



Have we met? MDP Based Speaker ID for Robot Dialogue

Filip Krsmanovic, Curtis Spencer, Daniel Jurafsky, and Andrew Y. Ng

Department of Computer Science
Stanford University, Stanford, CA, USA

filipk@cs.stanford.edu, curtis@cs.stanford.edu, jurafsky@stanford.edu, and
ang@cs.stanford.edu

Abstract

We present a novel approach to speaker identification in robot dialogue that allows a robot to recognize people during natural conversation and address them by name. Our Stanford AI Robot (STAIR) dialogue system attempts to mirror the human speaker identification process. We model the robot dialogue problem as a Markov Decision Process (MDP) and apply a reinforcement learning algorithm to try to learn the optimal dialogue actions. The MDP model works in conjunction with a traditional statistical cluster based speaker identification subsystem. Our approach also addresses open-set speaker identification, dynamically adding new speaker profiles as well as continuously updating known profiles.

Index Terms: dialogue, MDP, speaker identification, speaker recognition, robot conversation

1. Introduction

“The sweetest sound in any language is the sound of one’s own name.” – Andrew Carnegie

In this paper, we present an MDP based algorithm for speaker ID in a robot dialogue agent. The goal of our Stanford AI Robot (STAIR) project is a robotic assistant, one capable of conversation, for home/office. Such a system must carry out spoken conversational interactions with different humans, address known speakers by name, and recognize new speakers. The robot’s conversational dialogue agent must thus incorporate traditional speaker recognition. Potential applications for speaker ID in robot dialogue include:

- Addressing users by name. Users generally prefer spoken dialogue systems to be personalized [1].
- Authentication, so as not to follow orders of an unauthorized person or a young child.
- Answering questions of the form “Who sent you?”, etc.

Automatic speaker recognition, the task of recognizing a person based on her voice, has traditionally been split into two areas. *Speaker verification*, a binary classification task, deals with confirming whether a person is who she claims to be. Since an imposter may not be known to the system, speaker verification is generally referred to as an *open-set* task. *Speaker identification*, an N-way classification task, deals with finding the most likely match for an input voice sample from a known group of people. Identification usually requires the speakers to be known in advance, making this a *closed-set* problem. Our dialogue-based work lies in *open-set speaker identification*, and combines challenges from both verification and identification. Our system must decide when to identify a currently known speaker, and when to create a new profile for a speaker it has not previously met.

The task of dialogue based open-set speaker ID imposes a number of design constraints. The system must accept text-independent, variable length (often very short) utterances. Users must be identified in a conversationally natural way, much as a human might when they can’t remember someone’s name. Thus the agent must decide whether to make an identification attempt immediately or get more data through additional conversation. The system must be able to deal with unknown speakers, asking their name and adding them to the known set. We also desire incremental learning, continually adapting our model of known speakers in subsequent conversations.



Figure 1: The STAIR robot.

1.1. Proposed Solution

We propose a combination of an MDP/reinforcement learning based dialogue agent (the Dialogue Agent or DA) and a statistical clustering based speaker identification system (the Speaker Identifier or SI).

Following the current state of the art in dialogue agent design [2, 3], we model the dialogue problem as an MDP. In the MDP, the state of the system will capture our degree of certainty about the speaker’s identity (based on the SI’s output). Given the current state, our MDP DA will then choose the next conversation action. Specifically, it will first decide between identifying a speaker on a current dialogue turn or eliciting natural small talk to obtain more data. When identifying a person, the DA will further decide the manner and degree of certainty with which it will do so.

For the SI, we use Vector Quantization (VQ) [4] with MFCC features. VQ is robust and accurate even for small amounts of training data (~5-30 seconds in our domain) [5, 6].



VQ is also fast to train (a necessity since we constantly need to retrain known users), fast to test (hence close to real time identification), and simple. The issues addressed in this paper are also largely orthogonal to the specific SI system used; our MDP DA approach can be straightforwardly incorporated with other more sophisticated SI systems. We also note that most traditional statistical speaker identification systems need to know the possible speakers in advance whereas we are constantly adding new speakers. The DA will handle this limitation in its decision process, described later.

