D3 Toolkit: A Development Toolkit for Daydreaming Spoken Dialog Systems

Donghyeon Lee, Kyungduk Kim, Cheongjae Lee,
Junhwi Choi, and Gary Geunbae Lee

Department of Computer Science and Engineering
Pohang University of Science and Technology
{semko,getta,lcj80,chasunee,gblee}@postech.ac.kr

Abstract. Recently various data-driven spoken language technologies have been applied to spoken dialog system development. However, high cost of maintaining the spoken dialog systems is one of the biggest challenges. In addition, a fixed corpus collected by human is never enough to cover diverse real user's utterances. The concept of a daydreaming dialog system can solve the problem by making the system learn from previous human-machine dialogs. This paper introduces D3 (Daydreaming Dialog system Development) toolkit, which is a back-end support toolkit for the development of the daydreaming spoken dialog systems. For reducing human efforts, D3 toolkit generates new utterances with semantic annotation and new knowledge by analyzing the usage log file. The new added corpus is determined by verifying proper candidates using semi-automatic methods. The augmented corpus is used for building improved models and self-evolution of the dialog system is possible by replacing the old models. We implemented the D3 toolkit using web-based technologies to provide a familiar environment to non-expert end-users.

Keywords: Spoken Dialog System, Statistical NLP, Daydreaming Computer, Failure-driven Learning, Dialog Development Toolkit.

1 Introduction

In recent years, data-driven spoken dialog systems have been studied by many researchers. In general, spoken dialog systems (SDS) consist of three major components: automatic speech recognition (ASR), spoken language understanding (SLU) and dialog management (DM). For building each model, developers need to prepare annotated dialog corpus and domain-specific knowledge. However, corpus preparation requires tedious human efforts.

To reduce the laborious work, several development toolkits such as SUEDE [1], CSLU Toolkit [2] and DialogDesigner [3] have been developed to help developers for rapid system design. For example, Jung et al. [4] developed Dialog Studio to reduce engineering works and to upgrade all components including ASR, SLU, and DM together. Nevertheless, there are still problems when using and managing data-driven spoken dialog systems in a practical field. A fixed corpus collected by human is never enough to cover a real user's utterance patterns.

Data-driven user simulation techniques are widely used for learning optimal dialog strategies in a statistical dialog management framework and for automated evaluation of spoken dialog systems [5]. The user simulation technique is an alternative way to resolve common weaknesses of dialog systems such as the scarceness of the training corpus and the cost of an evaluation made by real users. However, the problem of data-driven user simulation is the limitation of user patterns. The response patterns from data-driven simulated user usually tend to be limited to the training data although several exploration algorithms are used to find unseen patterns. Therefore, the patterns lack of reality and it is not easy to simulate unseen user patterns.

The problems can be solved by a concept of a daydreaming computer [6]. The daydreaming computer should not be idle when left unemployed by users, but daydreaming. Learning from experience, the one important role of daydreaming, can be applied to the SDS. Therefore, the daydreaming dialog system can keep trying to learn from failures and successes while the users do not use the system. We developed a support tool for a daydreaming dialog system development, called D3 (Daydreaming Dialog Development) toolkit, to help the developer to find the real user's new utterance patterns.

This paper is organized as follows: Section 2 introduces the concept of the daydreaming dialog system. D3 toolkit components and detail strategies are proposed in section 3. Preliminary experiments and how to implement the D3 toolkit will be described in section 4. Finally, section 5 draws a conclusion and a future work.

2 Daydreaming Dialog System

An intriguing aspect of humans is that we spend much time engaged in thought not directly related to the current situation or environment. This type of thought, usually called daydreaming, involves recalling or imaging personal or vicarious experiences in the past of future. For both humans and computer, Mueller and Dyer [6] discussed the following function of daydreaming: support for creativity, future planning and rehearsal, learning from experience, emotion modification and motivation.

The daydreaming computer takes as input situational descriptions, and produces as output actions that it would perform in given situation and several daydreams when the computer is idle. The daydreaming computer learns as it daydreams by indexing daydreams, planning strategies, and future plans into memory for future use.

