TALKING TO MACHINES (STATISTICALLY SPEAKING)

Steve Young

Cambridge University Engineering Department
Trumpington Street, Cambridge, England, CB2 1PZ

ABSTRACT

Statistical methods have long been the dominant approach in speech recognition and probabilistic modelling in ASR is now a mature technology. The use of statistical methods in other areas of spoken dialogue is however more recent and rather less mature. This paper reviews spoken dialogue systems from a statistical modelling perspective. The complete system is first presented as a partially observable Markov decision process. The various sub-components are then exposed by introducing appropriate intermediate variables. Samples of existing work are reviewed within this framework, including dialogue control and optimisation, semantic interpretation, goal detection, natural language generation and synthesis.

1. INTRODUCTION

A statistical model of a spoken dialogue system is shown in Fig. 1. The system operates cyclically. It begins with a default, system-initiated, dialogue act A_s which is converted to an acoustic signal $\boldsymbol{X}_o$ inviting the user to speak. Based on the user's current beliefs, B_u, the user generates a signal $\boldsymbol{X}_i$ which is corrupted by noise before being input to a speech understanding component as the acoustic signal $\boldsymbol{Y}_i$. The noisy speech signal $\boldsymbol{Y}_i$ is decoded to give a set of dialogue acts A_u. These dialogue acts are interpreted by the domain model and used to transform the system state.

The precise definition of a *dialogue act* is not critical. Here it suffices to define it as a frame of information representing an *intention* with an internal structure consisting of slot/value pairs. The latter will be referred to here as *semantic concepts*. As an example, an automated pizza ordering system might say: "Pizza World - how may I help you?" and the user might respond: "I would like two pepperoni pizzas, please". In this case, A_u would be the single dialogue act

Purchase_Request{ qty = 2; topping = pepperoni; }

and it would result in the system state S being updated to record the request for two pepperoni pizzas. Since errors can occur, the components of A_u may have various confidence levels associated with them and these will be encoded within the system state S.

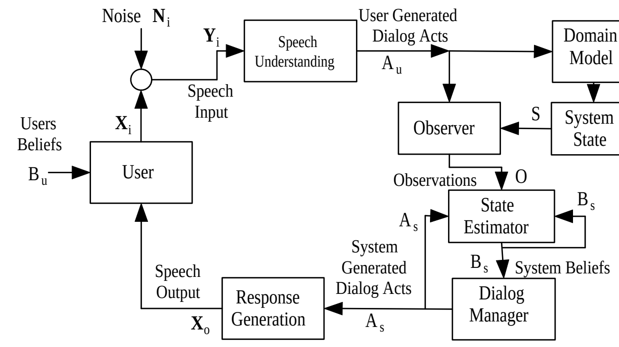

Fig. 1. Structure of a Spoken Dialogue System

The decoded A_u and the updated S are combined to form an observation O from which the system's beliefs B_s are updated. Based on these beliefs, the dialogue manager generates a set of system generated dialogue acts, encoding perhaps, a request to confirm the topping and give the size, e.g. "What size pepperoni pizzas would you like?". The whole cycle then repeats.

This statistical model of an SDS is characterised by a minimal dependence on explicit rules, and a heavy dependence on learning from data. The overall dialogue can be represented by a Partially Observable Markov Decision Process (POMDP) for which the optimal dialogue management policy can be learned from data. The speech understanding component can be designed using the traditional pattern-matching paradigm of estimating the posterior distribution $P(A_u|\boldsymbol{Y})$ and then decoding using a *maximum a posteriori* (MAP) rule. The response generation can be implemented by modelling $P(\boldsymbol{X}_o|A_s)$ and then sampling the distribution.

Although simplistic in certain aspects, the above framework is capable of modelling most of the hand-crafted limited domain applications being deployed today in areas such as information inquiry and e-commerce. The potential advantages of the statistical approach are greatly reduced deployment costs and more robust operation. The drawback, and a major inhibitor to progress, is the difficulty in obtaining suitable training data. This latter issue is dependent on the level of annotation required for training and one of the key research issues is to find robust ways of learning struc-

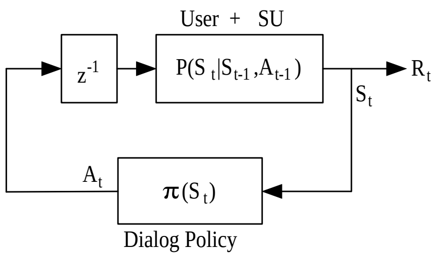

Fig. 2. MDP Representation of a Dialogue System

ture from unannotated and partially annotated data. The following sections explore these issues in more detail.

