A New Method for Testing Dialogue Systems Based on Simulations of Real-World Conditions

R. López-Cózar, A. De la Torre, J. C. Segura, A. J. Rubio, J. M. López-Soler
Dpto. Electrónica y Tecnología de Computadores, Granada University, Spain
{rlopezc,atv,segura,rubio,juanma}@ugr.es, Tel.: +34-958-243271, Fax: +34-958-243230

Abstract

This paper presents a new method for testing dialogue systems using a variety of real-world conditions simulated in lab. The method is based on the use of an additional dialogue system, called *simulator*, designed to behave as users interacting with the dialogue system to test. The behavior of the simulator is decided from diverse scenarios that represent user goals. The simulator tries to achieve the goals in the scenarios during the interaction with the dialogue system. We applied the method to test in noise conditions a dialogue system under development in our lab, considering white and babble noise. The method allowed to find out errors in the recognizer and in the strategy the system uses to handle user confirmations. Using the method, we tested the behavior of the system in terms of task completion, taking into account a VTS noise compensation technique. The experiments show that, if the VTS technique is not used, the average task completion is 31,43% for the white noise and 29,64% for the babble noise. If the VTS technique is used, the average task completion increases to 57,35% for the white noise and to 54,44% for the babble noise.

1. Introduction

Recent advances in speech recognition, speech understanding and speech synthesis have made possible to build computer systems able to communicate with human beings in restricted domains using speech. These systems are called *dialogue systems* or *conversational interfaces*. Many of them have been presented for a variety of applications, including air travel information, weather forecast, automatic call routing, car navigation, information retrieval, ticket reservation, etc. [1, 2, 3, 4, 5]. The remainder of the paper is as follows. Section 2 presents the new method proposed in this paper. The Section shows the main advantage and describes the components needed for the setting up: simulator, utterance corpus and scenario corpus. Section 3 presents the experiments. It initially describes the recognition set up (noise types, SNR values and recognition front-ends), later it describes how the system test has been carried out, and then it explains how the method has been useful to find out errors in the recognizer and the confirmation strategy. Finally, Section 4 presents the conclusions and some possibilities for future work.

2. The New Method

The *Wizard of Oz* is a very used technique to test dialogue systems that requires extensive interaction with users [6]. As testing these systems using real users is a very costly and time-consuming task, this test is not always possible in the initial development stages. In order to solve this drawback, the new method proposed in this paper allows testing the systems easily, using simulations of real-world conditions generated in lab. This test is useful to check whether the system components (e.g. speech recognizer, linguist analyzer, etc.) are robust enough [7]. It is also used to verify that the dialogue strategies have been properly designed. The method requires creating a simulator, collecting an utterance corpus concerning the system application domain, and defining a set of scenarios that represent user goals.

2.1 The simulator

The simulator is an additional dialogue system created to interact with the dialogue system to be tested. Its purpose is to behave as users interacting with the dialogue system: it receives the system prompts, analyses them and generates responses accordingly. When a user interacts with a dialogue system, it generally creates semantic representations from the user utterances to understand them. Since a semantic representation can be affected by speech recognition and understanding errors, the simulator uses this representation to check whether the system understood correctly its previous utterance. The simulator uses the system prompts to know the kind of data the system is prompting for during the conversation.

We set up a simulator for the SAPLEN dialogue system, which is under development in our lab for Spanish language, to deal with product orders and queries of fast food restaurants' clients using the telephone [8, 9]. For the setting up we considered all the system prompts and defined an action for each one that must be carried out by the simulator. The actions to define to set up the method depend on the domain the system is designed for. For example, in the ATIS domain, a typical user goal would be making a flight reservation. Then, a simulator action for this domain would be uttering the flight data items (departure and destination cities, departure time and date, etc.). Another action would be confirming all the flight data items understood by the system.