We believe that the concept of the daydreaming can be extended to the SDS because the daydreaming enables the SDS to learn from successes and failures. While SDS does not process user's utterances, self-evolutionary modules in the daydreaming dialog system might learn to process the problematic utterances at the background. This daydreaming process would correct the problematic situations and the system could generate appropriate responses when facing the same situation again in the future.

For example, the user says "I am travelling for sightseeing", but some errors occur due to wrong predictions in ASR, SLU or DM modules. While the SDS is idle, the

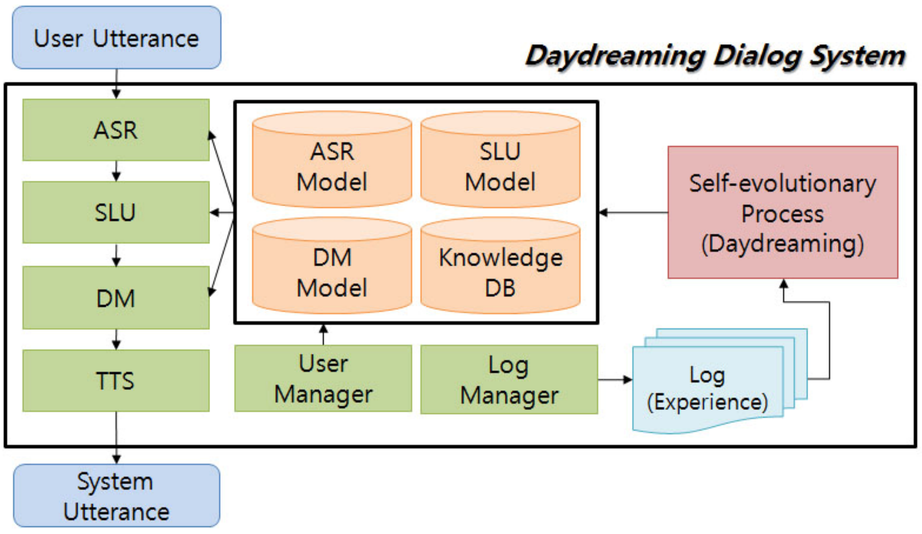


Fig. 1. Overview of daydreaming spoken dialog system architecture

system automatically detects and corrects the errors and updates all models to smoothly process the same utterance in the future. After daydreaming, the system could generate a response, like "How long are you staying", when the user says the same utterance.

Fig. 1 illustrates overview of daydreaming spoken dialog system architecture. The log manager creates log files, which are considered as system experiences including successes and failures. The user manager is used to manage user accounts and load models for a user. When the idling time of the system comes, the system starts a daydreaming process. Daydreaming is performed by a self-evolutionary process, which involves analyzing the human-machine dialog log, trying to extract the patterns, and updating the models to learn from successes and failures. Our D3 toolkit supports this self-evolutionary process of the daydreaming dialog system can be easily done by a real user.

3 Daydreaming Dialog System Development Toolkit

Dialog Studio has provided a convenient way to prepare dialog corpus and update all models without large human efforts. However, it is still time-consuming at the annotation step because the developer should annotate new utterances manually to add new patterns without most probable semantic tags.

To address this problem, we have developed the D3 support toolkit for easily maintaining SDS. The D3 toolkit supports three main functions: 1) finding out-of-patterns

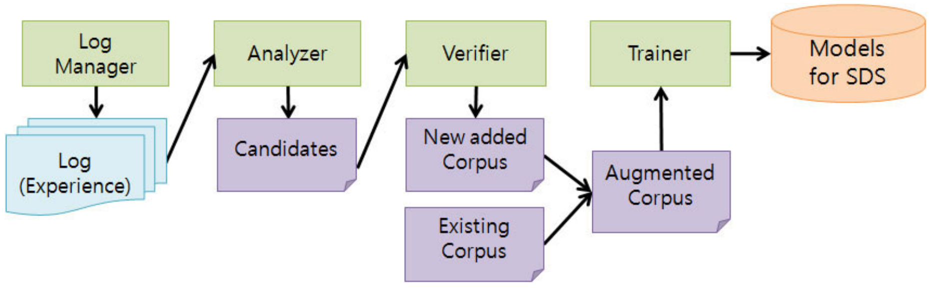


Fig. 2. Procedures of the D3 Toolkit

to make errors in human-machine dialog logs, 2) suggesting their transcriptions for training a language model, and 3) annotating semantic tags tentatively.

