Available online at www.sciencedirect.com
SCIENCE DIRECT
Knowledge-Based Systems 19 (2006) 153–163
Knowledge-Based SYSTEMS
www.elsevier.com/locate/knosys

Two-level speech recognition to enhance the performance of spoken dialogue systems

Ramón López-Cózar *, Zoraida Callejas [1]

Department of Languages and Computer Systems, Granada University, 18071 Granada, Spain

Received 24 September 2003; accepted 3 November 2005
Available online 13 December 2005

Abstract

Spoken dialogue systems can be considered knowledge-based systems designed to interact with users using speech in order to provide information or carry out simple tasks. Current systems are restricted to well-known domains that provide knowledge about the words and sentences the users will likely utter. Basically, these systems rely on an input interface comprised of speech recogniser and semantic analyser, a dialogue manager, and an output interface comprised of response generator and speech synthesiser. As an attempt to enhance the performance of the input interface, this paper proposes a technique based on a new type of speech recogniser comprised of two modules. The first one is a standard speech recogniser that receives the sentence uttered by the user and generates a graph of words. The second module analyses the graph and produces the recognised sentence using the context knowledge provided by the current prompt of the system. We evaluated the performance of two input interfaces working in a previously developed dialogue system: the original interface of the system and a new one that features the proposed technique. The experimental results show that when the sentences uttered by the users are out-of-context analysed by the new interface, the word accuracy and sentence understanding rates increase by 93.71 and 77.42% absolute, respectively, regarding the original interface. The price to pay for this clear enhancement is a little reduction in the scores when the new interface analyses sentences in-context, as they decrease by 2.05 and 3.41% absolute, respectively, in comparison with the original interface. Given that in real dialogues sentences may be out-of-context analysed, specially when they are uttered by inexperienced users, the technique can be very useful to enhance the system performance.

Keywords: Knowledge-based systems; Speech-based systems; Human–computer interaction; Dialogue systems; Automatic speech recognition; Speech understanding; Dialogue management

1. Introduction

Spoken dialogue systems can be considered knowledge-based systems developed to interact with users using speech in order to provide information or carry out a variety of simple tasks, such as travel information [14,26], language learning [7], car-driver assistance [2,4], weather information [20,31,34] and automatic call-routing [8,12], among others. Given that language is the most natural and efficient communication means for people, dialogue systems are developed to facilitate carrying out these tasks automatically, using eyes- and hands-free devices such as microphones and telephones. The initial systems were very limited regarding the user sentences and the types of task to carry out. However, the evolution of automatic speech recognition (ASR) and speech synthesis technologies in the last three decades has led to the development of sophisticated systems usable in real world conditions. Current systems offer a large potential for automation and increased functionality for telephone-based applications, allowing that users can talk more naturally, similarly as if they were talking to a human operator. Fig. 1 shows the typical structure of a current spoken dialogue system [19,21,23]. It is basically comprised of an input interface (speech recogniser and semantic analyser), an output interface (response generator and speech synthesiser), a dialogue manager between both interfaces, and some additional modules that provide knowledge to the previous modules.

The speech recogniser carries out the ASR process, i.e. receives the voice signal from the sentence uttered by the user and transforms it to a recognised sentence [3,29]. To do so, it uses acoustic models (AM) properly trained from a speech database, language models (LM) that determine the possible

* Corresponding author. Tel.: +34 958 240579; fax: +34 958 243179.
E-mail addresses: rlopezc@ugr.es (R. López-Cózar), zoraida@correo.ugr.es (Z. Callejas).
[1] Tel.: +34 958 240636; fax: +34 958 243179.

0950-7051/$ - see front matter © 2005 Elsevier B.V. All rights reserved.
doi:10.1016/j.knosys.2005.11.004

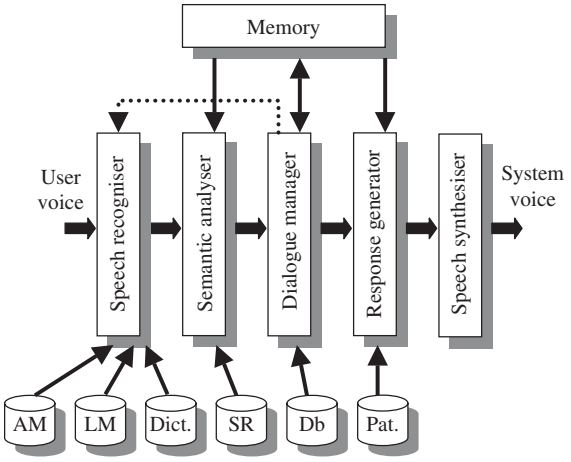

Fig. 1. Typical module structure of a spoken dialogue system.

sequences of words (sentences) and a Dictionary that contains all the possible words that can be recognised.

The semantic analyser finds out the meaning conveyed by the recognised sentences taking into account a set of semantic rules (SR) that map the syntactic and/or semantic structures found in the sentences to meaning representations, which are typically stored in the system memory in the form of semantic frames [1,5,6]. The work presented in this paper is concerned with the performance of the speech recogniser, while the experimental results shown in Section 3 are concerned with the performance of his module and the semantic analyser, given that both clearly influence the speech understanding process.

The dialogue manager implements the 'intelligence' of the system. It uses the semantic representation provided by the semantic analyser in the context of the dialogue and decides the next system response, which is typically built containing data extracted from a database (Db). The response can also be concerned with the dialogue management, such as prompting the user to confirm or rephrase the recognised sentence [13,32].

The response generator builds the system response, typically as a text sentence that must be syntactic and semantically correct [10,28]. To do so, it uses a set of patterns that contain some parts fixed and some others variable to be instantiated with the data extracted from the database. For example, the pattern '⟨company⟩ ⟨flight_id⟩ leaves at ⟨departure_time⟩ from gate ⟨gate_id⟩' can be used to generate the sentence 'American Airlines flight AA 1234 leaves at 18:30 h from gate B24'.

The speech synthesiser generates the system voice, using either a text-to-speech (TTS) conversion in the case of previously created text sentences [25], or playing pre-recorded segments (words and sentences). The TTS conversion is preferred when the vocabulary is very large, is unknown a priori or changes frequently. The recorded segments are used in small and fixed vocabulary applications since they generally provide more intelligibility, although some current TTS systems also provide excellent results.

