transcription, and Microsoft and Honda Research Institute USA for funding the challenge. Finally, we thank the participants for making the challenge successful.

References

AW Black, S Burger, B Langner, G Parent, and M Eskenazi. 2010. Spoken dialog challenge 2010. In *Proc SLT, Berkeley*.

D Bohus and AI Rudnicky. 2006. A 'K hypotheses + other' belief updating model. In *Proc AAAI Workshop on Statistical and Empirical Approaches for Spoken Dialogue Systems, Boston*.

2013. Dialog State Tracking Challenge Homepage. `http://research.microsoft.com/events/dstc/`.

H Higashinaka, M Nakano, and K Aikawa. 2003. Corpus-based discourse understanding in spoken dialogue systems. In *Proc ACL, Sapporo*.

D Huggins-Daines, M Kumar, A Chan, A W Black, M Ravishankar, and A I Rudnicky. 2006. PocketSphinx: A Free, Real-Time Continuous Speech Recognition System for Hand-Held Devices. In *Proc ICASSP, Toulouse*.

M Kendall. 1938. A new measure of rank correlation. *Biometrika*, 30(1-2):81–89.

Y Ma, A Raux, D Ramachandran, and R Gupta. 2012. Landmark-based location belief tracking in a spoken dialog system. In *Proc SigDial, Seoul*.

N Mehta, R Gupta, A Raux, D Ramachandran, and S Krawczyk. 2010. Probabilistic ontology trees for belief tracking in dialog systems. In *Proc SigDial, Tokyo*.

T Paek and E Horvitz. 2000. Conversation as action under uncertainty. In *Proc UAI, Stanford*, pages 455–464.

G Parent and M Eskenazi. 2010. Toward Better Crowdsourced Transcription: Transcription of a Year of the Let's Go Bus Information System Data. In *Proc SLT, Berkeley*.

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

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

JD Williams, A Raux, D Ramachandran, and AW Black. 2012. Dialog state tracking challenge handbook. Technical report, Microsoft Research.

JD Williams. 2010. Incremental partition recombination for efficient tracking of multiple dialogue states. In *Proc. of ICASSP*.

SJ Young, M Gašić, S Keizer, F Mairesse, J Schatzmann, 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 1: Sample dialog excerpts

Below are transcripts of calls in the DSTC corpus. The user's side shows utterances as transcribed by human annotators.

Group A

SYS: Welcome to the CMU Let's Go bus information system. To get help at any time, just say Help or press zero. What can I do for you?
USR: *61c evening hours*
SYS: For example, you can say, when is the next 28X from DOWNTOWN to THE AIRPORT? or I'd like to go from MCKEESPORT to HOMESTEAD tomorrow at 10 a.m.
USR: *i'd like to go [noise]*
SYS: Where are you leaving from?
USR: *from Kennywood tomorrow at 6:30 PM*
SYS: Leaving from MCKEE. Is this correct?
USR: *yes*
SYS: Okay What is your destination?
USR: *downtown*
SYS: Going to DOWNTOWN. Is this correct?
USR: *yes*
SYS: Right When do you want to travel?
USR: *6:30 PM*
SYS: Leaving at 6 30 p.m.. Is this correct?
USR: *[noise] yes*
SYS: Alright Hold on. I'll look that up. There is a 71D leaving FIFTH AVENUE AT DARRAGH MCKEE at 6 40 p.m.. It will arrive at FIFTH AVENUE AT MARKET at 6 58 p.m.. To get more information about buses related to this trip, you can say, when is the next bus, or, when is the previous bus. To ask about a different trip, you can say, start a new query. If you are finished, you can say goodbye.
USR: *start a new query*
SYS: Okay, let's start from the beginning.
USR: ...

Group B

SYS: East Pittsburgh Bus Schedules. Say a bus route, like 28X, or say I'm not sure.
USR: *61A*
SYS: Okay, 61A. To change, say go back. Where are you leaving from?
USR: *Downtown*
SYS: Okay, downtown. You can always say go back. And where are you going to?
USR: *East Pittsburgh East Pittsburgh*
SYS: Pittsburgh. Is that right?

USR: *No*
SYS: Oh, sorry, where are you going to?
USR: *East Pittsburgh*
SYS: Okay, East Pittsburgh. Do you want times for the next few buses? Say yes or no.
USR: *Yes*
SYS: Ok, the next few buses. To change, say go back. At 7:45 PM today, there is a 61 A, from Fifth Avenue And Market Street, Downtown, arriving Dynamo Alley And Beech Avenue, In East Pittsburgh, at 8:42 PM. Say repeat, next, previous, or start over.

Group C

SYS: Welcome to the Lets Go! bus information system for Pittsburgh. Where would you like to leave from?
USR: *[noise]*
SYS: Where do you wanna leave from?
USR: *[noise] um I'm leaving from Forbes avenue and Murray avenue*
SYS: I'm sorry but I do not know any stops in the moon neighborhood. The routes I currently cover are the following: 28X, 54C, 56U, 61A, 61B, 61C, 61D and 64.
USR: *um I need the scheduling information for the 61D*
SYS: Where would you like to leave from?
USR: *Forbes and Murray*
SYS: Where do you wanna go?
USR: *Greenfield*
SYS: When are you going to take that bus?
USR: *61D*
SYS: When are you going to take that bus?
USR: *61D*
SYS: When would you like to travel? Say next bus for the the next bus .
USR: *next bus*
SYS: The next bus from forbes avenue at murray to greenfield is the 61D, leaving at 8 43 p.m. . It arrives at murray avenue at loretta at 8 52 p.m. . You may now say. next bus, previous bus, restart, or goodbye.
USR: *[noise] good-bye*
SYS: Thank you goodbye

Appendix 2: Example log fragment

Following is a log from a call in TRAIN3. Some elements have been omitted for space. Logs are formatted in JavaScript Object Notation (JSON).

```
{
 "session-id": "dt-201007221854-4808D",
 "turns": [ {
  "input": {
   "asr-hyps": [
    { "asr-hyp": "sixty one b",
      "score": 0.834314,
    },
    ...
   ],
   "slu-hyps": [
    { "slu-hyp": [
      {"act": "inform",
       "slots": [["route", "61b"]]},
     ],
     "score": 0.834314,
    },
    ...
   ],
   "audio-file": "002.raw",
   "start-time": 7.504
   },
   "output": {
    "start-time": 0.078,
    "transcript": "East Pittsburgh Bus
      Schedules. Say a bus route, like
      28X, or say I'm not sure.",
    "dialog-acts": [
     { "act": "hello",
       "slots": [] },
     { "act": "request",
       "slots": [["route",null]] },
     { "act": "example",
       "slots": [["route","28x"]] },
     { "act": "example",
       "slots": [["route","dontknow"]] }
    ],
   }
   "system-specific": {
    "wcn": {
     "best_path_score": 0.9965,
     "best_path": {
      "word": ["sixty","one","b"],
      "prob": [ 1.0,1.0,0.9965 ],
     }
     "network": [ ... ]
    },
    "nbest": [
     {
      "normSpeechLhood": -152.654,
      "lastSpeechFrame": 266,
      "numFrames": 354,
      "udelta": -3.0280,
      "speechLikelihood": -15876.0,
     },
     ...
    ],
    ...
   },
  },
  ...
 ]
}
```