2. Implementation and Methodology

2.1. Speaker Identifier

Whenever a person speaks, our system extracts MFCC feature vectors (using HTK [7]), which are then concatenated and used for training/retraining at the end of every conversation. During training, we employ VQ (using Linde-Buzo-Gray (LBG) algorithm [8]) to find clusters of these training feature vectors, and represent an entire cluster by a single cluster centroid, or codeword. Each speaker is therefore represented by a set of codewords. During testing, we go through each speaker’s codeword set, and for each test vector, we compute its distance to the nearest codeword. The sum of these test vector distances is the total VQ distortion for that speaker, which reflects the likelihood of the test utterance belonging to her. Intuitively, the smaller the distortion, the more likely the utterance was by that speaker. Given an utterance, the SI returns a distortion for each known speaker; these distortions will be used by the DA to define the current MDP state.

2.2. Dialogue Agent

The DA uses an MDP model of the robot dialogue task. A finite-horizon MDP models a sequential decision process, and formally comprises a set of states S , a set of actions A , state transition probabilities $P(s'|s,a)$, a time horizon T , a reward function $R: S \times A \rightarrow \mathfrak{R}$, and a discount factor $0 \leq \gamma < 1$. The reward function R specifies which states are more or less desirable, and the agent’s goal is to choose actions so as to visit states with large positive rewards, and avoid states with negative rewards.

More formally, on each time step t , the agent is in some state $s_t \in S$, and has to choose an action $a_t \in A$. As a result of its choice, it transitions randomly to a new state s_{t+1} according to the transition probabilities $P(s_{t+1}|s_t,a_t)$. After T turns, the agent will have visited some sequence of states s_0, s_1, \dots, s_T . The goal of a reinforcement learning algorithm is to choose actions so as to maximize the expected value of $R(s_0, a_0) + \gamma R(s_1, a_1) + \gamma^2 R(s_2, a_2) + \dots + \gamma^T R(s_T, a_T)$. More precisely, it outputs a policy $\pi: S \mapsto A$ that indicates which action $\pi(s)$ it thinks we should take in each state s .

Given a full MDP model, there are standard algorithms such as value iteration [9] for efficiently finding the optimal policy. In our MDP model, the states, actions, horizon time, rewards, and discount factor are exactly specified, but the state transition probabilities will be estimated from (either simulation or real) data.

2.2.1. States

In our DA, the MDP states capture the current dialogue turn number and also reflect our uncertainty about the speaker ID.

This uncertainty is modeled using the distortions supplied by the SI. More precisely, we map these distortions into a finite number of buckets. Such a mapping is necessary since the dimension of the continuous-valued vector of distortions is unbounded and grows over time (since the set of speakers is constantly growing); but the simplest versions of most MDP algorithms (such as value iteration) require that the set states be finite and fixed in advance.

For this mapping, we begin by extracting the most likely speaker matches by taking the minimum 20% of the distortion values. Doing so provides robustness against outliers with very large distortions. We compute the mean and variance of this 20% subset of the distortion values, and group these values into buckets according to how many standard deviations (σ) they fall below their mean.

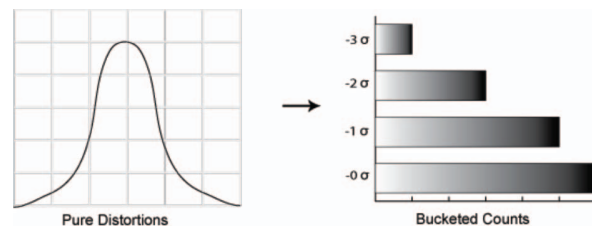


Figure 2: Transformation during the first dialogue turn

To build the MDP state from these buckets, we compute two values: $Count_{min}$ and $Count_{min+1}$ (see Table 1). We set $Count_{min}$ to be the number of distortions falling into the minimum σ bucket that is at least 2σ less than the mean, if such a bucket exists (otherwise, $Count_{min}$ is 0). If the number of distortions in this bucket is > 2 , we set its value to “ n ” because having any number greater than 2 results in fairly similar states where the correct identification is unclear. We similarly set $Count_{min+1}$ to be the number of distortions in the σ bucket that is 1σ closer to the mean and at least 1σ less than the mean (or 0 if no such bucket exists). For example, the bucketed counts in Figure 2 above would result in the state: $\{Turn = 1, Count_{min}=1, Count_{min+1}=n\}$.