The memory module supplies the dialogue context and historic information to several modules of the system. The semantic analyser uses contextual information to resolve possible anaphoric references in the user sentence (e.g. in the sentence 'I want the first flight', find the referent of 'first'). The dialogue manager uses the historic information as a record of previous system and/or user actions, which can be useful to make the system behaviour more intelligent. For example, using this information the system can notice continuous misunderstanding of the user sentence and thus transfer the call to a human operator. The contextual information is also very useful for the response generator since it can make the system responses more human-like. Taking it into account the system can decide the use of anaphora (pronouns instead of nouns) and ellipsis (omission of unnecessary words) as humans do when uttering sentences within a context.

In despite of the advances made in the last years in terms of ASR and speech synthesis, spoken dialogue systems are still restricted to well-known application domains that provide very valuable knowledge about the words and sentences the users may likely utter. For example, in the Air Travel Information Service (ATIS) some likely words are airport and city names as well as travel dates, types, duration and fares. The domain knowledge is very important to create the system dictionary, since a word not included in it (called Out-Of-Vocabulary word) can never be recognised, and thus causes recognition errors (it can be either changed by another acoustically similar or discarded). The application domain also provides knowledge about the task the system must carry out. For example, this knowledge makes the system ask the user for the departure city if only the destination city was provided in a query to travel from one city to another.

1.1. Stochastic approach to ASR: acoustic and language knowledge

Several approaches have been developed to face the ASR problem, such as expert systems and artificial neural networks [24]. This problem can be stated as follows: 'find the sequence of words uttered W, given a sequence of acoustic data A'. The technique presented in this paper is concerned with the stochastic approach, which is the one mostly used nowadays. According to this approach, W is the word sequence with the highest probability given the acoustic data, as shown in the following expression:

$$W = \max_{W} P(W|A) \tag{1}$$

Since it is not easy to calculate $P(W|A)$, the Bayes rule is used to ease the computation leading to the expression:

$$P(W|A) = \frac{P(A|W)P(W)}{P(A)} \tag{2}$$

Note that in this expression, the denominator is not necessary to compute $P(W|A)$ since it is independent of the word sequence W. Thus, Eq. (1) can be rewritten as follows

$$W = \max_{W} P(A|W)P(W) \tag{3}$$

which is the fundamental equation in the stochastic approach to the ASR problem. In this expression $P(A|W)$ is called

the acoustic model, while $P(W)$ is called the language model. The acoustic model represents knowledge about the pronunciation of speech units (e.g. phonemes). To model these units current ASR systems mostly employ Hidden Markov Models [11] which are trained for the speech units considered.

As said above, the domain knowledge is very important to decide the sentences the ASR system must deal with. $P(W)$ represents these sentences. To set up this model, current ASR systems typically use either linguistic knowledge in terms of finite-state grammars, or statistical knowledge in terms of probabilities of uttering words given previously uttered words. In the first case, grammar rules indicate the possible sentences that can be recognised. For instance, the JSGF (Java Speech Grammar Format) grammar shown in Fig. 2 indicates that a possible sentence is 'please open the file and the window'.

In the case of using statistical knowledge, the ASR system must find the probability of a sentence W that contains q words $(w_1,\ldots,w_q)$. To do so it employs the following expression:

$$P(W) = P(w_1, w_2, \ldots, w_q) = P(w_1)P(w_2|w_1)P(w_3|w_1w_2)\cdots P(w_q|w_1\cdots w_{q-1}) \quad (4)$$

Since calculating this probability is very difficult in practice, it is common to use histories of n words called n-grammars. This way, the probability of a word w_i is estimated taking into account the n preceding words and assuming statistical independence from the oldest part of the sentence. This method is called Markov assumption and can be expressed mathematically as follows:

$$P(w_i|w_1, w_2, \ldots, w_{i-1}) = P(w_i|w_{i-n+1}\cdots w_{i-1}) \quad (5)$$

Generally $n=2$, in which case the grammar is called bigram or first-order Markov model, or $n=3$, in which case it is called trigram or second-order Markov model. If the grammar is a bigram, computing the probability of a word w_i only requires considering the previous word, and thus the probability of a sentence can be expressed as follows:

$$P(W) = P(w_1, w_2, \ldots, w_q) = P(w_1)P(w_2|w_1)P(w_3|w_2)\ldots P(w_q|w_{q-1}) \quad (6)$$

In order to get the knowledge for building the acoustic and language models it is usual to use speech databases and sentence corpora (in text format) concerned with the application domain. The databases provide information about the pronunciation of the basic speech units (e.g. phonemes), which is used to train the acoustic models. The sentence corpora provide information about the sentence types the speech recogniser must deal with.

```
#JSGF v1.0
// Define the grammar name
grammar SimpleCommands;
// Define the rules
public <Command> = [<Polite>] <Action> <Object> (and <Object>)*;
<Action> = open | close | delete ;
<Object> = the window | the file;
<Polite> = please;
```

Fig. 2. JSGF grammar for ASR.

The remainder of the paper is organised as follows. Section 2 describes the structure of the proposed two-level speech recogniser, and discusses how to set up association between the prompts of a dialogue system and word-class pairs, which represent linguistic knowledge. It also explains algorithmically how to set up the procedure for analysing word graphs. Section 3 describes the corpora used in the experiments, and explains the differences between the original input interface of the experimental dialogue system, and a new interface, which is implemented using the proposed speech recogniser. Experimental results obtained with both interfaces for the so-called *in-context* and *out-of-context* sentence analysis is presented, and limitations of the proposed recogniser are discussed. Finally, Section 4 presents the conclusions and shows some possibilities for future work.

2. The two-level speech recogniser

In order to enhance ASR in spoken dialogue systems, many systems recognise the user sentence using a grammar associated with the current state of the dialogue. This state is determined by the prompt generated by the system. This approach to ASR is called based on prompt-dependent grammars in this paper because each prompt decides the set of sentences that can be recognised next. Visweswariah and Prints [30] observe that when a user converses with a dialogue system the state of the dialogue strongly influences the responses expected from the user. Moreover, the prompts generated by the system play an important role with respect to the language model used to recognise the sentences. The authors report that using knowledge about the dialogue state, the word error rate can be reduced by about 9%. This approach is appropriate in some cases (e.g. when the system restricts the possible user sentences) but fails if the user utters sentences not related to the current state, i.e. not permitted by the used grammar. For example, if the system generates the prompt 'Please say your telephone number' and the user utters a telephone number, the sentence can be correctly recognised. However, if s/he utters a different sentence type (e.g. an address) the recogniser output will be any of the possible telephone numbers and the address will never be recognised. Consequently, the user may feel uncomfortable using the system, as s/he may perceive that any deviation from the system prompts provokes the system malfunction.

