CLUSTERING BEHAVIORS OF SPOKEN DIALOGUE SYSTEMS USERS

Senthilkumar Chandramohan [1,3], *Matthieu Geist* [1], *Fabrice Lefèvre* [3], *Olivier Pietquin* [1,2]

[1] SUPELEC Metz Campus, IMS Research Group, France
[2] UMI 2958 (CNRS - GeorgiaTech), France
[3] Université d'Avignon et des Pays de Vaucluse, LIA-CERI, France

ABSTRACT

Spoken Dialogue Systems (SDS) are natural language interfaces for human-computer interaction. User adaptive dialogue management strategies are essential to sustain the naturalness of interaction. In recent years data-driven methods for dialogue optimization have evolved to be a state of art approach. However these methods need vast amounts of corpora for dialogue optimization. In order to cope with the data requirement of these methods, but also to evaluate the dialogue strategies, user simulations are built. Dialogue corpora used to build user simulation are often not annotated in user's perspective and thus can only simulate some generic user behavior, perhaps not representative of any user. This paper aims at clustering dialogue corpora into various groups based on user behaviors observed in the form of full dialogues.

1. INTRODUCTION

Spoken Dialogue Systems (SDS) are complex systems requiring the development of several speech and language processing modules. Especially, the dialogue management (DM) module is in charge of deciding what to do in a given context. Developing such a module has been casted into the Reinforcement Learning (RL) paradigm [1] since the end of the 90's [2, 3]. Dialogue management is indeed a sequential decision problem which can therefore be framed as a Markov Decision Process (MDP) and solved using reinforcement learning. Depending on the considered algorithm, the DM can be trained either directly from the dialogue corpora [4] or more generally using some user simulator [5, 6, 7], itself built using the dialogue corpora. Such user simulators are mandatory for some data expensive RL algorithms. User simulation is also widely used for assessing dialogue systems [8].

Most often, user simulation are statistical machines trained on data of weakly annotated dialogues [9, 10, 11]. A dialogue corpora contains a set of dialogues from different users, which can act very differently. For example, consider a user accustomed to such system (and who somehow shortcut the DM to get quickly the required information) compared to a newcomer who shall wait for machine's instruction before proceeding. In other words, the distribution of users behaviors is intrinsically multimodal. However, dialogue corpora are not annotated to reflect this basic fact and, often, a single user simulator is trained on the global set of data. Therefore, training DM on user simulators (or directly from dialogue corpora) resumes to train against non-representative users.

We claim that training DMs on different types of users instead of on a generic user would lead to more natural and more efficient interactions. However, annotating dialogue corpora to reflect this is cumbersome. Thus, one should cluster automatically users in such corpora. This paper proposes an approach to automatically cluster corpora according to user behavior types without prior model of this behavior. However, clustering user behaviors is far from being straightforward. The main difficulty relies in the fact that dialogues are sequences of interactions resulting in changes in the dialogue context. These sequences can be of different lengths, even if generated by a same user, which makes comparisons required for clustering quite tricky. In [12], the authors are also addressing the clustering problem. Yet, to avoid the problem of the length difference, a specific behavior is analyzed (clarification strategies) and a restricted context of fixed length is considered.

In [13], it is proposed to express user behaviors as a sequential decision making problem and to build user simulators based on inverse reinforcement learning [14]. Taking a similar route we represent user behaviors as a single vector and find a representative space to perform clustering.

2. CLUSTERING BEHAVIORS

In this section, we recall the notion of MDP and explain how a user can be modeled as an MDP, before explaining how this can be used to quantify users behaviors and perform subsequent clustering.

2.1. Markov Decision Process

Markov Decision Processes are a common framework for formalizing sequential decision problems involving an agent interacting with a dynamic system. An MDP is defined as a tuple $\{S, A, P, R, \gamma\}$ with S the state space, A the action space, $P : S \times A \to \mathcal{P}(S)$ as set of Markovian transition probabilities, $R : S \to \mathbb{R}$ the reward function and γ a discount factor weighting long-term rewards. A policy $\pi : S \to \mathcal{P}(A)$ defines how choosing action for each state. At each timestep t, the system is in a state s_t. The agent choose an action a_t according to a policy $\pi(.|s_t)$ and the system steps stochastically to s_{t+1} according to $p(.|s_t, a_t)$. It receives a reward r_t, which is a local hint on the quality of the control. The quality of a policy π is quantified by a so-called value function, defined as the expected cumulative reward starting in a state s and following a policy π :

$$V^\pi(s) = E[\sum_{t=0}^{\infty} \gamma^t r_t | s_0 = s, \pi]. \qquad (1)$$

The optimal policy π^* is the one for which the value function is maximal for every state: $V^*(s) \geq V^\pi(s), \forall s, \pi$. The behavior of an agent, that is how it acts in each situation, is thus specified by its policy (and indirectly by the value function).