3.1 Procedures of the D3 toolkit

As shown in Fig 2, D3 toolkit starts from loading a set of log files which have been generated when the real users talked to the dialog system in the past. Log analysis is divided into three parts including: ASR, SLU, and Knowledge part. Log analyzer generates candidates which include some system errors due to out-of-patterns. The developer is only required to verify and modify auto-generated candidates like active learning [7]. New utterance patterns are added to the existing corpus and training all models is executed.

3.2 Logging Step

All the processed human-machine dialogs are saved as log files, which are in the XML-format, by the log manager of SDS. An example of the log file is illustrated in Fig 3. Each module of SDS generates information, which is used in the analysis step. ASR module provides a recognized text (ASR RESULT), a confidence score (CONFIDENCE) and a wave filename (WAVE). SLU module provides a semantic structure (DIALOG_ACT, MAIN_GOAL, SLOT) and confidence score (SLU_CONFIDENCE). DM module records discourse history such as previous user intention (PREV_DA, PREV_MG, PREV_SL), previous system action (PREV_SA), filled slots (FILLED_SLOT) and current system action (SYSTEM_ACTION). The FILLED_SLOT represents which slots are filled by the user during the current dialog.

3.3 Analysis Step

ASR Evolution. Traditional ASRs requires both language model (LM) and acoustic model (AM). In our system, we use a domain-specific language model and a generic acoustic model to build a spoken dialogue system.

```
<UTTERANCE id="5" time="2010-06-01 21:39:23">
  <ASR>
    <WAVE>/wav/result005.wav</WAVE>
    <RESULT>오늘 21시에는 무슨 프로그램 하지?</RESULT>
    <CONFIDENCE>0.8587191</CONFIDENCE>
  </ASR>
  <SLU>
    <DIALOG_ACT>wh_question</DIALOG_ACT>
    <MAIN_GOAL>search_program</MAIN_GOAL>
    <SLOT name="date">today</SLOT>
   <SLOT name="time">21:00</SLOT>
    <SLU_CONFIDENCE>0.87671</SLU_CONFIDENCE>
  </SLU>
  <DM>
    <PREV_DA>request</PREV_DA>
   <PREV_MG>search_program</PREV_MG>
   <PREV_SL>date,genre</PREV_SL>
   <PREV_SA>inform(program)</PREV_SA>
   <FILLED_SLOT>date,time</FILLED_SLOT>
    <SYSTEM_ACTION>infrom(program)</ SYSTEM_ACTION>
  </DM>
</UTTERANCE>
```

Fig. 3. Example of the log file

ASR evolution involves LM adaptation and AM adaptation (Fig 4). ASR-1 represents the ASR of the SDS and ASR-2 represents the one of the toolkit in Fig 4. They are same modules except using different LM.

To learn out-of-patterns, our toolkit first tries to find mis-recognized utterances in the log file. Out-of-patterns may be recognized incorrectly because they were never seen in training LM of ASR-1. We preliminary used ASR confidence score [8] to detect out-of-patterns because the confidence score has been commonly used to detect erroneous utterances; out-of-patterns could be detected by the filter if their ASR confidence score at the utterance level is lower than a pre-defined threshold.

To obtain most probable transcriptions, out-of-patterns are recognized again by ASR-2 in which LM was trained with an extended corpus. The extended corpus contains a basic dialog corpus for SDS, a general conversational corpus, and a web data related to the domain to cover a variety of utterance patterns. In the ASR-2, we can use larger LM because this process is off-line and the decoding time is not critical.