As an attempt to solve this problem, some systems follow a different approach based on using a unique grammar to recognise any sentence uttered by the user, independent of the current prompt. We call this type of grammar G-grammar in this paper. A drawback of this approach is that this grammar tends to be much more perplex than the prompt-dependent grammars, and the vocabulary tends to be notably greater, which tends to increase the recognition errors. The technique presented in this paper combines the advantages of both approaches using a new type of speech recogniser, which we call two-level speech recogniser. This recogniser uses on the one hand a G-grammar to allow

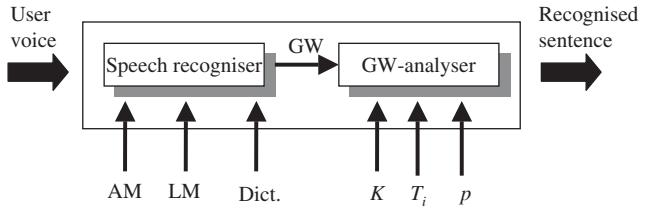

Fig. 3. Two-level speech recogniser.

recognising, in theory, any kind of sentence permitted in the application domain, independent of the current prompt generated by the system. This feature enables that the user can utter any kind of sentence at any time in the dialogue, since all the possible sentences considered in the application domain are permitted by the G-grammar. For example, in the ATIS domain a user could answer the system prompt 'Destination city?' with the sentence 'I will leave on Sunday', and the prompt 'Did you say you want to fly to Rome?' with the sentence 'I said I want to fly to Bonn'. On the other hand, in order to enhance the word recognition rate, the technique we propose uses information about the current dialogue state to favour recognising the most likely words given the state, without discarding other less likely words. Hence, if the prompt is 'Destination city?' the technique favours recognising sentences such as 'The destination city is Rome' and 'I must fly to Madrid', while if it is 'Did you way say you want to fly to Rome?' the technique favours recognising sentences such as 'Yes, that's correct', 'Yeah, it's right' and 'Sure, that's it'. The method to empower recognising the most likely sentences given a dialogue state is described in Section 2.2.

The structure of the two-level speech recogniser is shown in Fig. 3. The first module of the recogniser is a standard speech recogniser that receives the voice signal from the user sentence and produces as output a graph of words (GW). To do so it uses previously trained acoustic models (AM) from a speech database, language models (LM) (concretely word bigrams) compiled from a sentence corpus regarding an application domain (e.g. ATIS), and the application dictionary (i.e. set of words that can be recognised).

A GW is a network constituted of a set of nodes and a set of arcs, in which the nodes represent words and the arcs represent transitions between words. Fig. 4 shows a simple GW that allows recognising the sentences 'two small beers' and 'three small beers' in the fast food application domain, which is considered in the experiments. This GW has eight nodes: two null, start and final, and four for the words 'two', 'three', 'small' and 'beers'. Each arc has an acoustic probability (called 'a' in Fig. 4) and a language probability (called 'l' in the figure). Both probabilities are provided by the standard speech recogniser and used by the technique we propose (parameters p_a and p_l in Fig. 5) to compute the transition probability as the sum of both probabilities.

The second module of the two-level speech recogniser is called GW-analyser. It receives the GW generated by the first module, analyses it and outputs a recognised sentence according to the highest probability path in the GW. The analysis is carried out using three parameters: set of all the word-classes considered (K), current prompt type of the dialogue system (T_i) and probability increment (p). The latter is used to increment the transition probabilities in the GW following the procedure explained in Section 2.2.

2.1. Setup of associations between prompt types and sets of word-class pairs

The technique presented in this paper proposes to create a Ω set comprised of associations between prompt types T_i of a dialogue system and linguistic knowledge in the form of sets of word-class pairs C_i:

$$\Omega = \{T_i, C_i\}, \quad i = 1, \ldots, n$$

A set of word-class pairs C_i is defined as $C_i = (W_m, W_n)$, where W_m and W_n are word-classes as for example numbers ('zero', 'one', 'two', etc.), street names ('Elm', 'Arlington', etc.) and city names ('New York', 'Boston', etc.). The prompt types are known a priori and the sets of word-class pairs, which are initially empty, are created according to the following procedure carried out in three phases. In the first, a corpus of user-system dialogues regarding the application domain is analysed to know the sentence types F_i the users utter to answer the prompt types T_i, and associate each F_i to its corresponding T_i. For example, a dialogue system designed for the fast food domain may generate the prompt types: T_1 = 'Product order' (e.g. 'What would you like to order?'), T_2 = 'Telephone number' (e.g. 'Please say your telephone number'), T_3 = 'Address' (e.g. 'What is your address?') and T_4 = 'Confirmation' (e.g. 'Did you say a ham sandwich?'), among others. In response to these prompt types the users (restaurant clients) may utter the following sentence types: F_1 = 'Fast food orders' (e.g. 'I want a ham sandwich'), F_2 = 'Telephone numbers'