2. DIALOGUE MANAGEMENT AND CONTROL

As indicated in the preceding introduction, a spoken dialogue system can be regarded as a Markov Decision Process and reinforcement learning used to find optimal dialogue management policies. Early work in this area was pioneered by Pieraccini and Levin [1, 2]. They modelled an SDS as a fully observable MDP in which the user and the speech understanding component were combined to form a single system represented by a transition function $P(S_t|S_{t-1}, A_{t-1})$ where S_t is the system state and A_t is the action applied to the system at time t (see Fig. 2). In an SDS, A_t is a set of system dialogue acts A_s determined by a policy function $\pi(S_t)$. The combination of action and state at time t determines the expected *immediate reward* $r_{t+1} = r(S_t, A_t)$ and the goal is to find the policy which maximises the *total reward* R_0 where $R_t = \sum_{\tau=t}^{T-1} r_{\tau+1}$. The immediate rewards can be selected to meet any dialogue design criteria but commonly a small negative reward is given at each dialogue turn. A larger positive reward is then given on completing successfully and a corresponding negative reward is given for completing unsuccessfully. Thus, the system is motivated to succeed and complete the dialogue as quickly as possible.

This model fits the classic framework of reinforcement learning (see [3] for a good introduction). All solution methods depend on finding the value function

$$V^\pi(S) = E_\pi\{R_t|S_t = S\} \tag{1}$$

For any state S, this function gives the expected value of reward if the dialogue proceeds to the terminal state using policy π. Even more useful is the closely related function

$$Q^\pi(S,A) = E_\pi\{R_t|S_t = S, A_t = A)\} \tag{2}$$

which gives the expected value of reward if action A is taken from state S. If this Q function is known, then the policy π' determined by

$$\pi'(S) = \operatorname*{argmax}_A Q^\pi(S,A) \tag{3}$$

is guaranteed to be better or equal to π. This provides a basis for policy optimisation. If the system transition function is known, then Q can be found by dynamic programming using an update rule of the form

$$Q(S,A) \leftarrow r(S,A) + \sum_{S'} P(S'|S,A) \max_{A'} Q(S',A') \tag{4}$$

If the transition function is not known, then methods based on sampling actual dialogues can be used. In particular, *temporal difference learning* is a technique in which the Q function is updated after every dialogue turn $(S,A) \rightarrow (S',A')$ by comparing the actual one-step reward with the reward predicted by the Q function

$$Q(S,A) \leftarrow Q(S,A) + \alpha[r(S,A) + Q(S,A) - Q(S',A')] \tag{5}$$

where α determines the learning rate. In order to ensure that Q is considered over all reasonable combinations of (S,A), it is necessary to allow the dialogue to deviate from the optimal policy occasionally. This is commonly done by adopting a stochastic ϵ-soft policy. For each pair (S,A), let $A^* = \text{argmax}_A\{Q(S,A)\}$ then

$$\pi(S,A) = \begin{cases} 1-\epsilon+\epsilon/|\mathcal{A}(S)| & \text{if } A = A^* \\ \epsilon/|\mathcal{A}(S)| & \text{otherwise} \end{cases} \tag{6}$$

where $\mathcal{A}(S)$ is the set of all actions possible from state S. This policy is designed to mostly follow the locally optimal policy but to occasionally explore with probability $\epsilon/|\mathcal{A}(S)|$ a non-optimal action. As $\epsilon \rightarrow 0$, then the soft policy hardens to the optimal deterministic policy [4].

Comparing Fig. 2 with the dialog system shown in Fig. 1, it can be seen that some extensions are needed in order to handle a realistic dialogue within the MDP framework. In particular, complete knowledge of the system state is unrealistic since firstly, the full system state S will typically be intractably large and must therefore be approximated.[1]

Secondly, the user's beliefs cannot be directly observed and must therefore be inferred. Thus, in practice a dialogue system must base its management strategy on incomplete data. Fig. 1 shows how the framework of a Partially Observable Markov Decision Process (POMDP) allows this to be done. The system's beliefs B_s are represented by a finite (and computationally tractable) set of states which encode the most pertinent information relevant to managing the dialogue. The observation vector O is derived from a combination of the user's most recent dialogue acts A_u and the system state S, and it encapsulates the evidence needed to update and maintain the system beliefs B_s as accurately as possible.

Unfortunately, policy optimisation for a POMDP is much more complex than for an MDP. In a POMDP, a belief state

[1]Note that the system state S must encode all relevant history plus all system variables. Even the most simplistic implementation of the example pizza dialogue would require several thousand states.

is a distribution over the underlying state set and is therefore a continuous variable. Value functions over belief states are linear combinations of value functions over states

$$V^{\pi}(B) = \sum_{S} B(S).V^{\pi}(S) \tag{7}$$

Thus, each belief value function becomes a hyperplane in belief space and choosing the optimal policy involves finding, at each step, the hyperplane with the maximum value at any belief point. Although beliefs are not discrete and can therefore not be enumerated, it turns out that belief states can be partitioned into regions which share the same optimal policy. This leads to a number of algorithms for policy optimisation [5, 6]. Unfortunately, however, all current exact algorithms are effectively computationally intractable for all but very small state sets.