A domain-independent action is the *error recovery*, which allows the user to repair errors made by the system. Typically, these errors are due to speech understanding or speech recognition errors (insertions, deletions or substitutions). If deletion errors occur, some data items may be missing in the utterance understood by the dialogue system. To repair these errors, the user must utter again the missing data. This action is easily set up within the method proposed in this paper, since the simulator uses the data items contained in the scenario goals to answer the system prompts generated to obtain the missing data (see example in Section 2.3).

Insertion and substitution errors may be unnoticed by the dialogue system since all the expected data items may be in the understood utterance. However, the dialogue system may obtain wrong semantic representations. For example, in the ATIS domain, a user may utter *"I want to travel from Boston to Los Angeles"* and the system may understand the user wants to travel from Boston to Dallas. In order to handle these types of error, dialogue systems generally employ either explicit or implicit confirmations [10]. If an explicit confirmation is used, the user is explicitly asked to confirm both cities in his next turns. On the contrary, if an implicit confirmation is employed, both cities are uttered in the system next turns and they are considered confirmed unless the user disagrees. Both confirmation strategies are also set up easily within the method proposed in this paper. For handling explicit confirmations the simulator compares two semantic representations: (i) the one obtained by the system from the analysis of the previous utterance generated by the simulator, and (ii) a predefined correct semantic representation of the utterance. If both representations match the simulator generates a positive confirmation, and it generates a negative confirmation otherwise. For handling implicit confirmations the simulator does not generate any confirmation if both representations match, and it initiates a correction sub-dialogue otherwise.

2.2 The utterance corpus

The method uses an utterance corpus created in non-noisy conditions that is modified properly to represent the different real-world conditions, by artificially adding different types and levels of noise to the utterances. Then, the system test is carried out using the different contaminated versions of the original utterance corpus.

The corpus contains all the possible utterances (voice sample files) the simulator needs to interact with the dialogue system. Every time the simulator generates a response to answer a system prompt, it uses one of these utterances. The corpus also contains a phonetic transcription (text file) and a semantic representation for every utterance. The simulator chooses the utterance to generate a response considering its associated semantic representation, and it uses the phonetic transcriptions to create conversation log files (that contains the simulator prompts and the system prompts) to test the system. In order to build the utterance corpus for the experiments, we used a previously collected dialogue corpus that contains 523 recorded conversations between clients of a fast food restaurant and the restaurant staff. We selected randomly 264 different sentences from this corpus that were read spontaneously by nine speakers to create the utterance corpus, which contains 2,376 utterances. Table 1 sets out the sentence types used to create the utterance corpus.

Sentence type	# Utterances recorded per speaker
(a) Food order	63
(b) Drink order	60
(c) Telephone number	18
(d) Zip code	18
(e) Address	90
(f) Affirmative confirmation	5
(g) Negative confirmation	5
(h) Error indication	5
Total:	264

Table 1: Sentence types in the utterance corpus

Four of the speakers spoke standard Spanish, four spoke Spanish from southern Spain, and one speaker was a Japanese woman who speaks Spanish. The utterances were recorded under non-noisy lab conditions using a personal computer (PC), employing 16 bits/sample at 8KHz. The 264 sentences, in text format, constitute the phonetic transcriptions corresponding to the utterances, used to create the log files during the automatic generation of dialogues. In order to create the semantic representations corresponding to the utterances, we used the linguistic analyzer of the SAPLEN system, which created them automatically using the phonetic transcriptions.

2.3 The scenario corpus

The scenarios are based on the assumption that users have goals in mind when they interact with a dialogue system. So that, the system must find out the goals and carry out the corresponding actions to provide the service requested by the user. Following this scheme, each scenario includes a set of goals the simulator tries to achieve during the conversation with the dialogue system. These goals represent the goals of real users demanding the service provided by the dialogue system. A sample scenario used in the experiments is shown below.

(amount=1, food=SANDWICH, type=HAM)

(amount=1, drink=BEER, size=big)

(phone_number=958275360)

(zip_code=18001)

(id_address=STREET, name=ANDALUCIA, building_number=58, floor=FIRST, letter=E)

Figure 1: A sample scenario

The scenario defines semantically the user goals: two product orders (food and drink) and the data needed by the restaurant delivery service (phone number, zip code and address). In order to represent goals in the scenarios we used the semantic representations previously created by the linguistic analyzer of the SAPLEN system.