This research was partly funded by the EU FP7 project ILHAIRE (grant n^o 270780) and the Region Lorraine (France)

2.2. Modeling users with MDP

A user can be considered as a decision maker optimizing some unknown reward function and thus can be modeled thanks to an MDP. We consider here a generic slot filling task. The state space consists of the last dialogue manager act as well as values of different slots (if they have been provided by the user and/or confirmed by the DM). An action consists in greeting or closing the dialogue, as well as providing one or more slots or confirming a slot. Transition probabilities depend on the considered DM, it is not necessary to define them formally (as we will work on a simulation-based model-free basis). The reward function encodes the preferences of the user, it is also unknown. One can learn this reward function to build a user simulator [13] (see also this reference for more details on the state and action spaces) using inverse reinforcement learning [14]. Yet, it might not be necessary as shown hereafter.

2.3. Quantizing behaviors

Thanks to this MDP approach, each user is actually a policy. The initial state s_0 is always the same: there is no system act and all slots are empty. Let's assume that we have K different users, and therefore K different policies π_k. We assume that all users in a cluster try to optimize the same (unknown) reward function R. To each of these policies is associated the value function for the initial state $v^k = V^{\pi_k}(s_0)$. This quantity could be used to cluster user behaviors (quantified by policies), as it differs for each policy. However, this is not practical, as the reward function is unknown.

To remove this reward dependency, we assume that the reward function can be expressed as a linear combination of p predefined basis functions $\phi(s) = (\phi_1(s), \ldots, \phi_p(s))^T \in \mathbb{R}^p$, the set of associated weights being concatenated in the parameter vector $\theta = (\theta_1, \ldots, \theta_p)^T$:

$$R(s) = \sum_i \theta_i \phi_i(s) = \theta^T \phi(s). \tag{2}$$

The value v^k can be rewritten as follows:

$$v^k = E[\sum_{t=0}^{\infty} \gamma^t r_t | s_0, \pi_k] \tag{3}$$

$$= E[\sum_{t=0}^{\infty} \gamma^t \theta^T \phi(s_t) | s_0, \pi_k] \tag{4}$$

$$= \theta^T \mu^{\pi_k} \text{ with } \mu^{\pi_k} = E[\sum_{t=0}^{\infty} \gamma^t \phi(s_t) | s_0, \pi_k]. \tag{5}$$

The quantity μ^{π_k}, called *feature expectation*, does not depend on the reward function, only on the policy π_k. We propose to use this measure for clustering user behaviors.

Let's assume that the dialogue corpora contains N dialogues, each one of length H_n. For $1 \leq n \leq N$, we can compute vectors

$$\delta_n = \sum_{t=0}^{H_n} \gamma^t \phi(s_t). \tag{6}$$

Each vector δ_n therefore represents succinctly the trajectory (dialogue) information. Moreover, δ_n is an unbiased estimate of μ_{π_k} for some user (represented by the policy π_k). We assume that the intra-variability of one user (due to randomness of transitions) is lower than the inter-variability between users. Finally, one can use any vector quantization algorithm to cluster behaviors (by applying the clustering algorithms to vectors δ_n).

3. EXPERIMENTS

In this section user behavior clustering in the space spanned by δ is outlined. Following which a simple experiment using hand-crafted user simulation for a 3-slot restaurant information dialogue system is proposed to determine the effectiveness of clustering user behavior based on δ. Finally clustering of user behavior using data generated by actual human users while interacting with a 12-slot restaurant information dialogue system is outlined.

3.1. User behavior clustering using discounted feature vectors

As discussed in Section 2.3 discounted feature vectors of a trajectory provides a succinct representation of user behavior. Since the discounted feature vector in itself is a vector, it is a point in some high dimensional space $\mathcal{F}$. Thus a direct way to cluster user behaviors is to perform clustering on this space $\mathcal{F}$. In this paper, we consider the simple K-means algorithm for clustering the discounted feature vectors. It is important to note that this method is different than comparing directly any two trajectories since they may not be of equal length and the order of visited states and performed actions may vary from one trajectory to another.