Table 1: List of state features and their explanations.

Features	Values	Explanation
Turn(T)	0,1,2,3,4	Dialogue turn number.
$Count_{min}$	0,1,2,n	Number of speaker distortions that fall in the min σ range, (at least 2σ less than the mean). n means count is greater than 2.
$Count_{min+1}$	0,1,2,n	Number of speaker distortions that fall into min+1 σ range, (at least 1σ less than the mean).

The MDP state is then the concatenation of the current conversation turn, the value of $Count_{min}$, and the value of $Count_{min+1}$. This gives a total of 80 possible states. Informally, a state characterized by the lowest distortion value being a lone outlier far below the mean is one where the system would likely identify a known speaker. In contrast, a state that has more speaker distortions closer to the mean, and none much smaller than the mean, is one where we detect a new speaker. In that case, we ask for the speaker’s name, add her to the system and then proceed to learn her set of codewords.



2.2.2. *Actions*

The 5 actions (denoted A_1 - A_5) available to the DA at every state are illustrated in Table 2. They represent the spectrum of uncertainty with which the system can choose to identify a new or known user, naturally and without being rude.

Table 2: Set of actions in the MDP

Action	Response Template
A_1	Hello \$name.
A_2	I think you are \$name.
A_3	I think we have met before.
A_4	What is your name?
A_5	Small talk depending on turn number

The utterance for action A_5 is retrieved from a table containing various small talk responses for a given dialogue turn. With a fully functional Automatic Speech Recognition (ASR) system, we could expand the responses to be more context sensitive by making use of the additional semantic meaning, but in our prototype, simply examining the turn number was sufficient for adequately compelling small talk.

2.2.3. *Finite Horizon*

Our DA uses a finite horizon MDP. We found that speakers can only endure a few dialogue turns of small talk before becoming bored. Thus, after 5 turns, the DA must decide to make one of the possible identifications or ask for a name. The MDP also uses discounting, so that rewards obtained (from a correct identification) later in the conversation are worth less than rewards obtained earlier; this prevents the agent from becoming a chatterbox, encouraging it to make an identification in as few turns as possible.

2.2.4. *Reinforcement Learning*

Once the current state is calculated via the transformation from Section 2.2.1, we find the best action to take in that state by consulting the optimal policy for the MDP. Specifically, the DA chooses actions so as to try to maximize the expected discounted sum of rewards, or utility, obtained. We assign a 0 reward for non-terminal states, and for terminal ones where we attempt identification, we give higher constant reward values to correct and more confident actions (such as “Hello Bobby” rather than “I think you are Bobby”). The system is similarly penalized for incorrect identification, more so for a confident identification that is incorrect, than for a more tentative one. When the system decides to identify a known user, it retrieves the name of the speaker with the minimum distortion and inserts it into the response template. The reward function is dependent on not just the state and the action taken in that state, but also the user confirmation, which decides whether we assign a positive or negative reward. To get speaker confirmation, after the agent makes an identification or requests a new speaker’s name, the system either asks “Did I get your name right?” or “Have we met?” After each conversation, the transition probabilities are updated (see Section 2.2.5 for details), and the new policy is recalculated using one round of value iteration [9]:

$$V_{i+1}(s) := \max_a R(s,a) + \gamma \sum_{s'} P(s'|s,a) V_i(s') \quad (1)$$

where V_i is the value of a state after the i^{th} conversation, R is the reward function, and γ is the discount factor.

As the robot interacts with different users, the transition probabilities are continuously re-estimated, so that the computed policy slowly improves over time.