After re-recognizing out-of-patterns, the toolkit extracts some recognized utterances with a high confidence score. These may be recognized correctly now although they were mis-recognized. These utterances are used to train new LM in the ASR-1, and then the ASR-1 would recognize them correctly when facing similar patterns.

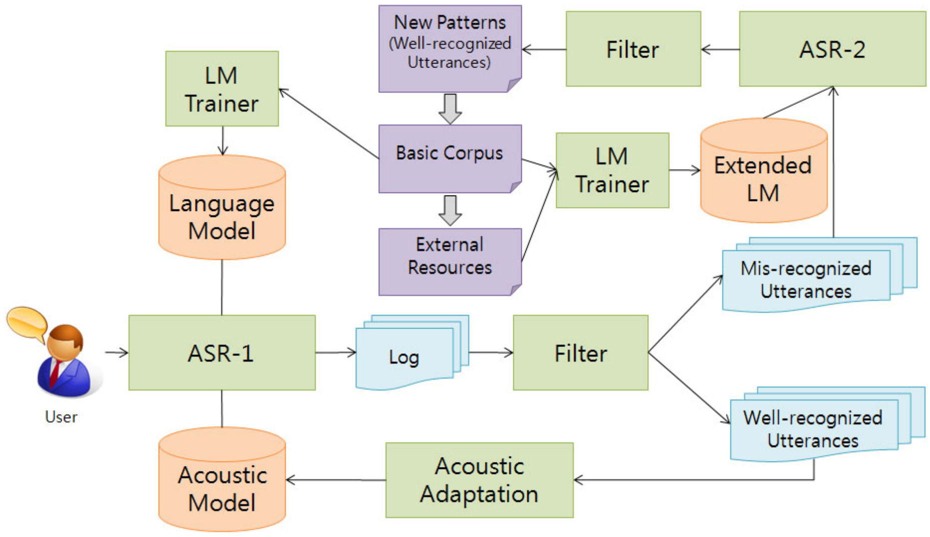


Fig. 4. ASR evolution flow

In addition, well-recognized utterances in the log file are also used to adapt AM using HTK toolkit [9] because acoustic adaptation for a specific speaker is important to increase the recognition accuracy.

SLU Evolution. The flow of SLU evolution is illustrated in Fig 5. For SLU Evolution, we have considered three cases:

- Case 1 : low ASR confidence score
- Case 2 : high ASR confidence score and low SLU confidence score
- Case 3 : high ASR confidence score and high SLU confidence score

In Case 1, the new utterances could be extracted by ASR evolution. In Case 2, the some errors occur in the current SLU model. Therefore, the utterances in Case 1 and Case 2 should be newly annotated by appropriate semantic tags to train the statistical SLU model. In our system, a hybrid approach to SLU module [10] is used to extract the semantic frames of the user's utterances.

The hybrid intention recognizer in Fig 5 is based on a hybrid model that combines the utterance model and the dialog context model. In the SDS, the utterance model is usually used for SLU to predict the user intention from the utterance itself. The dialog context model predicts the probable user intention given the current dialog context. For the dialog context model, we use CRF model trained on the dialog corpus for SDS. We use the following features for the discourse-based model: previous dialog act, previous main goal, previous component slots and previous system action. For each utterance, this information can be obtained in the log file. The hybrid model merges hypotheses from the current SLU model with hypotheses from the dialog context model to find the best overall matching to user intention.