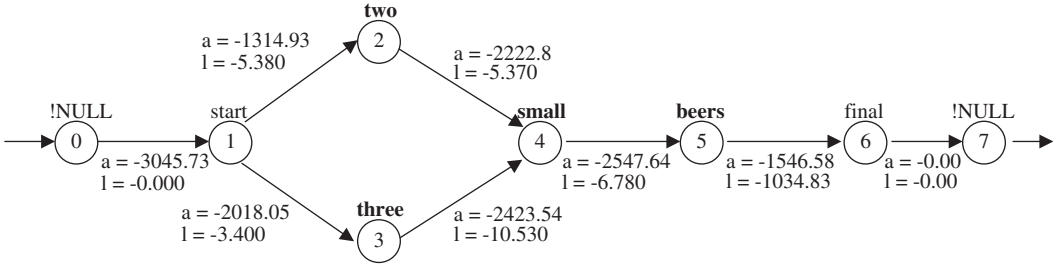

Fig. 4. Simple graph of words.

```
GWanalysis ( Input: GW, K, Ti , p ; Output: sentence)
{
    /* The following function processes the GW and creates a set that contains all its transitions. Each transition t has
    associated an acoustic probability pa and a language probability pl */

    transitions = process(GW)

    /* In order to create the best path in the GW, the procedure initialiseSearchSpace() adds two additional fields to each graph
    node: prob and previousWord. The prob field represents the accumulated probability of the word, which is initialised using
    a special value (MIN_VALUE) representing that the word does not have a probability assigned yet. The previousWord field
    stores a pointer to the previous word in the best path */

    initialiseSearchSpace()

    Ci = Ω ( Ti )                                          // Find Ci associated with Ti

    // The following loop creates the best path in the GW visiting all the transitions

    foreach t ∈ transitions do

      wS = transitionStartNode( t ).word
      wE = transitionEndNode( t ).word
      pa = acousticProbability( t )
      pl = languageProbability( t )
      pS = transitionStartNode( t ).prob
      increment = 0.0
      ∏ = wordClassesContaining( wS , K )
      ∑ = wordClassesContaining( wE , K )

      /* The following loop finds all the word-class pairs that contain the words wS and wE . If a pair is found in Ci the value of
      p is used to increment the transition probability */

      foreach Wm in ∏                                      // Wm word-class that contains wS
                foreach Wn in ∑                            // Wn word-class that contains wE
                      if ( Wm , Wn ) is in Ci then
                            increment = p
                            break
                      endif
                endfor
      endfor

      if ( transitionEndNode( wE ) = = MIN_VALUE ) then
                new_prob = pa + pl + increment
                setProb( wE , new_prob )
                setPreviousWord (wE , wS )               // create transition wS → wE in best path
      else
                new_prob = pa + pl + pS + increment
                if ( new_prob > currentProb( wE ) ) then
                      setProb( wE ) = new_prob
                      setPreviousWord (wE , wS )         // change transition wS → wE in best path
                endif
      endif
    endfor

    /* Now the procedure locates the word with the highest accumulated probability and follows back the created path until
    reaching a node that contains the special word !NULL. All the words in this backtracking are stored in a stack structure to
    get the recognised words in the correct order */

    w = highestAccumulatedProbabilityWord()

    while (w <> !NULL) do
      pushStack(w)
      w = previousWordInPath(w)
    enddo

    // Finally all the words in the stack are popped out to get the recognised sentence

    sentence = ""                                          // string initialisation

    while (not emptyStack()) do
      w = popStack()
      sentence = sentence + w + " "
    enddo
}
```

Fig. 5. Algorithm for analysing graph of words.

(e.g. '9 5 8 1 2 3 4 5 6'), F_3 = 'User addresses' (e.g. 'Arlington road, 54') and F_4 = 'Confirmations' (e.g. 'Yes'). Then, in the first phase F_1 is associated with T_1, F_2 with T_2 and so on.

In the second phase, the sentences of each sentence type F_i are analysed to know the words that are really necessary to obtain the sentence meanings (usually called keywords) and group them into word-classes W_1, W_2, W_3,... For example, in the fast food domain it would be possible to create the following six word-classes: W_1: AMOUNT = {'a', 'one', 'two', 'three', 'four', ...}, W_2: FOOD = {'sandwich', 'sandwiches', 'cake', 'cakes', 'salad', 'salads', ...}, W_3: INGREDIENT = {'ham', 'cheese', 'bacon', 'chocolate', ...}, W_4: DRINK = {'water', 'beer', 'beers', 'wine', 'cola', 'colas', 'milkshake', 'milkshakes', ...}, W_5: SIZE = {'small', 'big', 'medium', ...} and W_6: TASTE = {'orange', 'lemon', 'apple', 'chocolate', ...}.

The third phase is carried out in two steps. In the first, each sentence U of type F_i is analysed and transformed to a sentence U' in which every keyword is substituted by the word-class that contains[2] the keyword. In the second step each U' is analysed and a pair (W_m, W_n) is created with each word-class pair that appears consecutively in U'. Each created pair is added to the set C_i associated with the prompt type T_i (if it is not yet included). For example, let us suppose a dialogue system designed for the fast food domain generates a prompt of type T_1 = 'Product order' (e.g. 'What would you like to have?') and that sentences of type F_1 are the following: 'please one ham sandwich', 'two small beers' and 'a chocolate milkshake'. C_1 is the set of word-class pairs to create from these sentences. Thus, taking into account the sample word-classes shown above, the three sentences are transformed obtaining the following: 'please AMOUNT INGREDIENT FOOD', 'AMOUNT SIZE DRINK', 'AMOUNT INGREDIENT DRINK' and 'AMOUNT TASTE DRINK'[3]. The set of word-class pairs obtained is C_1 = { (AMOUNT, INGREDIENT), (INGREDIENT, FOOD), (AMOUNT, SIZE), (SIZE, DRINK), (AMOUNT, INGREDIENT), (INGREDIENT, DRINK), (AMOUNT, TASTE), (TASTE, DRINK) }.

2.2. Procedure for analysing graphs of words

The procedure for analysing GWs can be described algorithmically as shown in Fig. 5. The idea is to increment the probabilities of the most likely sentences taking into account the prompt type after which each sentence is uttered by the user. For example, if the system prompts for the telephone number it is very likely the user utters a telephone number. In this case the procedure increases the transition probabilities between words in the word-class NUMBER (e.g. 'zero', 'one', 'two', etc.) if they are found in the GW generated from the user response. The GW-analyser uses the p parameter to increment the probability of the transition $w_S \rightarrow w_E$ if the word-class pair (W_m, W_n) [4] is in the C_i set associated with the prompt type T_i. For example, if the system generates the prompt type T_1 = 'Product order' and in the analysis of the GW obtained from the recognised sentence uttered to answer this prompt the GW-analyser finds the transition 'one' → 'ham', then the probability of this transition is incremented by the value of the p parameter, given that 'one' is in AMOUNT, 'ham' is in INGREDIENT and (AMOUNT, INGREDIENT) is in C_1.

3. Experiments

The goal of the experiments is to evaluate the performance of a new input interface that uses the two-level speech recogniser. This interface is setup in the SAPLEN dialogue system, previously developed in our lab to deal with Spanish telephone-based orders and queries of fast food restaurants' clients [16]. This interface consists of the two-level speech recogniser and the same semantic analyser used in the original input interface of the system. The first module of the two-level speech recogniser (see Fig. 3) is the same HTK-based speech recogniser [33] used in the original interface, but slightly modified to generate as output GWs instead of recognised sentences.

The second module is a GW-analyser that works as explained in Section 2.2. The evaluation is carried out using two metrics: word accuracy (WA) and sentence understanding (SU) [15,22]. WA is calculated as follows: $WA = (w_t - w_i - w_s - w_d)/w_t$, where w_t is the total number of words in the input sentences, whereas w_i, w_s and w_d are the number of words inserted, substituted and deleted by the recogniser, respectively, due to recognition errors. SU is calculated as $SU = S_u/S_t$, where S_u is the number of input sentences correctly analysed by the semantic analyser, and S_t is the total number of sentences.