Thus, whilst the MDP framework appears to provide a sound basis for modeling spoken dialogues statistically, in practice, substantial compromises must be made in order to apply it. If the state space is too large, all methods become intractable. If the state space is pruned in the simple MDP case, the system becames non-Markovian and it can take no account of uncertain interpretation of the user's beliefs. If the POMDP framework is used, user uncertainty can be accounted for, but the supportable state space is even smaller. In all cases, the difficulty of collecting sufficient training data is acute, especially since changing the policy or the state/space can require a complete new set of data.

Despite the difficulties, progress is being made. Walker *et al* have modelled several real systems using simple MDPs, made tractable by reducing the state space to focus on characteristics of specific interest such as the prompt type and the confirmation type [7, 8]. They use training data collected from live use of the SDS running a random policy designed to explore the state-action space. They were able to obtain significant improvements in reward measure and they were also able to improve ASR performance by discovering more finely tuned strategies for interpreting confidence measures.

One limitation of the approach taken by Walker *et al* is that they were forced to fix the state-space in advance before collecting data. Scheffler and Young avoided this problem by adopting a two stage approach[9, 10]. Firstly, they trained a user simulation model using data obtained with a prototype dialogue system. Secondly, they performed policy optimisation using synthetic data generated by the user model. Like Walker *et al*, they also used a cut-down state-space, but following [11] they found that the use of elligibility traces can compensate somewhat for the non-Markovian behaviour of the resulting system.

Roy *et al* have studied simple robot dialogues within the POMDP framework. They used the ASR output directly as the observations and avoided the intractability of finding exact solutions by using an approximation known as Augmented MDPs[12]. This approximation essentially ranks belief vectors based on the most likely state and the entropy of the belief state. They found that using a POMDP framework led to better behaviour when recognition errors occurred. Zhang *et al* have also used POMDPs to provide robustness against errors [13]. They studied a system with 40 states, 15 actions and 25 observations and compared performance of a grid-based POMDP policy optimisation compared to various augmented MDPs. They found that the POMDP version consistently yielded better solutions suggesting that finding more efficient POMDP-based frameworks for SDS is a worthwhile way forward.

3. SPEECH UNDERSTANDING

Referring back to Fig. 1, the goal of the speech understanding component is to convert each input speech waveform $\boldsymbol{Y} = \boldsymbol{Y}_i$ into a set of dialogue acts $A_u = \{a_1, a_2, \ldots\}$ given the current beliefs about the dialogue state B_s i.e. we seek

$$\begin{aligned}\hat{A}_u &= \arg\max_{A_u} P(A_u|\boldsymbol{Y}, B_s) \\ &= \arg\max_{A_u} P(\boldsymbol{Y}|A_u, B_s)P(A_u|B_s) \end{aligned} \tag{8}$$

To solve this it is convenient to identify the word sequence W carried by $\boldsymbol{Y}$

$$\begin{aligned}P(\boldsymbol{Y}|A_u, B_s) &= \sum_{W} P(\boldsymbol{Y}, W|A_u, B_s) \\ &\approx \max_{W} P(\boldsymbol{Y}|W, A_u)P(W|A_u, B_s) \end{aligned} \tag{9}$$

Substituting equation 9 into 8 and making reasonable dependence assumptions gives

$$\begin{aligned}\hat{A}_u &= \arg\max_{A_u} \Big\{ \max_{W} \Big\{ P(\boldsymbol{Y}|W) \\ &\qquad P(W|B_s)P(A_u|W, B_s) \Big\} \Big\} \end{aligned} \tag{10}$$

A suboptimal sequential solution to equation 10 can be achieved by first solving

$$\hat{W} = \arg\max_{W} \{P(\boldsymbol{Y}|W)P(W|B_s)\} \tag{11}$$

and then solving

$$\hat{A}_u = \arg\max_{A_u} \Big\{ P(A_u|\hat{W}, B_s) \Big\} \tag{12}$$

Equation 10 emphasises the fact that the common practice of factoring the speech understanding problem into a two stage process is suboptimal and the use of a word lattice in place of the single best string $\hat{W}$ should be preferred. In

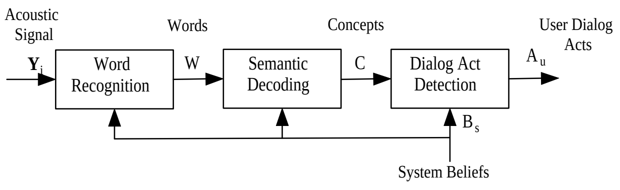

Fig. 3. Speech Understanding

both stages, the dependence on dialogue state B_s is clearly shown. This indicates that there are potential benefits from making the recogniser's language model dependent on the dialogue context[14, 15] and that the dialogue state is crucial for handling ambiguity and identifying underspecified dialogue acts.