3.2. Clustering user behavior of 3-slot dialogue problem

In order to explore the possibility of clustering, a simple experiment is proposed. The task studied here is a form-filling task about restaurant information, similar to the one studied in [15]. The primary task of the dialogue system is to give information about restaurants based on user preferences. The dialogue problem is composed of 3 slots: location, cuisine and price-range of the restaurant. The hand-crafted dialogue policy tries to obtain the values for these slots while interacting with the user or user simulator.

The user behavior is casted as an MDP (User-MDP) similar to that of dialogue management. The state of the MDP is represented by the Information State paradigm [16]. It is important to note that the state here is a summary of the dialogue course from a user's perspective. Apart from encoding the exchange of information between the user and the dialogue manager, the user state also includes the most recent system action. The state representation of User-MDP: {System-Act} {Slot1} {Slot2} {Slot3}, where the system-act field takes values in 0:13 (which deals with filling and confirming the 3 slots). Slot1, Slot2, Slot3 fields take values in 0:2; *i.e.* (0) the slot is empty (never provided by the user), (1) the slot has been provided by the user, (2) the slot is confirmed. The action space of User-MDP includes the following 10 user acts: remain silent (Silent), provide-all-values (AllSlots), provide-one-value (OneSlot: 3 actions), confirm slot value (Confirm: 3 actions), negate slot value (Negate: 3 actions) and hangup (CloseDialogue). The discount factor of the User-MDP is set to 0.95. Two simple hand-crafted user behaviors for User-MDP are specified as shown in Table 1. User behavior 1 outlines a proactive or expert user who prefers to furnish *all* the required information as soon as the system greets. Whereas user behavior 2 outlines a novice user who prefers to furnish *only* the information requested by the system. The user behaviors are purposely made simple so that the clustering results can be visually interpreted.

3.3. Clustering user behavior of 12-slot dialogue problem

A second experiment aims at answering the following concerns: (i) can the proposed method for user behavior clustering work on data collected from human users? (ii) can the method scale for large state space real world problem? (iii) are the resulting clusters distinct

Table 1. *Hand-crafted users behaviors*

SystemAct	UserActs 1 (probab.)	UserActs 2 (probab.)
Greet	Silent (0.1) AllSlots (0.9)	Silent (0.9) AllSlots (0.1)
AskSlot	OneSlot (0.95) AllSlots (0.05)	OneSlot (0.95) AllSlots (0.05)
Explicit-Conf	Confirm (1.0)	Confirm (1.0)
Implicit-Conf	OneSlot (0.9) Negate (0.1)	OneSlot (0.9) Negate (0.1)
CloseDialogue	Silent (1.0)	Silent (1.0)

from each other? For this experiment a 12-slot restaurant information dialogue system is considered (same as in [11]). The data used for clustering is generated from interaction between a hand-crafted dialogue policy and human users. This problem is a large real world dialogue problem similar to the 3-slot problem, but also includes several other user preferences such as type of drinks served, type of music played, number of stars *etc.*

The task of user interaction is again casted as an MDP. The state of the User-MDP is: {System-Act} {Preference} {Goal} {Annoyance} {Correctness} {ChangeIntention}, where system-act can takes values between 0 and 10, preference and goal fields can take values from 0 to 2 (0 means no information is furnished or obtained, 1 meaning partial exchange and 2 means all constraints have been informed or all requested information have been obtained from the system), while annoyance is a boolean value which indicates whether the user is annoyed or not, correctness field indicates whether the information presented by the system is what user actually conveyed and changeIntention field indicates whether the system can find results for the user constraints or not.

4. RESULTS AND DISCUSSION

The primary goal of the 3-slot experiment (see Section 3.2) is to determine the possibility of user behavior clustering using discounted feature vectors. The state-action space is spanned by 3780 features. A set of 3000 dialogue trajectories are generated by randomly choosing one of the two user behaviors. Of the 3000 dialogue trajectories 1544 belongs to the expert user and 1456 belongs to the novice users. Discounted feature vector corresponding to each of these 3000 trajectories are computed. K-means method with K set to 2 (since it is known that only two user behaviors are simulated) is used for clustering the discounted feature vectors. Euclidean distance measure is used to cluster the user behaviors for this experiment.

Upon clustering it is observed that the two clusters have 1549 and 1451 elements. The frequencies of user action selection for the hand-crafted user behaviors as well as for the two clustered behaviors are shown in Figure 1. It can be observed from the histogram that the expert user behavior correlates well with clustered user behavior 1 and same is the case with the novice user and clustered user behavior 2. Since the clustered user behaviors correlates with the respective hand-crafted user behaviors we can claim that clustering user behavior using their discounted feature vector is a viable option.