During a conversation with the system, the simulator focuses only on a particular scenario and tries to achieve all the goals it contains, i.e. the simulator tries to make the system understand all the data items in the goals. To do so, the simulator uses rules to select the appropriate scenario goal taking into account the system prompt, and analyses the goal to extract its data items. For instance, the first goal in the sample scenario (the food order) contains three data items: amount=1, food=SANDWICH and type=HAM. Then, if the system generates the prompt: *"How many ham sandwiches did you say"*, the simulator uses the amount data item to generate the appropriate response: *"one"*.

Selecting randomly different utterances from the utterance corpus, we created 18 different scenarios for the experiments. Every scenario contains 1-5 product orders, a phone number, a zip code and an address. The semantic representations corresponding to the utterances were included in the scenarios to represent the user goals.

3. Experiments

We used the method proposed in this paper to test the SAPLEN system and obtain an estimation of its performance under white and babble noises. The system uses an HTK (Hidden Markov Model) recognizer [11, 12] that receives as input both the user utterance and a bigram, which is determined by the dialogue manager of the system considering its current prompt [13]. The system can use up to 126 bigrams previously compiled from the utterances in the corpus: 109 bigrams can be used for the recognition of the user data (phone number, zip code and address) and the 17 remaining bigrams can be used for the recognition of product orders, queries, confirmations and corrections. The recognizer uses acoustic models trained with 4,767 sentences in the Albayzín phonetic database [14]. Table 2 summarizes the speech recognition parameters used in the experiments.

Noise Type	SNR (dB)	Recognition Front-End
White	30,2	MFCC
Babble	26,5	MFCC + VTS
	21,4	
	15,6	

Table 2: Speech recognition parameters: noise types, SNRs and recognition front-ends

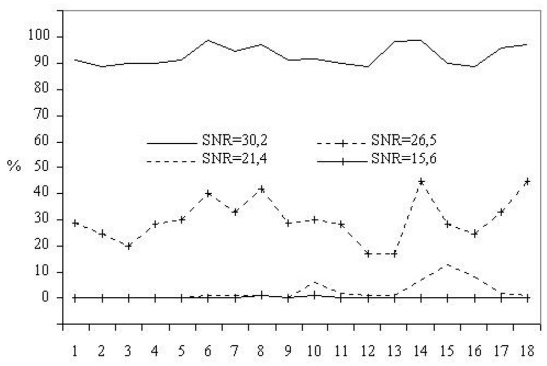
We used four levels of white and babble noises to artificially contaminate the utterance corpus created in non-noisy conditions by the nine speakers. Then, the system was tested using 2 x 4 = 8 different utterance corpora and two different recognition front-ends: (a) standard MFCC (Mel Frequency Cepstal Coefficients) parameters, and (b) standard MFCC with a VTS (Vector Taylor Series) noise compensation technique [15].

One hundred dialogues were generated for each scenario, each noise type, each SNR value and each recognition front-end, which amounts for 100 x 18 x 2 x 4 x 2 = 28,800 dialogues. The test focused on *task completion* (TC) [16]: a dialogue had task completion if the simulator made the system understand correctly all the goals defined in the scenario used to generate the dialogue. To avoid possible deadlocks during the automatic generation of dialogues, the simulator used an *interaction limit* parameter (set to 30 simulator interactions) to automatically cancel too long dialogues. Figure 3 sets out the results obtained for the 18 scenarios created for the experiments. As can be observed, the VTS technique enhances significantly TC for both the white noise (Figures 3.a and 3.b) and the babble noise (Figures 3.c and 3.d). It can also be observed that the lower SNR, the lower TC.