In order to evaluate equation 12, it is convenient to introduce another intermediate representation

$$P(A_u|\hat{W}, B_s) \quad = \quad \sum_C P(A_u|C, B_s)P(C|\hat{W}, B_s) \quad (13)$$

where C represents the set of semantic concepts encoded within the word sequence $\hat{W}$. Decoding sequentially (and again suboptimally) gives

$$\hat{C} \quad = \quad \operatorname*{argmax}_C \left\{ P(C|\hat{W}, B_s) \right\} \quad (14)$$

and

$$\hat{A}_u \quad = \quad \arg\max_{A_u} \left\{ P(A_u|\hat{C}, B_s) \right\} \quad (15)$$

Equation 14 represents the process of semantic decoding and equation 15 represents the process of combining the system's beliefs with the decoded semantics to find the most likely dialogue acts. The combination of equations 11, 14 and 15 represent the complete decoding chain for speech understanding and it is illustrated in Fig. 3.

The benefits of using a statistical approach for the word recognition component are well-established and require no further discussion here (see [16, 17] for recent reviews). However, the statistical approach to semantic decoding and dialogue act detection are less well-developed.

4. SEMANTIC DECODING

Semantic decoding in conventional limited domain SDS typically depend on a semantic template grammar and some form of robust parser to extract the required semantic concepts[18, 19, 20]. For example, an utterance such as "I would like two pepperoni pizzas please" might yield the parse shown in Fig. 4 where semantic rules such as "PIZZA $\rightarrow$ [QTY] [TOPPING] pizza[s]" have been used to represent phrasal structure. These rules are domain specific and often require many iterations of user-testing before they achieve adequate coverage.

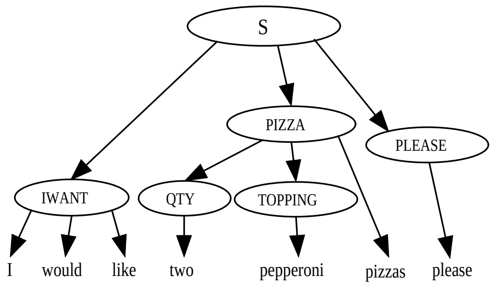

Fig. 4. Example Semantic Parse Tree

The statistical approach to semantic decoding attempts to solve equation 14 directly without any explicit requirement to produce a parse tree. Thus, in the example, extraction of the semantic concepts qty=2; topping=pepperoni is all that is required[2]. This suggests an approach in which each word w in an utterance W is simply tagged with a single discrete concept label c [21, 22]. In the example, "two" would be tagged with QTY and "pepperoni" with TOPPING. Irrelevant words are tagged with a dummy marker and subsequently discarded.

Under this model of semantic decoding, equation 14 can be rewritten as[3]

$$\hat{C} \quad = \quad \operatorname*{argmax}_C \left\{ P(\hat{W}|C)P(C|B_s) \right\} \quad (16)$$

$$\approx \operatorname*{argmax}_C \left\{ \prod_{t=1}^{T} P(w_t|w_{t-1} \ldots w_{t-n+1}, c_t) \right.$$
$$\left. P(c_t|c_{t-1} \ldots c_{t-m+1}, B_s) \right\} \quad (17)$$

In equation 16, $P(C|B_s)$ is often referred to as the *semantic model* and $P(\hat{W}|C)$ is referred to as the *lexical model*. The sequence of concepts $c_1 \ldots c_T$ is modelled as an m-gram conditioned on the current system beliefs and the words are modelled as an n-gram conditioned on the semantic concepts c_t. If $m = 2$ and $n = 1$, the result is a conventional 1st order Markov model with states labelled by semantic concepts and transitions given by the concept bigram probabilities.

As Levin and Pieraccini demonstrated in the 1994 ATIS evaluations [23], this *flat* model of semantic decoding can give surprisingly good results. However, the left-right structure does not allow any hierarchical grouping of the concepts and this weakens the predictive power of the model

[2]Note however that in practice the sequential processing model shown in Fig. 3 may require the set of semantic concepts to be extended beyond simple domain action/entity arguments. For example, IWANT in Fig. 4 might be retained as a semantic concept in order to assist in the subsequent decoding of the relevant dialogue act.

[3]In practice start and end sentence markers are used to enable prediction of the first word and last word of the sentence. This detail is omitted here for clarity.

since adjacent symbols are only weakly coupled. Furthermore, the expressive power is limited by its inability to represent nested structures.

The representational power of the flat concept model can be extended by converting the simple finite state transition network underlying the HMM into a recursive transition network such that any concept-state can itself be a finite state network[24]. This so-called *Hidden Understanding Model (HUM)* extends the class of supported languages from regular to context-free. To apply the HUM, a modified version of the Earley parsing algorithm is used [25] which generates the sequence of concepts $C_1^Q = c_1..c_Q$ or *parse path* corresponding to a depth-first scan of the entire parse tree. In this case, the probabilities of the semantic and lexical models are computed in an efficient interleaved manner as

$$P(C_1^Q)=\prod_{q=1}^{Q}\left\{\begin{array}{ll} P(c_q|c_{q-1},c*) & \text{if } c* \in \text{SemModel} \\ P(w_q|w_{q-1},c*) & \text{if } c* \in \text{LexModel}\end{array}\right. \quad (18)$$

where $c*$ denotes the concept corresponding to the current model along the parse path where both the semantic and lexical models are context-dependent bigrams.

