SPOKEN DIALOG SYSTEMS: FROM THEORY TO TECHNOLOGY

Giuseppe Riccardi[*]*, Paolo Baggia' and Pierluigi Roberti**
*DIT Department, University of Trento, 'Loquendo

ABSTRACT

Interacting with machines that listen, understand and react to human stimuli has been for many years the holy grail of scientists across disciplines. In the last three decades scientists have made great contributions to the training, design and testing of conversational systems. In this paper we present the fundamentals of Spoken Dialog Systems (SDS) from Automatic Speech Recognition, to Spoken Language Understanding and to Text-to-Speech Synthesis. We report on the spoken dialog system architecture and experiments within a university help-desk application.

Index Terms— Automatic Speech Recognition, Spoken Language Understanding, Spoken Dialog Systems.

1. INTRODUCTION

Interacting with machines that listen, understand and react to human stimuli has been for many years the holy grail of scientists across disciplines. First attempts to build such machines dates back to Kratzenstein's and Von Kempelen's (18^{th} century) mechanical speech synthesizer. However, it was not until first programmable computers were invented that computer scientists experimented with models of language understanding and human-interaction.

One of the debate in the scientific community has been primarily about (not) making anthropomorphic machines. Should we program the peripheral auditory processing of humans into computers? Should we parse language according to linguistic theories? Should computers be emotional? The answer to these questions has been different at times depending on factors such as the knowledge about human speech/language processing as well as the availability of large databases and fast machines. Last but not least. in the last decade, some of the scientific contributions have contributed to generate technology used by millions of users in commercial applications.

In this paper we review the fundamental components of spoken dialog machines and their theoretical motivation. We decompose the "*speech and language chain*" into its building blocks. The Automatic Speech Recognizer (ASR) decodes the speech signal into word string hypotheses. The Spoken Language Understanding (SLU) maps word hypotheses into semantic interpretations. The Dialog Manager (DM) takes as input the semantic interpretations and instantiates a machine action. The Natural Language Generator (NLG) instantiates a linguistic realization from the machine action. In the last step of the speech chain, the Text-to-Speech (TTS) synthesizer verbalizes the linguistic realization provided by the NLG.

In the next sections we review each component and the modeling framework. We review the international standards for spoken dialog systems. In the last two sections of the paper we present an application based on state-of-the-art spoken dialog technology.

2. SPOKEN DIALOG SYSTEMS

2.1 Automatic Speech Recognition

In ASR the goal is to decode the word string that is *encoded* in the spoken utterance. The modern approach to ASR is to frame this problem in the noisy channel paradigm. The spoken utterance can be viewed as the noisy version of the word string and thus the decoding problem can be posed as:

$$\hat{W} = \arg\max_{W} P(W \mid A) = \arg\max_{W} \frac{P(A \mid W)P(W)}{P(A)} \quad (1)$$

where $\hat{W}$ is the decoded word sequence, A is the speech signal, W is the generic word sequence drawn from a vocabulary V The term *P(A|W)* is computed by estimating the *acoustic* model, while P(W) is the stochastic language model (SLM). The denominator *P(A)* is not relevant for the optimization problem in Eq. 1. In modern ASR the speech signal A is compressed into acoustic features that are required to be minimal (wrt number), robust to noise and discriminative (e.g. wrt to speech sounds). An effective approach to feature extraction is computed through Mel Frequency Cepstrum Coefficient (MFCC) analysis. The core of this procedure is the Mel filter bank which is motivated by human speech perception experiments [1]. Alternative approaches to feature extraction for ASR both in the frequency and time domain can be found in [2]. The probability *P(A|W)* is estimated by the acoustic model which needs to account for how acoustic features are generated to produce sequence of speech units (e.g. vowels) and ultimately words. The most successful mathematical model to represent the generation of speech units within a stochastic framework is the Hidden Markov Model (HMM). HMMs define a double stochastic process with a 1) state transition probability matrix $\boldsymbol{A}=\{a_{ij}\}=\{P(s_j \mid s_i)\}$ over state pair space and an 2) output symbol probability distribution $\boldsymbol{B}=\{b_k\}=\{P(z|s_k)\}$ for each state s_k and symbol z in the dictionary of speech units (e.g. phoneme) [3]. HMMs allow us to define the search space