3.1. Description of the corpora used in the experiments

In order to develop the SAPLEN system, we previously collected a dialogue corpus in a fast food restaurant that contains about 800 recorded dialogues in Spanish regarding telephone conversations between clients and restaurant assistants. These dialogues contain product orders, telephone numbers, post-codes, addresses, queries, confirmations, greetings and other types of sentence. In order to be used for previous work e.g. [16–18], the dialogues were transcribed and analysed to include tags regarding the speakers, sentence types, pragmatic function of sentences and other kind of information [9]. To obtain the experimental results presented in this paper we created two corpora of spoken sentences, one for training and another for testing, selecting at random 5250 client

[2] If a keyword is in several word-classes (e.g. the keyword 'chocolate' is in W_3 and W_6) then several U's are obtained from the same U, using a different word-class to create each U'.

[3] The sentences 'AMOUNT INGREDIENT FOOD' and 'AMOUNT TASTE DRINK' are created because the 'chocolate' keyword is in two word-classes (INGREDIENT and TASTE).

[4] (W_m, W_n) is any word-class pair in C_i verifying that w_S is in W_m and w_E is in W_n.

Table 1
Sentence types used in the experiments

Type	Sentence description	No. of sentences
F_1	Product order	500
F_2	Telephone number	500
F_3	Post-code	500
F_4	Address	500
F_5	Query	250
F_6	Confirmation	250
F_7	Amount	250
F_8	Food name	250
F_9	Ingredient	250
F_{10}	Drink name	250
F_{11}	Size	250
F_{12}	Taste	250
F_{13}	Temperature	250
F_{14}	Street name	250
F_{15}	Building number	250
F_{16}	Apartment floor	250
F_{17}	Apartment letter	250

sentences among the 17 sentence types shown in Table 1. No training sentences were included in the test corpus.

The test corpus was used to evaluate both the original and the new input interface of the SAPLEN system. It contains 1000 spoken sentences, 250 of each of the first four sentence types shown in Table 1 (i.e. 250 product orders, 250 telephone numbers, 250 post-codes and 250 addresses). The training corpus was used to create the Ω set of associations between prompt types T_i and word-class pairs C_i used by the new input interface (following the procedure described in Section 2.1). This corpus contains 4250 spoken sentences, 250 sentences of each type shown in Table 1.

We also included in both corpora the orthographic transcriptions (sentences in text format) corresponding to the spoken sentences, as well as their corresponding semantic representations (frames). The word-classes for the words in these sentences were created for the previous work, analysing manually the orthographic transcriptions. Table 2 sets out some of the word-classes obtained (translated from Spanish to English). Using the orthographic transcriptions we compiled a G-grammar (word bigram) that was used by the two-level speech recogniser to obtain the recognised sentences.

Table 2
Examples of word-classes

Word-class	Different words in word-class	Sample words
Number	103	Fourteen, twenty, …
Food_name	6	Sandwich, cake, …
Ingredient	28	Ham, cheese, …
Drink_name	16	Beer, wine, …
Size	6	Small, large, …
Taste	6	Orange, lemon, …
Temperature	6	Cold, hot, …
Address_ type	5	Street, square, …
Street_name	324	Elm, Melrose, …
Building_floor	20	First, second, …
Apartment_ letter	28	a, b, c, d, e, …

Table 3
Sample sets of word-class pairs (translated from Spanish to English)

Pair set	Pair set description
C_1	(Number, indredient)
	(Ingredient, food)
	(Number, ingredient)
	(Number, drink)
	(Number, size)
	(Number, taste)
	(Taste, size)
	(Size, taste)
	(Taste, temperature)
	(Size, temperature)
C_2	(Number, number)
C_3	(Number, number)
C_4	(Address_name, address_type)
	(Address_type, number)
	(Number, building_floor)
	(Building_floor, appartment_ letter)

To create the Ω set the 4250 training sentences were automatically analysed using the word-classes. As result of the analysis, for each set F_i in Table 1 we obtained a set C_i that we associated with the corresponding prompt type T_i the SAPLEN system generates. Table 3 shows some sample sets of word-class pairs, while Table 4 sets out sample associations of prompt types and sets of word-class pairs (translated from Spanish to English).

The first four sentence types shown in Table 1 were also used to create four word bigrams to evaluate the original input interface of the system. Each bigram was compiled from half of the sentences of each type (i.e. a bigram was compiled from 250 product orders, other from 250 telephone numbers, etc.) while the other half was used for testing.

3.2. Differences between the original and the new input interface

The original input interface of the SAPLEN system was designed to use 17 prompt-dependent grammars for ASR (word bigrams). Each grammar was compiled using a particular sentence type (e.g. a grammar was compiled using product orders, other using telephone numbers, etc.). Taking into account the current dialogue state, the system dialogue manager uses the parameter prompt_T to decide the bigram to use for recognising each sentence. For example, when the system generates the prompt 'Please say your telephone number', the dialogue manager sets prompt_T = 'Telephone number' and then the speech recogniser uses the grammar compiled from telephone numbers.

Table 4
Sample associations of prompt types (T_i) and sets of word-class pairs (C_i)

T_i	C_i	Sample sentence
T_1 = Product order	C_1	A ham sandwich
T_2 = Telephone number	C_2	9 5 8 1 3 2 4 1 5
T_3 = Post-code	C_3	1 8 0 1 4
T_4 = Address	C_4	Elm street 23 first a

Table 5
Average WA and SU results using prompt-dependent grammars

Sentence type F_i	Prompt type T_i	WA	SU
Product order	Product order	93.39	94.36
	Telephone number	0.1	0
	Confirmation	−0.13	0
Telephone number	Telephone number	94.36	92.61
	Post-code	−37.3	0
	Confirmation	−0.33	0
Post-code	Post-code	94.82	91.49
	Address	−0.5	0
	Confirmation	−0.15	0
Address	Address	96.3	85.66
	Post-code	−0.03	0
	Confirmation	−0.21	0

The new input interface uses a G-grammar instead of several prompt-dependent grammars, and three parameters instead of just one: (i) sentence_T [5] to indicate the sentence type to analyse, (ii) prompt_T to indicate the prompt type supposedly generated by the system given that the input interface analyses a sentence of type sentence_T, and (iii) probability increment (p) to indicate how much the GW-analyser must increment transition probabilities during the GW analysis. This parameter allows evaluating the effect of the proposed technique in the sentence analysis. If $p=0$ the knowledge concerned with the word-class pairs is not used in the GW analysis. In this case, the GW-analyser behaves just like a standard Viterbi recogniser [24]. On the contrary, when $p>0$ the GW transitions are incremented following the procedure described in Section 2.2. We tested several values for this parameter (0, 1, 2, 3, …) until noticeable differences in the experimental results were obtained (19 values were tested in total).