The key problem in building a model for semantic decoding is providing suitable training data since, in general, large quantities of fully annotated tree-bank data will not be available for every (or even most) applications. The ideal semantic model will therefore have the following properties

- powerful enough to capture the necessary semantic structures
- training data easily generated by the dialogue designer
- cost of producing the training data less than the cost of hand-crafting a semantic grammar

In some sense, the two models above represent opposite ends of the spectrum. The flat model of Pieraccini and Levin can be adapted to work with relatively simple training data annotations (e.g. an unaligned list of semantic tags). However, the representational power of the flat model is not generally adequate. On the other hand, the hierarchical HUM of Miller *et al* requires fully annotated tree-bank data. Any attempt to use simpler annotations and let EM discover the hidden structure are very unlikely to work since there are far too many degrees of freedom in an unrestricted context free model [26].

The way forward is therefore likely to be a compromise. It seems reasonable that the training data should be annotated with the corresponding dialogue acts (or equivalent semantic schema) since this type of data is relatively straightforward for the dialogue designer to produce and it can legitimately be regarded as part of the application de-

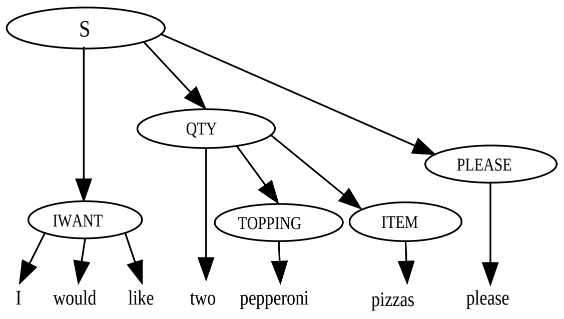

Fig. 5. Right Branching Parse Tree

sign process[4]. The goal of the training algorithm is then to learn to associate words and phrases with particular schema elements.

Wang and Acero have recently demonstrated the viability of this approach using a combination of grammar templates derived automatically from semantic schema and constraint-based parsing [27, 28]. Their basic idea is to use the schema to generate a set of high-level grammar rules. The key informational items in the schema are augmented with optional pre- and post- modifers. A robust parser then scans the training data and associates phrases with the key informational items, associating words with the modifiers as a side-effect. These modifier-word associations fill out the grammar with the required lower level rules. Semantic decoding can also be viewed as an information extraction task. In [29], Chelba and Mahajan describe how a structured language model [30] intialised on general treebank data can be trained for a specific SDS. Essentially this approach is similar to Wang and Acero in that a two level grammar is learnt. The high level is derived from the semantic schema and then used as constraints to parse the training data and learn the lower slot level grammar rules.

An alternative to using automatically inferred grammar templates to provide the needed constraints, is to restrict the underlying formalism. For example, limiting a stochastic grammar to be strictly right-branching leads to a model for which EM-based parameter estimation is tractable[31]. The parse tree for the example sentence is shown in Fig. 5. Note that although the right-branching constraint forces somewhat unnatural dominence relationships, the essential phrase structure can still be maintained. Unlike the general hierarchical model, this probabilistic model is well-suited to left-right decoding. Since each partial path covering $w_1..w_t$ contains exactly the same number of probabilities, paths can be compared directly without normalisation and pruning. Compared to the general SCFG model, the effective state-space and model size is much reduced.

Finally, note that since all of the above methods lead to

[4]Essentially this was the approach taken in ATIS where SQL queries were provided for all training utterances

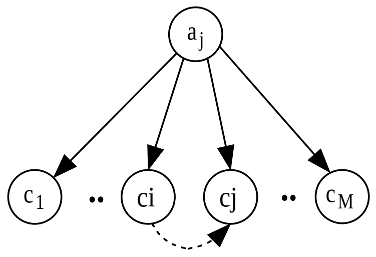

Fig. 6. Dialogue Act Detection using a Bayesian Network

models which can assign overall probabilities to interpretations of a word sequence, their extension to process lattices of alternative word sequences, as suggested in section 3, is relatively straightforward.

5. DIALOGUE ACT DETECTION

The methods described in the previous section were focussed on solving equation 14, this section briefly discusses the solution of equation 15.