[*]G. Riccardi and P. Roberti are partially funded by Marie Curie Actions MEXT-CT-2005-022593 and MIRG-CT-2006036550

of symbol sequences by designing the state topology. A review of HMMs and alternative models can be found in [4].
In Eq. 1 the term *P(W)* is computed by estimating the probability of a generic word sequence $W=w_1, w_2, ..w_M$ where $w_i \in V$ and *M* is the number of words in the spoken utterance. In general, we do not know the *grammar* used by the human interacting with computers. There have been many attempts at formalizing grammars describing the space of allowed word sequences but with limited success in ASR, mostly due to coverage and brittleness issues. The most successful approach to modelling word sequences is purely statistical. The probability P(W) is decomposed into *n*-gram probabilities:

$$P(W) = \prod_i P(w_i \mid w_1 \cdots w_{i-1}) \approx \prod_i P(w_i \mid h(w_1 \cdots w_{i-1})) \quad (2)$$

where the function $h(w_1 \cdots w_{i-1})$ maps the substring $w_1, w_2, ..w_{i-1}$ into an equivalent class. The most used equivalence class function is $h(w_1 \cdots w_{i-1}) \rightarrow (w_{i-n+1} \cdots w_{i-1})$ and $P(W) = \prod_i P(w_i \mid w_{i-n+1} \cdots w_{i-1})$. The *n*-gram probability $P(w_i \mid w_{i-n+1} \cdots w_{i-1})$ requires counts of *n*-tuples $C(w_{i-n+1} \cdots w_{i-1} w_i)$ estimated from large corpus of transcription of spoken utterances. Since most of *n*-tuples counts is zero smoothing techniques are used to estimates a non-zero probability for all word *n*-tuples[5].

2.2 Spoken Language Understanding

ASR engine must have access to prior knowledge about the language to be recognized, in order to have good performance. If a dictation application or advanced SDS heavily relies on probabilistic knowledge to constrain the ASR decoding, such as the *n*-gram probabilities, in SDS it is very effective to pre-define all the possible formulations that the user might say as well as define a mapping between words and their meaning.

```
<grammar version="1.0" xml:lang="en-US"
  xmlns="http://www.w3.org/2001/06/grammar"
  tag-format="semantics/1.0" root="fromto">

  <rule id="fromto" scope="public">
    from <ruleref uri="#city"/>
         <tag>out.fromcity=rules.latest();</tag>
    to <ruleref uri="#city"/>
       <tag>out.tocity= rules.latest();</tag>
  </rule>
  <rule id="city">
    <one-of>
      <item>Brussels<tag>out="BRU"</tag></item>
      <item>Paris<tag>out="CDG"</tag></item>
      <item>Rome<tag>out="FCO"</tag></item>
    </one-of>
  </rule>
</grammar>
```

Fig. 1. Simple SRGS grammar with SISR script

The most common kind of speech grammars are Context Free one (type 2 of the Chomsky hierarchy [6]) which can be integrated in the speech decoding process in a tractable, efficient and well performing algorithm. Relevant work has been done by World Wide Web Consortium (W3C) in the definition of a standard syntax for grammars: SRGS -"Speech Recognition Grammar Specification Version 1.0" [7], which is largely adopted by the speech engine products.
A second improvement from W3C has been the definition of a standard for *semantic interpretation*, which is a part of a speech grammar devoted to the generation of a semantic result. In many architectures the ASR result is a string of recognized words or multiple results (e.g. N-bests/lattices). SISR, which stands for "Semantic Interpretation for Speech Recognition Specification Version 1.0" [8], offers two alternative ways to encode semantic interpretation, the first is called *literal semantics*, which allows to transliterate a recognized word for instance to return a single value for alternative pronunciation ("coke" for "coca cola").
In alternative SISR 1.0 allows *script semantics*, which is based on ECMAScript/JavaScript (ECMA-327, the computationally constraint version of ECMA-262). With script semantics the ASR engine can return complex semantic results, apply validations to increase or reduce the reliability of the recognized result, or encode the result to be suitable for the speech application. The grammar in Figure 1 includes SISR script semantics inside `<tag>` element, so that for "from Rome to Paris" returns the following ECMAScript object `{fromcity: "FCO",tocity: "CDG"}`. While CFG formalism is powerful in SLU we aim at understanding conversational speech, where we need to be apply robust parsing techniques for mapping words to concepts [9].