3.3. Experimental results

3.3.1. Prompt-dependent grammars

First, we tested the original input interface of the SAPLEN system. Table 5 sets out the average WA and SU results obtained when the product orders, telephone numbers, post-codes and addresses in the test corpus (1000 sentences in total) were in-context analysed (i.e. the sentence type F_i was the same as the prompt type T_i) and out-of-context analysed (i.e. the sentence type was different to the prompt type). In the out-of-context analysis the sentences were analysed using a grammar compiled from a different sentence type (e.g. product orders were analysed using a grammar compiled from telephone numbers).

As can be observed in the table, the performance of the input interface is acceptable when it analysed sentences in-context. However, the performance is totally unacceptable for the out-of-context analysis since the sentences were not correctly recognised and consequently were not understood. Almost all WA scores are even negative due to the high rate of insertion recognition errors. The outputs of the HTK-based recogniser were the sentences permitted by the used grammar. Hence, if for example the grammar was compiled from telephone numbers, the outputs were telephone numbers independent of the sentence types analysed.

3.3.2. Two-level recogniser for in-context analysis

Second, we tested the performance of the new input interface when the sentences in the test corpus were in-context analysed, considering the 19 values of the p parameter. Table 6 sets out the average WA and SU results obtained.

It can be observed that the lowest results were obtained when $p=0$ since in this case the transition probabilities in the GWs were not incremented, and then the knowledge provided by the word-class pairs was not used. The results increase with the value of p until $p=13$, and from this point they decrease. Thus, 13 is the best value of the parameter for the sentence corpus used. When $p<13$ the GW-analyser did not get enough benefit from the knowledge provided by the word-class pairs. This fact is easily observed from the trace files generated when the sentences were analysed, as there were many word substitutions in the recognition process. For example, the word 'sí' ('yes') was often substituted by the words 'seis' ('six') or 'sin' ('without'), the word 'veintitrés' ('twenty three') was often substituted by the words 'verde tres' ('green three'), the word 'cero' ('zero') was often substituted by the word 'pero' ('but'), etc. These substitutions occurred because the words sound very similarly in Spanish and the p parameter had a value that was not high enough as to correct the wrong transitions in the GWs.

On the contrary, when $p>13$ the GW-analyser incremented excessively the probability transitions in the GWs, causing a

Table 6
Average WA and SU results for the two-level speech recogniser (in-context sentence analysis)

p	WA	SU
0	88.57	83.43
1	88.97	83.75
2	89.45	84.53
3	89.82	85.09
4	90.22	85.60
5	90.54	85.95
6	90.87	86.34
7	91.16	86.60
8	91.39	86.83
9	91.61	86.85
10	91.89	87.17
11	92.19	87.41
12	92.42	87.50
13	92.66	87.62
14	92.37	87.19
15	92.03	86.74
16	91.75	86.38
17	91.35	85.59
18	91.05	85.24

[5] This parameter is only used in the experiments since in real dialogues it is not possible to be certain, a priori, about the sentence type the system will analyse at every dialogue state, given the wide range of potential users.

distortion in the analysis. The trace files show there were many word insertions, which tended to follow the syntactic structure of the word-class pairs. Also, meaningless words (that were not included in the word-classes) were often substituted by keywords (that were included). For example, in many occasions the word 'pero' ('but') which is not a keyword was substituted by the word 'queso' ('cheese') which is a keyword. When the best performance was achieved ($p=13$) the technique we propose allowed incrementing WA by 4.09% absolute, from 88.57 ($p=0$) to 92.66% ($p=13$), and SU by 4.19 absolute, from 83.43 ($p=0$) to 87.62% ($p=13$).

3.3.3. Two-level speech recogniser for out-of-context analysis

Finally, we assessed the performance of the new input interface for the eight cases of out-of-context sentence analysis shown in Table 7, which may occur in real dialogues when the users try to correct system errors. Following the dialogue strategy implemented in the dialogue manager, the SAPLEN system initially prompts the user to order products, then prompts for his/her telephone number and finally (if the telephone number is not in the user database) prompts for his/her post-code and address. After obtaining each data item, the system generates an explicit confirmation if has low confidence on the data recognition (e.g. 'Did you say your telephone number is 9 5 8 1 2 3 4 5 6?'), and includes an implicit confirmation in the prompt to get the following data item if the previous item was not explicitly confirmed (e.g. 'Ok, telephone number 9 5 8 1 2 3 4 5 6. What is your post-code?'). Taking into account this dialogue strategy, it is possible the system misunderstands the order and the user tries to correct the error, rephrasing the order (e.g. 'I said a ham sandwich') when the system prompts for a yes/no confirmation (case 2 in Table 7). Similarly, the system may misunderstand the telephone number and the user may try to correct the error, rephrasing the telephone number (e.g. 'I said my telephone number is 9 5 8 1 2 3 4 5 6') when the system prompts for the post-code (case 3 in Table 7).

For the out-of-context sentence analysis, the output of the original speech recogniser of the SAPLEN system is completely wrong. However, using the two-level speech recogniser many sentences are correctly analysed. Table 8 shows the average WA and SU results obtained for the eight out-of-context analysis types and the 19 values of the p parameter considered.

Table 7
Cases of out-of-context sentence analysis

Case	Sentence type F_i	Prompt type T_i
1	Product order	Telephone number
2	Product order	Confirmation
3	Telephone number	Post-code
4	Telephone number	Confirmation
5	Post-code	Address
6	Post-code	Confirmation
7	Address	Post-code
8	Address	Confirmation

Table 8
Average WA and SU results for the two-level speech recogniser (out-of-context sentence analysis)

p	WA	SU
0	89.21	78.27
1	89.19	78.27
2	89.18	78.27
3	89.17	78.22
4	89.17	78.17
5	89.16	78.13
6	89.13	78.12
7	89.08	78.09
8	89.07	78.08
9	89.06	78.08
10	89.03	77.95
11	88.97	77.95
12	88.91	77.70
13	88.90	77.42
14	88.89	77.32
15	88.80	77.13
16	88.72	77.09
17	88.64	76.85
18	88.55	76.53

It can be observed that the trend is different to the one observed for the in-context analysis (Table 6), as the best WA and SU scores were achieved when $p=0$ and they decrease slightly as p increases. The reason is that in all the out-of-context cases studied but in the third one, the GW-analyser incremented the probabilities of wrong transitions. For example, in the first case it incremented the probabilities of transitions between numbers instead of incrementing the probabilities of the transitions between ingredients and food names. On the contrary, in the third case it incremented the probabilities of right transitions (between numbers) given that the C_3 set (see Tables 3 and 4) used for the 'post-code' prompt includes the word-class pair (NUMBER, NUMBER). Thus, in this out-of-context case sentences were analysed just like if they were in-context analysed.