Our second experiment was conducted on a large real world problem (12 slots) using data collected from human users. A total of 480 dialogue trajectories (weakly annotated from user perspective) generated from interaction between human users and a hand-crafted dialogue manager. The associated discounted feature vectors of all these trajectories are computed. Assuming that these vectors *i.e.* these points in the feature space $\mathcal{F}$ are linearly separable K-means method is applied for clustering.

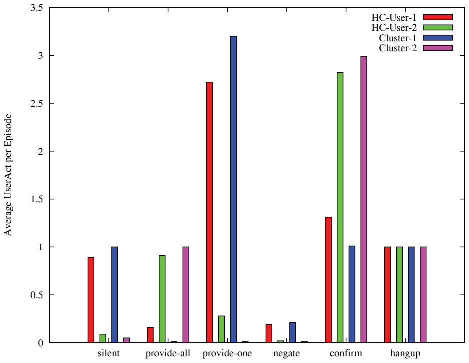

Fig. 1. *Frequency of user actions per episode for the 3-slot problem*

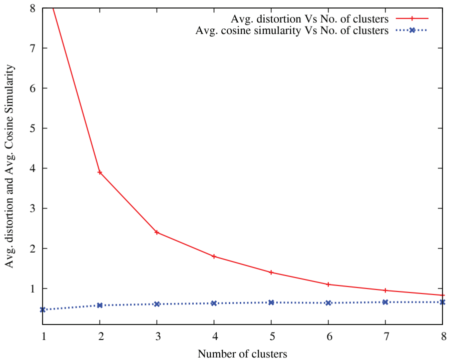

Fig. 2. *Cumulative distortion with varying number of clusters*

Since the data is generated by human users, the number of different types of user behaviors and hence the number of clusters is not known beforehand. In machine learning one way of choosing the number of clusters, *i.e.* the value of K, is to observe the average cumulative distortion with varying number of clusters. In addition to using average distortion, a measure of intra-cluster cosine similarity (*i.e.* normalized dot product between two discounted feature vectors) is also used to decide the number of clusters: $\cos(A,B) = \frac{A.B}{\|A\|.\|B\|}$.

From Figure 2, where the average distorsion and cosine simularity among clusters is computed for several values of K, it can be observed that for K $>$ 4, the distortion as well as the intra-cluster cohesion do not improve significantly. Thus for clustering the data collected from human users K is set to 4. As it can be observed from the User-MDP state the data used for clustering is symbolic, hence K-means clustering is performed using cosine similarity measure rather than Euclidean distance. The 480 trajectories are grouped as 4 clusters each with 35, 124, 114 and 207 elements.

The primary problem while dealing with human user data unlike the simple simulated user behavior (as for the 3-slot problem) is that the behaviors grouped under various clusters may not be easily differentiated. Thus we propose to use Kullback-Leibler (KL) divergence measure [17] to ensure the correctness of the clustering. The KL divergence measures the dissimilarity between two probability distributions. The KL divergence between two distributions P and Q is defined by: $D_{KL}(P||Q) = \sum_{i=1}^{M} p_i log(\frac{p_i}{q_i})$, where p_i (resp. q_i) is the frequency of dialogue act a_i in the histogram of distribution P (resp. Q). KL divergence close to 0 corresponds to identical

Table 2. *Inter-cluster Kullback-Leibler divergence*

Behavior comparison	KL-Divergence
Cluster-1 vs Cluster-2	2.52
Cluster-1 vs Cluster-3	2.51
Cluster-1 vs Cluster-4	1.99

Table 3. *Intra-cluster cosine similarity*

Behavior-Type	Cosine similarity
Cluster-1	0.68
Cluster-2	0.60
Cluster-3	0.55
Cluster-4	0.70

behavior and larger non zero values correspond to significant divergence from each other. Based on this we make an assumption: if the behavior of the users in two different clusters are truly distinct then there should be a noticeable divergence between the user acts distributions. Thus KL divergence is computed using the frequency of user action selection across all the clusters. As it can be observed from Table 2 the clusters formed and hence the user behaviors have significant KL divergence. Thus it tends to confirm that every cluster indeed represents a specific user behavior which is different from other clusters.

It is also important to establish that there is good cohesion among the elements within a cluster, *i.e.* to show that intra-cluster cohesion is non negligeable. In order to measure the intra-cluster cohesion we used cosine similarity between the centroid of the cluster and the elements within the cluster. Since discounted feature vector is indeed a vector, cosine similarity is an appropriate measure to compute the intra-cluster similarity. The value for cosine similarity while comparing 2 elements can range form -1 (meaning totally opposite elements), 0 (meaning independent elements),1 (meaning identical elements). Cosine similarity is computed for each cluster by measuring the similarity between the cluster's centroid and its own elements. Without clustering, the cosine similarity of the complete set of data to the average vector is 0.4. Table 3 shows that the cosine similarity of all 4 clusters have values closer to 1 (which means most vector within the cluster point to the same direction). This reveals strong intra-cluster cohesion or similarity between discounted feature vectors/user behaviors.