2.3 Dialog Manager (DM)

The dialog manager implements the control algorithm that takes the semantic interpretation hypotheses from SLU and computes which action the machine should take. For example if the SLU has hypothesized a user request for *flight status*, the DM might generate a machine request for *which flight*. The DM controls the type of actions (linguistic vs non-linguistic), timing of actions (when to perform them) as well as their sequences (strategies). In most DMs the underlying assumptions, is that the machine is expected to help the user perform a pre-defined task (e.g. travel information and planning). Task complexity has a direct impact on the complexity of the DM control algorithm. The simplest approach to DM is to encode the control via a set of rules or finite state automata [10]. While the latter approach is simple as well as limited, more recently there have been promising approaches to stochastic modelling of DM control [11].

2.4 Natural Language Generation (NLG)

Natural language generation is the process of generating natural language output from DM actions. Other source of knowledge can affect such process (e.g. flight databases). NLG is the inverse of SLU with the most notable difference that the input is not-linguistic. The NLG algorithm is expected to select what-to-say and how-to-say. Since there are many linguistic choices for a given input, NLG algorithm decides what is the best amongst many alternatives. NLG in Spoken Dialog Systems is currently limited to selection from predefined textual realization or template filling at best. However in text processing there have been early attempts to design complete NLG systems [10].

2.5 Speech Synthesis

The main technique in modern TTS is the *Unit Selection* concatenation technique [12]. The basic idea is to extend the concatenation units from diphones to wider units from a large speech database. The result is highly intelligibile and natural compared with early attempts in the '70s. Such advances were possible also due to increased availability of CPU power and memory.

The TTS architecture [13] is composed of two main modules: the *Text Analyzer*, whose goal is to convert the input text into a phonetic and prosodic representation; and the *Speech Synthesizer*, whose goal is to find the optimal sequence of units, then to concatenate them to obtain the synthesizer output. Text analysis relies on language knowledge, lexicons and rules; Speech Synthesis relies on speech processing techniques to find the optimal sequence of segments and smoothly concatenate them. Acoustic databases are designed to provide high phonetic/prosodic coverage of the intended language (or application domain).

One of the research topic is to generate output with emotional control [14-15]. Loquendo TTS voices are produced with a list of "expressive cues", which enable TTS users to enliven their voice prompts. This is a concrete attempt in the direction of expressive synthetic speech. Another important area is the multilinguality, where the TTS should be able to read multiple languages, even mixed in the same sentence. A common approach is to record voices from bilingual or polyglot speakers and allow the TTS engine to select the proper language. A more general approach is based on Phoneme Mapping [16], where each word/sentence is transcribed according to its language, and then the phonemes are mapped into similar ones according to articulatory features in the target language. Phonemes that do not belong to the native phonological system of the voice must be replaced by the most similar sounds available. Moreover there is increased interest in exploting advanced techniques developed for ASR into TTS, see [17].

3. SPEECH STANDARDS

Today commercial SDS are based on VoiceXML 2.0/2.1 [18-19], released by the W3C Voice Browser (VBWG) in 2004. VoiceXML markup describes in a declarative way the dialog interaction, where a VoiceXML platform downloads the documents by http from an application server. This allows to decouple the dialog logic from the platform and to re-use the web architecture to produce and deploy speech applications. For these reasons the VoiceXML is today pervasive into commercial speech applications.

A second area of interest for the application of speech technologies is the multimodal environment, which is pushed by the increasing presence of mobile devices, kiosks, and laptop. In a multimodal applications the DM must be able to integrate and coordinate different input/output modalities, such as speech, gesture, audio, video, etc. To promote standards for MMI W3C launched in 2002 the Multimodal Interaction working group, in [20] the proposals of a Multimodal architecture and other related standards (EMMA for rich annotation of input and InkML). The first generation of standards (VoiceXML 2.0/2.1) is now a pervasive reality, but a new generation is under development, such as SCXML [21] which allows a DM to control different modalities and is going to become the build block of future speech and multimodal applications.