Assuming that there is a finite set $\mathcal{A} = \{a_1, .., a_N\}$ of allowed dialogue acts and a finite set $\mathcal{C} = \{c_1, .., c_M\}$ of semantic concepts, the problem is essentially that of determining a mapping from a subset $C \subset \mathcal{C}$ to a subset $A \subset \mathcal{A}$ As such, a variety of solutions are possible. If A is limited to being a single dialogue act, then binary decision trees can be used in which the leaf nodes correspond to dialogue acts and each tree node q is labelled with a subset $C_q \subset \mathcal{C}$ implying the question "Is any input concept $c \in C_q$". Such a tree can easily be trained automatically from examples of actually occurring concept sets and the corresponding dialogue acts.

An alternative which provides a soft-classification and which allows detection of multiple dialogue acts uses Bayesian Networks (BNs) [32, 33]. In the approach developed by Meng and colleagues, one BN is defined for each possible dialogue act as shown in Fig.6. The conditional probabilities $P(c_k|a_j)$ and the priors $P(a_j)$ are learnt from training data. Given a set of input concepts (the evidence), then $P(a_j|c_1, .., c_M)$ is computed for each act a_j, and the most likely are selected as the decoded output. As shown by the dotted line in Fig.6, concepts are not all conditionally independent and performance can be improved by including between concept dependencies. In [34], an automatic method is proposed for learning the most significant dependencies based on minimum description length. Once the evidence has been entered into the network, the posterior concept probabilities can be computed. A concept which has a high posterior but which was not supplied in the evidence is probably needed information, and should be asked for. Similarly, a concept with low probability but which was in the evidence might be spurious, and it should be confirmed with the user. Thus, BNs used in this way can assist with mixed initiative dialogue management. [35, 36].

Finally, in this context the work of Bellegarda and Silverman should be mentioned[37]. They construct a matrix $W = \{w_{jk}\}$ which records the number of times that concept c_k occurred in an utterance conveying dialogue act a_j. They compute the SVD $W = USV^T$ and then at run-time they use U^T to map input concept vectors into a much reduced subspace where a simple distance metric can be used to identify likely dialogue acts. In fact they include words rather than concepts in the input vectors and dispense with semantic decoding altogether giving an effective and robust technique for simple domains.

6. SPEECH GENERATION

Generation is essentially the inverse of the understanding process described in section 3. Given the speech acts A_s and the embedded concepts C_s output by the dialogue manager, an acoustic signal $\boldsymbol{X}_o$ is required which conveys the intended meaning to the user in the most natural possible way, i.e. we seek

$$\arg\max_{\boldsymbol{X}_o} P(\boldsymbol{X}_o|A_s, C_s) = \arg\max_{\boldsymbol{X}_o} \left\{ \sum_W P(\boldsymbol{X}_o|W) P(W|A_s, C_s) \right\} \quad (19)$$

Again decoding sequentially leads to

$$\hat{W} = \operatorname*{argmax}_W \{P(W|A_s, C_s)\} \quad (20)$$

and

$$\hat{\boldsymbol{X}}_o = \arg\max_{\boldsymbol{X}_o} \left\{ P(\boldsymbol{X}_o|\hat{W}) \right\} \quad (21)$$

Equation 20 suggests that the domain specific models used in understanding (or a similar set trained on sample outputs rather than sample inputs) can be reused in "generation mode" to produce the most likely word sequence. However, the model $P(W|C)$ used for understanding was designed to extract the key content words and ignore syntax. Thus, a generation model based on this would lead to outputs with unacceptable surface structure. The solution to this is to smooth equation 20 with a separate language model trained on a large amount of well-formed data. Since this model is only required to capture surface structure, a simple N-gram will suffice. Equation 20 then becomes

$$\hat{W} = \operatorname*{argmax}_W \left\{ P(W|A_s, C_s) + \gamma \tilde{P}(W) \right\} \quad (22)$$

where γ controls the weight placed on grammatical well-formedness. In practice, the two stage process illustrated in

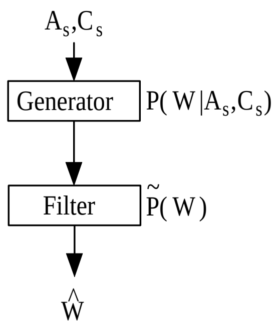

Fig. 7. A Statistical Generation Model

Fig. 7 is used to implement equation 22. First $P(W|A_s, C_s)$ is used to generate a lattice of possible word sequences. $\tilde{P}(W)$ is then used to select the most likely single sequence from the lattice. Language generation based on this *over-generate and filter* paradigm was first proposed by Langekilde and Knight in a system called Nitrogen [38, 39]. They used a hand-crafted semantic template grammar as the generator and a bigram language model for the filter. Atomic concepts in the template grammar were expanded using a lexicon and morphological expansion was performed by rule. However, since the model is tolerant to over-generation, the rules can be kept very simple.

The lack of an explicit tree-based representation of syntax can work well in narrow domains, but wider coverage benefits from more general syntactic knowledge. In [40], a tree-based model is introduced based on the UPenn XTAG grammar. This is claimed to allow richer output forms including the ability to model long-range effects such as the separation of parts of a collocation through embedded adjuncts.