2.2.5. *MDP Parameter Learning*

As with any machine learning approach, an early hurdle is gathering realistic training data, either real or simulated [10]. In order to train our sample MDP DA for observation, we develop an efficient dialogue simulation engine to run the MDP through many dialogues to initialize its states and transition probabilities. We record 10 speakers, each engaging in free-form conversation for 15 seconds. We then split up the utterances into smaller 3 second clips and use different permutations of clip sequences as sample dialogues to the DA. This allows us to create many sample dialogues out of a small amount of real world data even in the absence of an ASR system, because only the MFCC features, not the content, of the utterance are used for speaker ID. Thus, we build up the MDP with possible states as well as initial transition probabilities using the following formula:

$$P(s'|s,a_i) = \frac{\text{Count}(s \xrightarrow{a} s')}{\text{Count}(s \xrightarrow{a} S_{sa})} \quad (2)$$

where Count is the number of occurrences and S_{sa} is the set of all states that transition from s via action a .

Once the MDP is initialized, we have a system that can then be used in live dialogue to obtain additional data, which allows us to more accurately learn the state transition probabilities that we can then use to re-compute the policy.

3. Results

The ideal evaluation would involve having a complete robotic agent interacting with users, whose reactions we would then compare with and without our system. However, the STAIR platform is still in development, and we must defer such a test to the future. Given that we are primarily interested in incorporating speaker ID into spoken dialogue, we were still able to build a complete system, as described above, and evaluate its interactions with live users.

3.1. Dialogue Agent Results

The following two tables illustrate an example dialogue with the tuned MDP agent using the learned policy, and give the computations that occurred during a part of the conversation.

Table 3: Sample Dialogue with MDP Dialogue Agent

Speaker	Utterance	State	Action
Human	Hello, Robot.		
Robot	Hello, how are you today?		A_5
Human	I am doing well.		
Robot	I am really looking forward to spring break. What about you?		A_5
Human	Definitely. I need a break.		



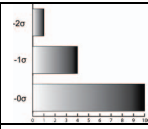
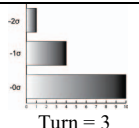
Robot	I think you are Bobby. Did I get your name right?		A_2
Human	Yes		

Table 4: Expected values of state actions and the optimal action chosen in Robot’s 3rd turn

State	Expected Value	Best Action
 Turn = 3 Count _{min} = 1 Count _{min+1} = n	$R(s, A_1) + \sum_s P(s' s, A_1)V(s') = 2.4$	$A_2 = \arg \max_a R(s, a) + \sum_s P(s' s, a)V(s')$
	$R(s, A_2) + \sum_s P(s' s, A_2)V(s') = 3.0$	
	$R(s, A_3) + \sum_s P(s' s, A_3)V(s') = 1.6$	
	$R(s, A_1) + \sum_s P(s' s, A_1)V(s') = 0.6$	
	$R(s, A_2) + \sum_s P(s' s, A_2)V(s') = 0.3$	

3.2 Speaker Identifier

We tested the accuracy and speed of the VQ SI to ensure reasonable performance. We performed a closed-set test on 10 live speakers, training each speaker model on short (10 second) utterances, and testing on both a typical short utterance (1 sec) and a typical long one (5 sec). We also tried training on utterances with both the same words and different words across speakers. We used 128 codewords, a value empirically observed to give the best performance.

Our SI test accuracy was 70% on 1 second utterances and 90% on 5 second utterances. We found no significant differences in SI accuracy between training with same and training with different utterances across users.

Training took 11.46 seconds on average for 10 second utterances. Testing time took between 0.53 seconds (for 1 second utterances) and 1.31 seconds (for 5 second utterances).

4. Discussion and Analysis

Our Dialogue Agent interacted appropriately and adapted based on the user responses. Although a good distortions to states mapping merits additional research, our mapping worked well and correctly handled a growing speaker set, even when the speakers’ distortions differed greatly. A finite horizon MDP also proved helpful, since users were consequently not bored given the 5 turn limit.

In our experiments, the policy started off overly confident, and thus was heavily penalized in the early stages of learning. Over time, the system learned to make identifications that were better calibrated for confidence; for example, making more tentative identifications when uncertain. However, states with significant outliers in the SI distribution resulted in identifications that were correct as well as confident and were therefore awarded positively. Neutral states, as predicted, resulted in STAIR making some new friends.