4. SPOKEN DIALOG SYSTEM

In this section we describe the architecture of an SDS based on speech grammars and VXML standard platform used in our lab. In Figure 2 the architecture of the Spoken Dialogue System is supported by the VXML platform. The DM engages the user in a conversation to drive the user through the completion of its request. The DM processes the semantic representation of the utterance returned by the VXML platform and query the following resources to generate a machine action. **Dialog History:** the database where the information about past spoken dialog events (e.g. users' transcriptions, semantic interpretations, machine actions etc.) is stored. The **Domain Concept Ontology (DCO):** A hierarchical description of the set of domain concepts. The **Action Ontology (AO):** A set of predicates that operate on one or more concepts. The **Action Constraints (AC) :** a set of pre-conditions that are necessary to activate any of the predicate in the Action Ontology. **Strategy Planner (SP):** this modules generates a sequence of actions to be taken based on the input from AO, AC and Dialog History. The DM matches the current semantic interpretation against the DCO and retrieves the possible machine actions from the AO. This set is validated by matching the action constraints present a given dialog turn. The set of available machine actions are then used to build a strategy plan via the SP module. The key resource of the DM control is the dialog history which is instrumental to take decisions in the context of current or past dialog interactions. A more detailed discussion of our data-centric approach to Dialog Management can be found in [22]. When the DM has selected the *best* machine action, the NLG module is invoked and then its output verbalized by the platform TTS. In the current implementation the NLG is driven by templates or text tables indexed by machine actions. The DM iterates the previously described steps until the call reaches a final state (e.g., in the case of a routing application, the call is transferred to a live agent, an IVR or the caller hangs up or the caller reaches the goal).

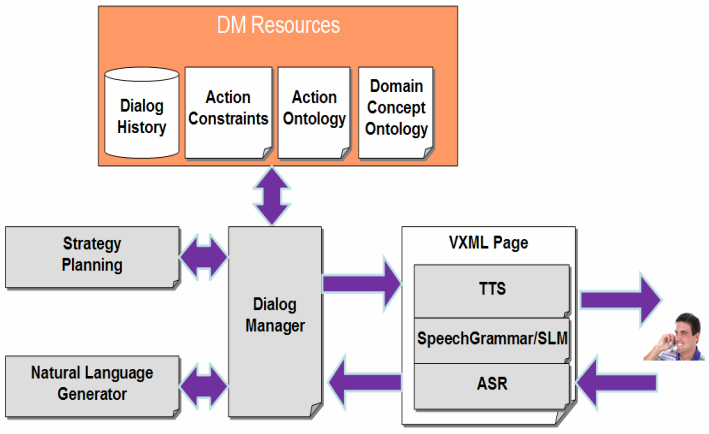

Fig. 2. Spoken Dialog System Architecture

In our architecture we have isolated the data (e.g. Dialog History, DCO, etc..) from the control (DM) so that it can support multiple DM models (e.g. rules, finite state automata, stochastic models) and adapt to new in-domain contexts or multiple domains (e.g. different DCO).

5. EXPERIMENTS

In this section we describe the experiments we performed within an application domain. The selected domain is university help-desk (The UNITN Help-Desk). Students call in

to ask general information (e.g. date of exams) or request services (e.g. sign-up for exams). This is an on-going research project at University of Trento and in the next two sections we describe the current functionalities of our SDS system as well the spoken corpora we have collected. Currently our SDS supports the following machine sub-tasks:

- **Subscribe**. the caller selects one of classes from the database and sign up for classes or mid-terms.
- **Cancel**. the caller cancels previous subscriptions.
- **Information.** The caller asks information about classes (date, time, room, type of exam, ...)
- **Operator.** The caller asks for a human operator.

During the call, the caller can perform more than one subtask. For instance he can first ask information about a class and then sign-up for it.

5.1.Help-Desk Dialog Corpus and Task Accuracy

We have collected 200 spoken dialogs in a Wizard-Of-Oz modality. The user calls in the telephony platform and the SDS described in the previous section handles the call until a human operator (Wizard) takes control of the conversation to complete, if necessary, the call. This protocol provides us a spoken dialog corpora with a selection of automated interactions as well as human-human *recovery* dialogs. The ultimate goal is to adapt the SDS and learn from the human-human dialogs.

The ASR and TTS engine and VoxNauta platform were provided by Loquendo. We wrote the speech grammars based on the specification described in paragraph 2.2. They are based on the phrase-filler structure commonly used for slot-filling tasks. We have designed the ontologies for the domain concept and actions (DCO and AO) as well as the strategy planner and DM controller. Current DM controller is rule based. One of the research challenges in DM control is the evaluation phase. The evaluation is in general complex and should take into account several aspect of the interaction. Moreover, dialog evaluation might require supervision of human annotators to determine "what the user actually said or meant" [23]. This is time-consuming and expensive. In our evaluation experiments we examined automatically (no supervision) the concept and task accuracy by collecting dialog history statistics. We queried the dialog history database and retrieved those concepts or actions that were successfully completed. Success was detected when the user gave positive feedback (e.g. he/she gave a positive answer to a confirmation request.). The estimated task success rate on the entire set (200 dialogs) is 59% which is the percentage of times the task was successfully completed. The task required to collect information (e.g. concepts in the DCO) and perform and action (e.g. sign-up for a mid-term). In the unsuccessful dialogs (task not completed) some of the concepts had been correctly understood. In these cases, 29.5% of the goal has been correctly understood (e.g. sign-up for mid-term *vs* operator request) while 11.5% of unsuccessful dialogs have correctly identified the goal and the target concept (e.g. the name of the class).