In real dialogues it is not possible to know in advance whether a sentence would be either in-context or out-of-context analysed as it depends on the user behaviour. However, it is reasonable assuming that users may follow the system indications most of the times (e.g. they may utter a telephone number if the system prompts for a telephone number) and may behave differently occasionally (e.g. when they try to correct system errors). From this assumption and the in-context experimental results shown in Table 6 follows that the technique proposed is this paper should be used with $p=13$, for both in-context and out-of-context sentence analysis.

To compare the results obtained for both input interfaces, Table 9 shows the average WA and SU results obtained when $p=13$ for the same eight cases of out-of-context analysis considered (shown in Table 7), as well as the average results obtained for the in-context analysis of the first four sentence types shown in Table 1 (product orders, telephone numbers, post-codes and addresses).

As can be observed, the best results were obtained for the in-context analysis given that in the out-of-context analysis some

Table 9
Average WA and SU results for the two-level speech recogniser in both in-context and out-of-context sentence analysis ($p=13$)

Sentence type F_i	Prompt type T_i	WA	SU
Product order	Product order	90.48	88.88
	Telephone number	87.9	80.42
	Confirmation	84.67	79.23
Telephone number	Telephone number	93.57	87.7
	Post-code	93.57	87.7
	Confirmation	91.86	80.17
Post-code	Post-code	94.12	89.32
	Address	91.4	77.79
	Confirmation	91.62	77.9
Address	Address	92.49	84.6
	Post-code	85.61	68.29
	Confirmation	84.62	67.9

transitions in the GWs were wrongly incremented. Although the results obtained for the out-of-context analysis were not excellent, they are much better than those obtained when the prompt-dependent grammars were used for this kind of analysis (compare Tables 5 and 9).

3.4. Limitations of the technique proposed in this paper

A major limitation of the technique we present in this paper is that the word-class pairs can only capture the short-distance context dependency within a 2-word window. However, many of the context dependencies in natural language occur beyond such a window [27]. Another disadvantage is that these pairs do not take into account the relationships between particular words in the word-classes, which can be very important for languages that have gender and number correspondences. For example, let us suppose the words 'one' and 'two' are in the word-class AMOUNT. The word-class pairs mapped to the prompt type T_1 = 'Product order' can be used to increase the transition probabilities of the sentence 'one ham sandwiches', which is grammatically incorrect as number correspondence is not observed. Such linguistic knowledge concerning correspondence would be very useful to reduce word error rate in ASR. Finally, the technique does not consider any knowledge to filter out sentences that have no meaning in the domain. For example, the sentences 'one chocolate sandwich' and 'one ham cake' have no meaning in the domain as these products do not exist in the system database. As the current setup does not take into account semantic knowledge, the two-level speech recogniser increments the transition probabilities of these wrong sentences.

4. Conclusions and future work

The experimental results show that the technique proposed in this paper enhances notably the performance of the new input interface of the SAPLEN system, which uses a G-grammar and the two-level speech recogniser. They also show the knowledge concerning word-classes is very important when analysing GWs and the value of the p parameter clearly influences the analysis. Table 6 shows that for the in-context analysis the average WA and SU scores increase as the value of p increases, since the GW-analyser increments transition probabilities correctly. However, an excessive value (greater than 13 for the sentence corpus used) causes a distortion in the GWs analysis that provokes a performance reduction. Table 8 shows that for the out-of-context analysis the trend is different. Average WA and SU scores decrease as the value of p increases since the GW-analyser increments transition probabilities incorrectly (except for the third out-of-context case).

A comparison of the out-of-context results presented in Tables 5 and 9 show that when the two-level speech recogniser carries out the out-of-context analysis using the best p value for the in-context analysis ($p=13$), WA increments by 93.71% absolute, from −4.81 (Table 5) to 88.90% (Table 9), and SU increments by 77.42% absolute, from 0 (Table 5) to 77.42% (Table 9). In other words, the technique allows the system to understand correctly approximately 8 out of 10 sentences out-of-context analysed.

The price to pay for this clear enhancement in the out-of-context analysis is a little reduction in the scores for the in-context analysis, as can be observed when comparing the in-context results set out in Tables 5 and 9. This comparison shows that when the two-level speech recogniser carries out the in-context analysis using $p=13$, WA decrements by 2.05% absolute, from 94.71 (Table 5) to 92.66% (Table 9), and SU decrements by 3.41% absolute, from 91.03 (Table 5) to 87.62% (Table 9). These results indicate that if the users of the dialogue system would always answer the prompts with the appropriate sentence types, it would be preferable to use the original input interface. However, we cannot assume this user behaviour in real dialogues since inexperienced users may utter sentences not permitted by the prompt-dependent grammars (e.g. when they try to correct system errors), causing the system malfunction. Hence the proposed technique should be used to allow the analysis of these sentences.

The future work includes studying alternative ways to enhance the procedure used to increment the transition probabilities in the GWs. Using word-class pairs eases the procedure but enables that occasionally the probabilities of some transitions get wrongly incremented. In order to avoid this problem, it would be possible to include syntactic and semantic rules to decide whether to increment probabilities. For example, a syntactic rule would suggest not to increment the probability of the transition 'dos' → 'bocadillo' ('two' → 'sandwich') because the number correspondence between both words is not observed, and a semantic rule would indicate not incrementing the probability of the transition 'cerveza' → 'tinto' ('red' → 'beer') because the product 'red beer' does not have meaning in the application domain as it does not exist in the product database of the system. However, semantic rules would be domain-dependent and then should be adapted if the system application is changed to deal with travel information, weather forecasts or other domains with different sentence types.

Acknowledgements

The authors thank the reviewers for their valuable comments to enhance this paper.