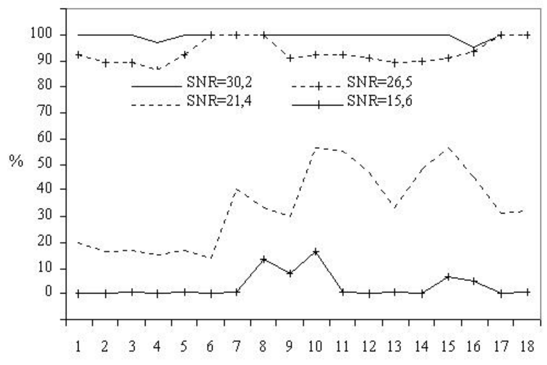
Table 3 shows the average TC considering both noise types, the 18 scenarios, the four SNR values and the two recognition front-ends. As can be observed, TC is slightly higher for both front-ends, which means that the system is more robust when it deals with the white noise

MFCC		MFCC + VTS	
White	Babble	White	Babble
31.43	29.64	57.35	54.44

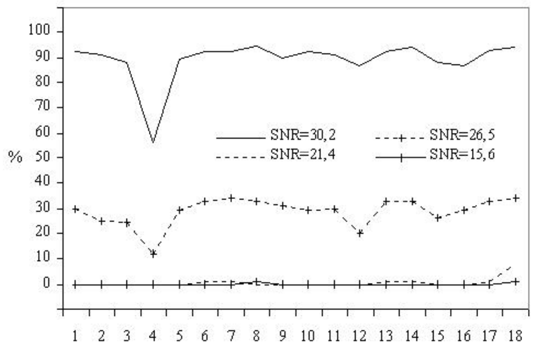
Table 3: Average TC results



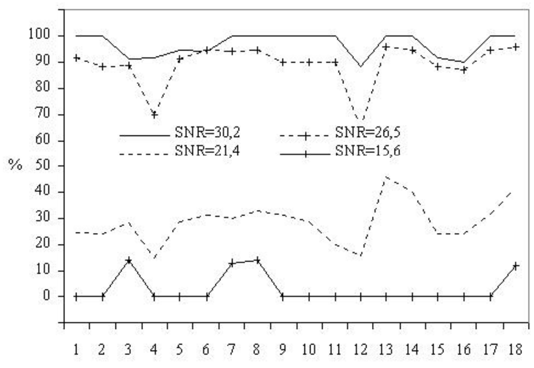
(a) White noise MFCC



(b) White noise MFCC + VTS



(c) Babble noise MFCC



(d) Babble noise MFCC + VTS

Figure 3: TC results for each scenario, each noise type, each SNR and each recognition front-end

3.1 Analysis of generated dialogues

The experiments showed that the lowest TC was achieved for scenarios 2, 3, 4, 5 and 12. In order to improve the system, we focused on these scenarios and analyzed the log files created when the simulator used them to interact with the dialogue system. The analysis allowed to find out errors in the recognizer and in the strategy employed by the system to handle user confirmations.

On the one hand, we observed that the recognizer output was often wrong for some words used in these scenarios. For example, this happened with the word *"bacon"* because this English word was incorrectly transcribed phonetically in the system dictionary, which was automatically created using a tool we previously developed for the phonetic transcription of Spanish words. The recognizer had also problems with other Spanish words used in these scenarios due to the strong Andalusian (southern Spain) accent of four speakers who were recorded to create the utterance corpus. During the automatic generation of dialogues, these problematic words were often substituted for other words. This problem caused a high rate of dialogues without TC since the goals containing such words were often misunderstood by the system.

On the other hand, we observed that the system confirmation strategy failed in occasions. For example, in some dialogues the system asked to confirm a wrongly recognized data item, the simulator answered *"no"* to refuse, but the recognizer output was *"yes"* instead of *"no"*. This type of error confused the dialogue system, as it considered the data item was confirmed by the simulator. When this problem happened, the dialogue ended without TC as a goal was not correctly understood by the system. In other occasions, the system asked to confirm a correctly recognized data item, the simulator answers *"yes"* to confirm, but the recognizer output was *"no"* instead of *"yes"*, which forced the system to employ additional turns to obtain the data item. This type of error caused that some dialogues ended without TC, as the simulator exceeded the interaction limit.