6. CONCLUSION

We have described the fundamental information processes of spoken dialog systems. Current state-of-the-art mathematical models rely on statistical modeling of data from automatic speech recognition to text-to-speech synthesis. Formal model of concept and domain knowledge is required to complete the design of spoken dialog systems.

7. ACKNOWLEDGEMENTS

We would like to thank LOQUENDO for granting us their speech technologies and VoxNauta multilingual platform.

8. REFERENCES

[1] S.S. Stevens, J. Volkman, and E.B. Newman, "A scale for the measurement of the psychological magnitude of pitch", *Journal of the Acoustical Society of America*, 8, 185–190, 1937.

[2] X. Huang, A. Acero, H.W. Hon, *Spoken Language Processing*, Prentice Hall, 2001.

[3] L.R. Rabiner, "Tutorial on Hidden Markov models and selected applications in speech recognition", *Proc. IEEE*, vol. 77(2), pp. 257-286, 1989.

[4] J. A. Bilmes, "What HMMs can do", *IEICE Trans. Inf and Syst.*, vol. E89-D(3), 2006.

[5] J. Goodman, "A Bit of Progress in Language Modeling", *Computer Speech and Language*, vol. 15, n. 4, pp. 403-434, 2001.

[6] N. Chomsky, "On certain formal properties of grammars", *Information and Control*, vol. 2, pp. 137-169, 1959.

[7] A. Hunt, and S. McGlashan, "Speech Recognition Grammar Specification Version 1.0", *W3C Recommendation*, Mar. 2004.

[8] L. van Tichelen, and D. Burke, "Semantic Interpretation for Speech Recognition (SISR) Version 1.0", *W3C Recommendation*, Apr. 2007.

[9] A. L. Gorin, G. Riccardi and J. H. Wright, "How May I Help You?", *Speech Communication*, vol. 23, pp. 113-127, 1997,

[10] D. Jurafsky, and J. Martin, *Speech and Language Processing*, Prentice Hall, 2000.

[11] E. Levin, R. Pieraccini and W. Eckert, "A stochastic model of human-machine interaction for learning dialog strategies", *IEEE Trans. SAP*, vol. 8(1), 2000.

[12] M. Balestri et Al., "Choose the best to modify the least: a new generation concatenative synthesis system", *Proc. of EUROSPEECH-99*, Vol. 5, pp. 2291-2294, 1999.

[13] R. Sproat (ed.), *Multilingual Text-to-Speech Synthesis: The Bell Labs Approach*, Kluwer Academic Publishers, 1997.

[14] E. Eide, et al., "Towards Synthesizing Expressive Speech", *Text to Speech Synthesis: New Paradigms and Advances*, pp. 219-248, Prentice Hall, 2004.

[15] E. Zovato et al., "Towards emotional speech synthesis: a rule based approach", *Proc. 5th ISCA Speech Synthesis Work.*, 2004.

[16] L. Badino, C. Barolo and S. Quazza"Language Independent phoneme mapping for foreign TTS", *Proc. ISCA Speech Synthesis Work.*, pp.217-218, 2004.

[17] M. Ostendorf, and I. Bulyko, "The Use of Speech Recognition Technology in Speech Synthesis," *Text-to-Speech Synthesis: New Paradigms and Advances*, pp. 109-133, Prentice Hall, 2004.

[18] S. McGlashan, et al., "Voice Extensible Markup Language (VoiceXML) Version 2.0", *W3C Recommendation*, Mar. 2004

[19] M. Oshry, et al., "Voice Extensible Markup Language (VoiceXML) 2.1", *W3C Recommendation*, Jun. 2007

[20] W3C Multimodal Interaction : http://www.w3.org/2002/mmi

[21] J. Barnett, et al., "State Chart XML (SCXML): State Machine Notation for Control Abstraction", *W3C Working Draft*, Feb. 2007.

[22] S.Varges and G. Riccardi, "A Data-Centric Architecture for Data-Driven Spoken Dialog Systems", Proc. IEEE ASRU Workshop, Kyoto, 2007.

[23] Walker M. et al., "PARADISE: A Framework for Evaluating Spoken Dialogue Agents",Proc. Assoc. Comp. Linguistics Conf. 1997.