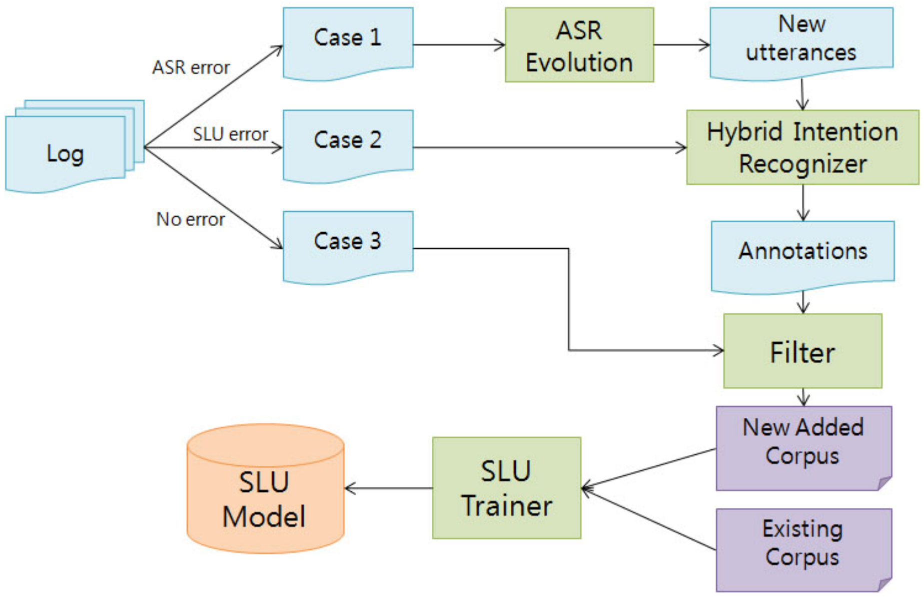


Fig. 5. SLU evolution flow

In Case 3, we considered the SLU results as machine-labeled utterances for unlabeled utterances in semi-supervised learning method [11].

Knowledge Database. One of the most important resources in building SDS is knowledge database (KDB) containing domain-specific information (e.g. TV program schedule) to be informed or suggested to the user. The SDS cannot sometimes find the information because the KDB does not contain what the user said. For example, the user says "I want to watch drama 'Cinderella's Sister'", but the system does not provide any information because the new program 'Cinderella's Sister' was not updated previously.

In general, the KDB is used to build ASR and SLU modules because they have a set of slot values to be recognized and to be extracted. However, the users do not know which items are included in the KDB; therefore, the users often say out-of-vocabulary (OOV) terms for the slot values. In this case, the daydreaming dialog system has to detect OOV terms to handle unseen items for the next time.

In our toolkit, the OOV terms are detected by using word confidence scoring method [12]. In this method, a search network for recognition is constructed using utterance patterns and a keyword slot is assumed as an OOV term if a word confidence score is low. Next, the phonetic recognizer transcribes the phonemes of the OOV terms, and then phoneme to grapheme conversation module generates n-best keywords. After that, the new knowledge is extracted using a knowledge importer, which is a domain-specific module to extract structured information from external

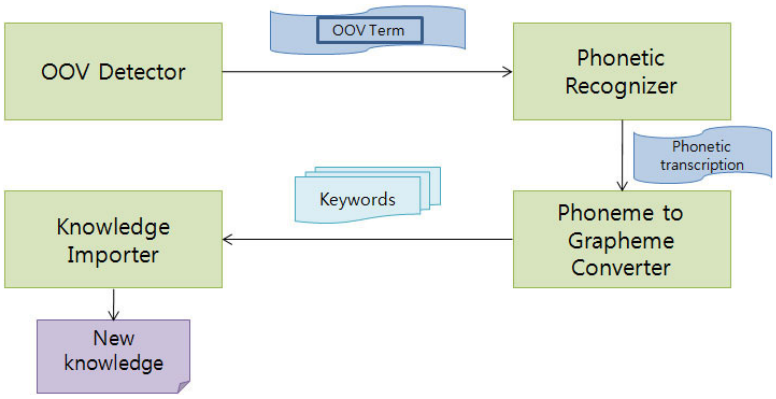


Fig. 6. A strategy of knowledge acquisition for OOV terms

knowledge sources such as web. Fig 6 shows a strategy of knowledge acquisition for OOV terms.

3.4 Verification Step

At the analysis step, the D3 toolkit suggests a set of unseen utterance patterns which should be adapted to each model for improving the performance. However, these candidates may still include many errors because they were generated automatically through each evolution module. Therefore, these candidates should be verified whether they are correct before training each model. The D3 toolkit provides the verification step using active learning method [7]. We empirically defined a set of thresholds for ASR and SLU models. If the confidence score of the candidate exceeds a given threshold, the candidate is automatically added to train the model. Otherwise, the candidate is verified by a real user.