4. Conclusions and future work

This paper presented a new method for testing dialogue systems using a variety of real-world conditions simulated in lab. The method relies on the use of an additional dialogue system, called simulator, designed to behave as users interacting with the system to test. The paper described the simulator performance as well as the utterance and scenario corpora it uses to interact with the system. Using the method, a dialogue system under development in our lab was tested to check its performance under noise conditions. The method was also used to check the performance if it uses a VTS noise compensation technique. The experimental results show that: (i) the system is more robust for the white noise than for the babble noise, (ii) the performance decreases significantly when the SNR is low, and (iii) the VTS noise compensation technique is very useful to deal with noisy utterances, as it increases the average TC from 31.43 to 57.35% for the white noise, and from 29.64 to 54.44% for the babble noise.

The problems found out by analyzing the log files of the generated dialogues that achieved very low TC indicate that it is necessary to enhance the recognizer, and that a new strategy must be set up for the system to confirm data items. The analysis showed that when the SRN is low there are many recognition errors in the confirmations. As a future work, we plan to enhance the system in such a way that it can take into account the SNR to change from voice to touch-tone the input mode. Then, if the SNR is under a threshold, to confirm a data item the system would prompt *"Please use the phone keypad and press 1 to confirm or 2 to refuse"* instead of prompting *"Please say yes or no"*. Also, we plan to allow the system transferring the call to a human operator is the SNR is too low.

5. References

[1] Zue V. et al., "JUPITER: A telephone-based conversational interface for weather information", IEEE Trans. on Speech and Audio Processing; pp. 85-95, 2000

[2] Nakano et al., "Mokusei: A Telephone-Based Japanese Conversational System in the Weather Domain", Eurospeech '01, pp. 1331-1334

[3] Rahim et al., "Voice-IF: A Mixed-Initiative Spoken Dialogue System for AT&T Conference Services", Eurospeech '01, pp. 1339-1342

[4] Sasajima M. et al., "MINOS-II: A Prototype Car Navigation System with Mixed Turn Taking Dialogue", Eurospeech '01, pp. 1311-14

[5] Azzini I. et al., "First Steps Towards an Adaptive Spoken Dialogue System in Medical Domain", Eurospeech '01, pp. 1327-1330

[6] Bernsen N. O., Dybkjaer L., Heid U., "Current Practice in the Development and Evaluation of Spoken Language Dialogue Systems", Eurospeech 99, pp. 1147-1150

[7] Pellom B. et al., "The CU communicator: An architecture for dialogue systems", ICSLP 2000, pp. 723-726

[8] López-Cózar R, et al., "Evaluation of a Dialogue System Based on a Generic Model that Combines Robust Speech Understanding and Mixed-Initiative Control", LREC 2000; pp. 743-748

[9] López-Cózar R. et al., "Real-Time Performance of a Spoken Dialogue System", Journal of the Spanish Society for Natural Language Processing, pp. 169-174, 2001

[10] Souvignier B. et al., "The Thoughtful Elephant: Strategies for Spoken Dialog Systems", IEEE Trans. on Speech and Audio Processing, vol. 8, 2000, pp. 51-62

[11] Hain T. et al., "The 1998 HTK System for Transcription of Conversational Telephone Speech", ICASSP 99

[12] Woodland P. C. et al., "The 1998 HTK Broadcast News Transcription System: Development and Results", DARPA Broadcast News Workshop, 1999

[13] Niimi Y. et al., "A task-independent dialogue controller based on the extended frame-driven method", ICSLP 2000, pp. 114-117

[14] Casacuberta F. et al., "Development of Spahish Corpora for Speech Research (ALBAYZIN)", Workshop on International Cooperation and Standardization of Speech Databases and Speech I/O Assesment Methods, Chiavari, Italy; pp.26-28, 1991

[15] Moreno P. J.; 1996; Speech recognition in noisy environments; Ph Thesis, CMU

[16] Billi *et al.*, "Field trial evaluations of two different information inquiry systems", Speech Communication, 1997