References

[1] J. Allen, Natural Language Understanding, The Benjamin/Cummings Publishing Company, Inc., Menlo Park, CA, 1995.
[2] J.A. Baca, F. Zheng, H. Gao, J. Picone, Dialogue systems for automotive environments, Proceedings of Eurospeech, Geneva, Switzerland, 2003, pp. 1929–1932.
[3] R. Beutler, B. Pfister, Integrating statistical and rule-based knowledge for continuous German speech recognition, Proceedings of Eurospeech, Geneva, Switzerland, 2003, pp. 937–940.
[4] N.O. Bernsen, One-line user modelling in a mobile spoken dialogue system, Proceedings of Eurospeech, Geneva, Switzerland, 2003, pp. 737–740.
[5] J. Boye, M. Wirén, Robust parsing of utterances in negotiative dialogue, Proceedings of Eurospeech, Geneva, Switzerland, 2003, pp. 649–652.
[6] C.J.K. Ekanadham, J.M. Huerta, Topic-specific parser design in an air travel natural language understanding application, Proceedings of Eurospeech, Geneva, Switzerland, 2003, pp. 629–632.
[7] F. Ehsani, J. Bernstein, A. Najmi, An interactive dialog system for leaning Japanese, Speech Communication 30 (2000) 167–177.
[8] T. Fegyó, P. Mihajlik, M. Szarvas, P. Tatai, G. Tatai, Voxenter™—intelligent voice enabled call center for Hungarian, Proceedings of Eurospeech, Geneva, Switzerland, 2003, pp. 1905–1908.
[9] H. Hardy, K. Baker, H. Bonneau-Maynard, L. Devillers, S. Rosset, T. Strzalkowski, Semantic and dialogic annotation for automated multilingual customer service, Proceedings of Eurospeech, Geneva, Switzerland, 2003, pp. 201–204.
[10] K. Hirose, J. Tago, N. Minematsu, Speech generation from concept for realizing conversation with an agent in a virtual room, Proceedings of Eurospeech, Geneva, Switzerland, 2003, pp. 1693–1698.
[11] X. Huang, A. Acero, H. Hon, Spoken Language Processing. A Guide to Theory, Algorithm and System Development, Prentice-Hall, Englewood, Cliffs, NJ, 2001.
[12] Q. Huang, S. Cox, Automatic call-routing without transcriptions, Proceedings of Eurospeech, Geneva, Switzerland, 2003, pp. 1909–1912.
[13] N. Kitaoka, N. Kakutani, S. Nakagawa, Detection and recognition of correction utterance in spontaneously spoken dialog, Proceedings of Eurospeech, Geneva, Switzerland, 2003, pp. 625–628.
[14] L. Lamel, S. Rosset, J.L. Gauvain, S. Bennacef, M. Garnier-Rizet, B. Prouts, The LIMSI arise system, Speech Communication 31 (2000) 339–353.
[15] D.J. Litman, S. Pan, Designing and evaluating and adaptive spoken dialogue system, User Modelling and User-Adapted Interaction 12 (2002) 111–137.
[16] R. López-Cózar, A.J. Rubio, J.E. Díaz Verdejo, A. De la Torre, Evaluation of a dialogue system based on a generic model that combines robust speech understanding and mixed-initiative control, Proceedings of Language Resources and Evaluation Conference, Athens, Greece, 2000, pp. 743–748.
[17] R. López-Cózar, D.H. Milone, A new technique based on augmented language models to improve the performance of spoken dialogue systems, Proceedings of Eurospeech, Aalborg, Denmark, 2001, pp. 741–744.
[18] R. López-Cózar, A. De la Torre, J.C. Segura, A.J. Rubio, V. Sánchez, Testing dialogue systems by automatically generating conversations, Interacting with Computers 14 (2002) 521–546.
[19] R. López-Cózar, M. Araki, Spoken, Mulilingual and Multimodal Dialogue Systems: Development and Assessment, Wiley, London, 2005.
[20] N. Nakano, Y. Minami, S. Seneff, T.J. Hazen, D.S. Cyphers, J. Glass, J. Polifroni, V. Zue, Mokusei: a telephone-based Japanese conversational system in the weather domain, Proceedings of Eurospeech, Aalborg, Denmark, 2001, pp. 1331–1334.
[21] Y. Niimi, O. Tomoki, T. Nishimoto, M. Araki, A task-independent dialogue controller based on the extended frame-driven method, Proceedings of International Conference on Speech and Language Processing, China, 2000, pp. 114–117.
[22] T. Paek, Empirical methods for evaluating dialogue systems, Proceedings of Second SIGdial Workshop on discourse and dialogue, Aalborg, Denmark, 2001.
[23] B. Pellom, W. Ward, S. Pradham, The CU communicator: an architecture for dialogue systems, Proceedings of International Conference on Speech and Language Processing, China, 2000, pp. 723–726.
[24] L.R. Rabiner, B.H. Juang, Fundamentals of Speech Recognition, Prentice-Hall, Englewood Cliffs, NJ, 1993.
[25] P. Rutten, J. Fackrell, The application of interactive speech unit selection in TTS systems, Proceedings of Eurospeech, Geneva, Switzerland, 2003, pp. 285–288.
[26] S. Seneff, J. Polifroni, Dialogue management in the mercury flight reservation system, Proceedings of ANLP-NAACL Workshop on Conversational Systems. Seattle, Washington, 2000.
[27] M. Siu, M. Ostendorf, Variable n-grams and extensions for conversational speech language modeling, IEEE Transactions on Speech and Audio Processing 8 (2000) 63–75.
[28] M. Takeuchi, N. Kitaoka, S. Nakagawa, Generation of natural response timing using decision tree based on prosodic and linguistic information, Proceedings of Eurospeech, Geneva, Switzerland, 2003, pp. 609–612.
[29] M. Tang, S. Seneff, V.W. Zue, Modeling linguistic features in speech recognition, Proceedings of Eurospeech, Geneva, Switzerland, 2003, pp. 2585–2588.
[30] K. Visweswariah, H. Prints, Language models conditioned on dialog state, Proceedings of Eurospeech '01, Aalborg, Denmark, 2001, pp. 251–254.
[31] C. Wang, S. Cypherws, X. Mou, J. Polifroni, S. Seneff, J. Yi, V. Zue, MuXing: a telephone-access Mandarin conversational system in the weather domain, Proceedings of International Conference on Speech and Language Processing, China, 2000, pp. 715–718.
[32] H.-M. Wang, Y.-C. Lin, Sentence verification in spoken dialogue systems, Proceedings of Eurospeech, Geneva, Switzerland, 2003, pp. 621–624.
[33] S. Young, D. Kershaw, J. Odell, Ollason Dave, V. Valtchev, P. Woodland, The HTK book (for HTK Version 3.0), Microsoft Corporation, 2000.
[34] V. Zue, S. Seneff, J. Glass, J. Polifroni, C. Pao, T. Hazen, L. Hetherington, Jupiter: a telephone-based conversational interface for weather information, IEEE Transactions on Speech and Audio Processing 8 (1) (2000) 85–96.