None of the above systems pay much attention to sentence planning. In [41], a sentence planner called SPoT is described which operates on the overgenerate and filter paradigm. In this case the generator takes as input primitive dialogue acts, and stochastically combines these using a set of built-in combination operators such as merge, conjoin, relative-clause, etc. The filter is trained on a corpus of sentences graded by human judges to rank the generator output based on a number of features, the features themselves being determined automatically from the human-ranked data. On evaluation, the plans selected by SPoT are found to be statistically indistinguishable from the best plan selected by human judges.

Turning finally to the generation of the acoustic signal, speech synthesis systems implement equation 21 in essentially the same way as above. They have a large database of phone models and they attempt to find an expansion of $\hat{W}$ which meets various acoustic constraints. Most systems store the waveform segment corresponding to each phone model directly in the database making synthesis a trivial concatenation process [42]. Space precludes a detailed discussion of synthesis methods. However, within the theme of this paper, the work of Tokuda is noteworthy in that he has developed a system which uses HMMs directly as generators thus neatly implementing equation 21 directly [43].

7. CONCLUSIONS

This paper has attempted to provide a general statistical framework for spoken dialogue systems and a variety of on-going work has been described within that framework. There are clearly many hard problems to solve before this general holistic approach is ready for widespread deployment. However, the long-term pay-off is the ability to construct systems which are trainable from data, cheap to build, robust in operation and which can adapt their behaviour on-line.

8. REFERENCES

[1] E Levin and R Pieraccini, "A stochastic model of computer-human interaction for learning dialogue strategies," in *Proc Eurospeech*, Rhodes,Greece, 1997, pp. 1883–1886.

[2] E Levin, R Pieraccini, and W Eckert, "Using markov decision processes for learning dialogue strategies," in *Proc Int Conf Acoustics, Speech and Signal Processing*, Seattle,USA, 1998.

[3] RS Sutton and AG Barto, *Reinforcement Learning: An Introduction*, Adaptive Computation and Machine Learning. MIT Press, Cambridge, Mass, 1998.

[4] SJ Young, "Probabilistic methods in spoken dialogue systems," *Philosophical Transactions of the Royal Society (Series A)*, vol. 358, no. 1769, pp. 1389–1402, 2000.

[5] CC White, "Partially observed markov decision processes: A survey," *Annals of Operations Research*, vol. 32, 1991.

[6] LP Kaelbling, ML Littman, and AR Cassandra, "Planning and acting in partially observable stochastic domains," *Artificial Intelligence*, vol. 101, pp. 99–134, 1998.

[7] MA Walker, JC Fromer, and S Narayanan, "Learning optimal dialogue strategies: A case study of a spoken dialogue agent for email.," in *Proceedings of ACL/COLING 98*, 1998.

[8] S Singh, DJ Litman, M Kearns, and M Walker, "Optimizing dialogue management with reinforcement learning: Experiments with the njfun system," *Journal of Artificial Intelligence Research*, vol. to appear, 2002.

[9] K Scheffler and SJ Young, "Corpus-based dialogue simulation for automatic strategy learning and evaluation," in *Proc NAACL-2001 Workshop on Adaptation in Dialogue Systems*, Pittsburgh, USA, 2001.

[10] K Scheffler and SJ Young, "Automatic learning of dialogue strategy using dialogue simulation and reinforcement learning," in *HLT 2002*, San Diego, USA, 2002.

[11] J Loch and S Singh, "Using eligibiliby traces to find the best memoryless policy in partially observable markov decision processes," in *15th In Conf on Machine Learning (ICML-98)*, 1998.

[12] N Roy, J Pineau, and S Thrun, “Spoken dialogue management using probabilistic reasoning,” in *Proceedings of the ACL 2000*, 2000.

[13] B Zhang, Q Cai, J Mao, and E Chang, “Spoken dialogue management as planning and acting under uncertainty,” in *Eurospeech*, Aalborg, Denmark, 2001.

[14] G Riccardi and AL Gorin, “Stochastic language models for speech recognition and understanding,” in *Int Conf Spoken Language Processing*, Sydney, Australia, 1998.

[15] W Xu and A Rudnicky, “Language modeling for dialog system,” in *Proceedings of ICSLP 2000*, Beijing, China, 2000.

[16] SJ Young, “Statistical modelling in continuous speech recognition,” in *Proc Int Conference on Uncertainty in Artificial Intelligence*, Seattle, 2001.

[17] M Padmanabhan and M Picheny, “Large-vocabulary speech recognition algorithms,” *IEEE Computer*, vol. 35, no. 4, pp. 42–50, 2002.

[18] SJ Young and CE Proctor, “The design and implementation of dialogue control in voice operated database inquiry systems,” *Computer Speech and Language*, vol. 3, no. 4, pp. 329–353, 1989.