The Speaker Identifier was found to have adequate accuracy, especially with the longer test utterances. This reinforces our DA’s goal to elicit small talk to gather more data. Nonetheless, the short utterances were often sufficient to make an identification, which is important for the frequent, very short phrases, such as “Hello, Robot.”

Our fast running time (just over real time for training, much faster for test) is critical for performing identification during conversation, and was one of the benefits of using the simple VQ technique for SI.

5. Conclusions

We present a novel approach for embedding speaker identification within a dialogue system for a robotic assistant. The task requires open-set identification, incremental learning and identifying speakers within a conversational context, which we achieve by combining an MDP and reinforcement learning based dialogue agent with a traditional speaker recognition subsystem. The SI subsystem is found to give reasonable performance, and the resulting MDP DA converges to a good policy that allows a robot assistant to naturally identify users, much like humans would.

Future research directions include using more accurate speaker recognition [11,12], using a Partially Observable MDP (POMDP) [13] to model uncertainty in SI output, or implementing a more sophisticated reward function based on the overall quality of the interaction (not just on whether the system properly identified the user). The latter may help correctly balance false negatives (the robot says it has never met the user) and false positives (calling them the wrong name). We hope that our current work and such future work will make natural speaker identification systems a crucial part of any robotic assistant.

6. Acknowledgements

We thank Daniel Chavez, Cheng-Tao Chu, and Jeremy Hoffman for their collaboration and insightful discussions, and the reviewers and Grace Chung for helpful comments.

7. References

- [1] Cole, R. et al., “Perceptive Animated Interfaces: First Steps Toward a New Paradigm for Human Computer Interaction,” *IEEE: Special Issue on Multimodal HCI*, 2003.
- [2] Levin, E., Pieraccini, R. and Eckert, W., “A stochastic model of human-machine interaction for learning dialogue strategies,” *IEEE Trns. on Speech and Audio Processing*, 8:11-23, 2000.
- [3] Singh, S., Litman, D., Kearns, M. and Walker, M., “Optimizing Dialogue Management with Reinforcement Learning: Experiments with the NJFun System”, *Journal of Artificial Intelligence Research*, 16:105-133, 2002.
- [4] Song, F. K., Rosenberg, A. E. and Juang, B. H., “A vector quantization approach to speaker recognition,” *AT&T Technical Journal*, 66-2:14-26, 1987.
- [5] Do, M. N. and Wagner, M., “Speaker recognition with small training requirements using a combination of VQ and DHMM,” *Proc. of Speaker Recognition and Its Commercial and Forensic Applications*, 169-172, Avignon, France, 1998.
- [6] Matsui, T. and Furui, S., “Comparison of text independent speaker recognition methods using VQ-distortion and discrete/continuous HMM’s”, *Proc. ICASSP 1992*, II:157-160.
- [7] Woodland, P. C. et al., “The 1994 HTK Large Vocabulary Speech Recognition System”, *Proc. ICASSP 1995*, Detroit.
- [8] Linde, Y., Buzo, A. and Gray, R., “An algorithm for vector quantizer design,” *IEEE Trans. on Comm.*, 28:84-95, 1980.
- [9] Sutton, R. and Barto, A. G., *Introduction to Reinforcement Learning*, MIT Press, Cambridge, MA, 1998.
- [10] Stuttle, M., Williams, J. and Young, S., “A Framework for Dialogue Systems Data Collection using a Simulated ASR Channel,” *ICSLP 2004*, Jeju, Korea.
- [11] Reynolds, D. A. and Rose, R. C., “Robust text-independent speaker identification using gaussian mixture speaker models,” *IEEE Trans. on Speech and Audio Processing*, 3(1):72-83, 1995.
- [12] Ferrer, L. et al., “The contribution of cepstral and stylistic features to SRI’S 2005 NIST Speaker Recognition evaluation system,” *Proc. ICASSP 2006*.
- [13] Roy, N., Pineau, J., and Thrun, S., “Spoken dialogue management using probabilistic reasoning,” *Proc. ACL-2000*.