At the verification step, the D3 toolkit provides useful information such as the most likely probable transcription and semantic tags for the user to select which utterances should be added to new models. The user can modify the candidate by listening to the recorded speech files. We believe that our toolkit can be easily used by the real users because the verification step is completed by just selecting the checkbox based on the user's intuition.

3.5 Training Step

After the verification, the existing corpus combines with the new corpus. The augmented corpus is used for building new ASR, SLU and knowledge model. By replacing the models, the self-evolution process is completed.

4 Preliminary Experiments

4.1 Corpus

For preliminary experiments, we collected human-machine dialogs of about 114 utterances from 25 dialogs in Korean, which were based on a set of pre-defined subjects relating to the TV guidance task. We tested on our example-based dialog system [13]. Table 1 shows the result type of the collected utterances. The result type relates to three cases mentioned in SLU evolution. The number of utterances in Case 2 is small because SLU models are closely related to ASR models in SDS development. After the evolutionary process, 4 utterance patterns with semantic annotations and 2 knowledge sources are newly added to the existing corpus.

Table 1. Result type of the collected utterances

	Case 1	Case 2	Case 3
# of utterances	51	9	54

4.2 Dialog System Performance

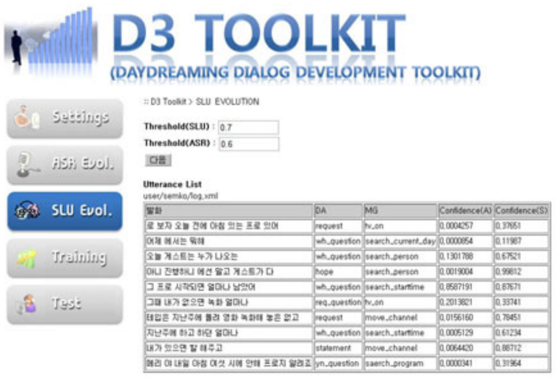
To evaluate the effectiveness of our toolkit, we compared the dialog system performance before and after the daydreaming. We quantified ASR according to word error rate (WER) and SLU according to concept error rate (CER). After updating model, WER is decreased from 48.64% to 46.97% and CER is decreased from 32.58% to 29.21%. WER is too high compared to WER in conventional SDSs because all dialogs include many new patterns. We calculated the task completion rate (TCR) for the dialog system evaluation. TCR is increased from 48.00% to 56.00%, which means two more tasks are completed. The experiment results are shown in table 2. The system performance is improved after daydreaming because the system succeeds in processing the utterances, which cause errors before daydreaming, by updating all models.

Table 2. Experiment results

	Before daydreaming	After daydreaming
WER (ASR)	48.64%	46.97%
CER (SLU)	32.58%	29.21%
TCR (DM)	48.00%	56.00%

4.3 Implementation

D3 toolkit was implemented into client-server architecture. To access the toolkit easily and provide a familiar environment, the graphic user interface is implemented using web-based technologies. For efficiency and adaptability, the analyzing and training part was developed using standard C++ library. The screen shot in fig. 7 shows the interface of D3 toolkit.



:: D3 Toolkit > SLU EVOLUTION

Threshold(SLU) : 0.7

Threshold(ASR) : 0.6

Utterance List
user/semko/log.xml

발화	DA	MG	Confidence(A)	Confidence(S)
로 보자 오늘 전에 아침 있는 프로 있어	request	tv_on	0.0004257	0.37651
어제 에서는 뭐해	wh_question	search_current_day	0.0000854	0.11987
오늘 게스트는 누가 나오는	wh_question	search_person	0.1301788	0.67521
아니 진행하니 예선 말고 게스트가 다	hope	search_person	0.0019004	0.99812
그 프로 시작되면 얼마나 남았어	wh_question	search_starttime	0.8587191	0.87671
그때 내가 없으면 녹화 얼마나	req_question	tv_on	0.2013821	0.33741
테입은 지난주에 틀러 영화 녹화해 놓은 없고	request	move_channel	0.0156160	0.78451
지난주에 하고 하던 얼마나	wh_question	search_starttime	0.0005129	0.61234
내가 있으면 말 해주고	statement	move_channel	0.0064420	0.88712
혜리 아 내일 아침 여섯 시에 안해 프로지 알려조	yn_question	saerch_program	0.0000341	0.31964