[19] W Ward, “Understanding spontaneous speech,” in *Proc Int Conf Acoustics, Speech and Signal Processing*, Toronto, Canada, 1991, pp. 365–368.

[20] V Zue, S Seneff, JR Glass, J Polifroni, C Pao, TJ Hazen, and L Hetherington, “Jupiter: A telephone-based conversational interface for weather information,” *IEEE Trans Acoustics, Speech and Signal Processing*, vol. 8, no. 1, pp. 85–96, 2000.

[21] R Pieraccini, E Levin, and C-H Lee, “Stochastic representation of conceptual structure in the atis task,” in *Proc Speech and Natural Language Workshop*, Pacific Grove, CA, 1991, pp. 121–124, DARPA.

[22] R Pieraccini and E Levin, “Stochastic representation of semantic structure for speech understanding,” *Speech Communication*, vol. 11, no. 2, pp. 283–288, 1992.

[23] E Levin and R Pieraccini, “Chronus, the next generation,” in *Proc Spoken Language Systems Technology Workshop*, Austin, Texas, 1995, pp. 269–271.

[24] S Miller, R Bobrow, R Schwartz, and R Ingria, “Statistical language processing using hidden understanding models,” in *Proc Human Language Technology Workshop*, Plainsboro, NJ, 1994, pp. 278–282, ARPA.

[25] A Stolcke, “An efficient probabilistic context-free parsing algorithm that computes prefix probabilities,” *Computational Linguistics*, vol. 21, no. 2, pp. 165–201, 1995.

[26] K Lari and SJ Young, “The estimation of stochastic context-free grammars using the inside-outside algorithm,” *Computer Speech and Language*, vol. 4, no. 1, pp. 35–56, 1990.

[27] Y Wang and A Acero, “Grammar learning for spoken language understanding,” in *Proceedings of ASRU Workshop*, Madonna di Campiglio, Italy, 2001.

[28] Y Wang and A Acero, “Evaluation of spoken grammar learning in the atis domain,” in *Proc Int Conf on Acoustics, Speech, and Signal Processing*, Orlando, Florida, 2002.

[29] C Chelba and M Mahajan, “Information extraction using the structured language model,” in *EMNLP 01*, 2001.

[30] C Chelba and F Jelinek, “Structured language modeling,” *Computer, Speech and Language*, vol. 14, no. 4, pp. 283–332, 2000.

[31] SJ Young, “The statistical approach to the design of spoken dialogue systems,” Tech. Rep. CUED/F-INFENG/TR.433, Cambridge University Engineering Department, 2002.

[32] D Heckerman and E Horwitz, “Inferring informational goals from free-text queries: a bayesian approach,” in *Proc 14th Conf on Uncertainty in AI*, 1998, pp. 230–238.

[33] HM Meng, W Lam, and C Wai, “To believe is to understand,” in *Eurospeech*, Budapest, Hungary, 1999, pp. 2015–2018.

[34] HM Meng, W Lam, and KF Low, “Learning belief networks for language understanding,” in *IEEE Workshop on Automatic Speech Recognition and Understanding*, Keystone, Colorado, 1999.

[35] S Pulman, “Conversational games, belief revision and bayesian networks,” in *CLIN VII: 7th Computational Linguistics in the Netherlands meeting*, 1996.

[36] HM Meng, C Wai, and R Pieraccini, “The use of belief networks for mixed-initiative dialog modelling,” in *Int Conference on Spoken Language Processing*, Beijing, China, 2000.

[37] J Bellegarda and KEA Silverman, “Toward unconstrained command and control: Data-driven semantic inference,” in *Proc ICSLP*, Beijing, China, 2000, pp. 1258–1261.

[38] I Langkilde and K Knight, “Generation that exploits corpus-based statistical knowledge,” in *Proc !7th Int Conf on Computational Linguistics (COLING-ACL 1998)*, Montreal, Canada, 1998.

[39] I Langkilde and K Knight, “The practical value of n-grams in generation,” in *Proc 9th Int Natural Language Workshop (INLG 98)*, Niagara-on-the-Lake, Canada, 1998.

[40] S Bangalore and OC Rambow, “Exploiting a probabilistic hierarchical model for generation,” in *Proc 18th Int Conf on Computational Linguistics (COLING 2000)*, Saarbrucken, Germany, 2000.

[41] M Walker, OC Rambow, and M Rogati, “A trainable approach to sentence planning for spoken dialogue,” *Computer Speech and Language*, vol. to appear, 2002.

[42] AW Black and P Taylor, “Automatically clustering similar units for unit selection speech synthesis,” in *Proc Eurospeech*, Rhodes,Greece, 1997, pp. 601–604.

[43] Y Tokuda, T Yoshimura, T Masuko, T Kobayashi, and T Kitamura, “Speech parameter generation algorithms for hmm-based speech synthesis,” in *Proc Int Conf Acoustics Speech and Signal Processing*, Istanbul, Turkey, 2000, pp. 1315–1318.