Fig. 7. Screen shot of the D3 Toolkit user interface

5 Conclusions and Future Work

We introduced the daydreaming spoken dialog system which can be semi-automatically improved by learning from the past human-machine dialogs. We also implemented the D3 toolkit to support the self-evolutionary process of the daydreaming dialog system. The D3 toolkit generates new corpus by analyzing the log file. The models of the dialog system can be semi-automatically upgraded using the verification step. The main advantage of D3 toolkit is that little human effort is required to maintain the SDS by feeding out-of-patterns into the models. In the future work, we will apply D3 toolkit to different domains and extensively evaluate the updated system. In addition, the process for a dialog model evolution will be considered at the analyzing step.

Acknowledgments. This work was supported by the Industrial Strategic technology development program, 10035252, Development of dialog-based spontaneous speech interface technology on mobile platform funded by the Ministry of Knowledge Economy (MKE, Korea).

References

1. Anoop, K.S., Scott, R.K., Chen, J., Landay, J.A., Chen, C.: SUEDE: Iterative, Informal Prototyping for Speech Interfaces. In: Video poster in Extended Abstracts of Human Factors in Computing Systems: CHI 2001, Seattle, pp. 203–204 (2001)
2. Sutton, s., Cole, R., de Villiers, J., Schalkwyk, J., Vermeulen, P., Macon, M., Yan, Y., Kaiser, E., Rundle, B., Shobaki, K., et al.: Universal Speech Tools: the CSLU toolkit. In: Proc. Internat. Conf. on Spoken Language Processing (ICSLP), pp. 3221–3224 (1998)
3. Dybkjær, H., Dybkjær, L.: DialogDesigner: Tools support for dialogue model design and evaluation. Lang. Resour. Eval. 40(1), 87–107 (2006)
4. Jung, S., Lee, C., Kim, S., Lee, G.G.: Dialog Studio: A workbench for data-driven spoken dialog system development and management. The Journal of Speech Communication 50(8-9), 683–697 (2008)

5. Scheffler, K., Young, S.: Corpus-based dialogue simulation for automatic strategy learning and evaluation. In: NAACL Workshop on Adaptation in Dialogue Sys-tems, pp. 64–70 (2001)
6. Mueller, E.T., Dyer, M.G.: Daydreaming in humans and computes. In: Proceedings of the Ninth International Joint Conference on Artificial Intelligence. University of California, Los Angeles (1985)
7. Bonwell, C.C., Eison, J.A.: Active learning: Creating excitement in the classroom. In: ASHE-ERIC Higher Education Report No. 1. The George Washington University, School of Education and Human Development, Washington (1991)
8. Jiang, H.: Confidence measures for speech recognition. Speech Communication 45(4), 455–470 (2005)
9. Hidden Markov Toolkit (HTK), `http://htk.eng.cam.ac.uk/`
10. Lee, S., Lee, C., Lee, J., Noh, H., Lee, G.G.: Intention-based Corrective Feedback Generation using Context-aware Model. In: Proceedings of the 2nd International Conference on Computer Supported Education (CSEDU 2010), Valencia (2010)
11. Tur, G., Tur, D.H., Schapire, R.E.: Combining Active and Semi-Supervised Learning for Spoken Language Understanding. The Journal of Speech Communication 45(2), 171–186 (2005)
12. Hazen, T., Burianek, T., Poliforni, J., Seneff, S.: Recognition Confidence Scoring for Use in Speech Understanding Systems. In: Proceedings of ISCA ASR 2000 Tutorial and Research Workshop, Paris (2000)
13. Lee, C., Jung, S., Kim, S., Lee, G.G.: Example-based Dialog Modeling for Practical Multi-domain Dialog System. Speech Communication 51(5), 466–484 (2009)