5. CONCLUSION

Identifying and/or simulating the different user behaviors is a crucial task for learning as well as evaluating adaptive dialogue strategies. However until now, compact means for representing user trajectory/behavior such as discounted feature vector are not used in the dialogue management domain. Representing the user behavior in the form of discounted feature vector gives an opportunity to further explore schemes like automatic clustering of different user behaviors. The experimental results for the 3-slot dialogue problem validated the use of discounted feature vector for user clustering by grouping the two hand-crafted trajectories appropriately. Experimental results from the 12-slot dialogue problem showed that even human users can be clustered and the resulting clusters can be shown distinct and coherent.

With regard to future directions of work, once the user behaviors are clustered it is possible to build user simulations which can simulate different types of users. Such efforts will yield more than one user simulations with distinct behaviors and hence can be used for (adaptive) dialogue policy optimization and evaluation. Another interesting task to be explored is to find automatic measure to quantify a specific user behavior. Currently we have clustered the behaviors, but it would be interesting to see in what way one cluster is distinct from others. Understanding the distinctive features of clusters can help generating data missing in the dialogue corpora. Also, in this paper, we have used the K-means algorithm which is sensible to the fact that some clusters may contain a low amount of data. Other vector quantization algorithms are less sensible to non-uniform distributions of the data among clusters such as the neural gas algorithm.

6. REFERENCES

[1] R. Sutton and A. Barto, *Reinforcement Learning: An Introduction*, The MIT Press, 3rd edition, March 1998.

[2] E. Levin and R. Pieraccini, "Using markov decision process for learning dialogue strategies," in *Proc. ICASSP'98, Seattle (USA)*, 1998.

[3] O. Lemon and O. Pietquin, "Machine learning for spoken dialogue systems," in *Proc. of InterSpeech'07*, Belgium, 2007.

[4] O. Pietquin, M. Geist, S. Chandramohan, and H. Frezza-Buet, "Sample-Efficient Batch Reinforcement Learning for Dialogue Management Optimization," *ACM Transactions on Speech and Language Processing*, vol. 7, no. 3, pp. 7:1–7:21, May 2011.

[5] O. Pietquin and T. Dutoit, "A probabilistic framework for dialog simulation and optimal strategy learning," *IEEE Transactions on Audio, Speech & Language Processing, 14(2): 589-599*, 2006.

[6] J. Schatzmann, B. Thomson, K. Weilhammer, H. Ye, and S. Young., "Agenda-based User Simulation for Bootstrapping a POMDP Dialogue System," in *Proc. of HLT/NAACL*, 2007.

[7] K. Georgila, J. Henderson, and O. Lemon, "Learning User Simulations for Information State Update Dialogue Systems," in *Eurospeech*, 2005.

[8] W. Eckert, E. Levin, and R. Pieraccini, "User Modeling for Spoken Dialogue System Evaluation," in *Proc. of ASRU*, 1997, pp. 80–87.

[9] O. Pietquin, "Consistent goal-directed user model for realistic man-machine task-oriented spoken dialogue simulation," in *Proc. of ICME'06*, Toronto (Canada), July 2006, pp. 425–428.

[10] J. Schatzmann, K. Weilhammer, M. Stuttle, and S. Young, "A survey of statistical user simulation techniques for reinforcement-learning of dialogue management strategies," *Knowledge Engineering Review, vol. 21(2), pp. 97–126*, 2006.

[11] S. Keizer, M. Gasic, F. Jurcicek, F. Mairesse, B. Thomson, K. Yu, and S. Young, "Parameter estimation for agenda-based user simulation," in *Proc. of SigDial 2010*, 2010, pp. 116–123.

[12] V. Rieser and O. Lemon, "Cluster-based User Simulations for Learning Dialogue Strategies," in *Proc. of Interspeech/ICSLP 2006*, 2006.

[13] S. Chandramohan, M. Geist, F. Lefèvre, and O. Pietquin, "User Simulation in Dialogue Systems using Inverse Reinforcement Learning," in *Proc. of Interspeech 2011*, Florence (Italy), 2011.

[14] P. Abbeel and A. Ng, "Apprenticeship learning via inverse reinforcement learning," in *Proc. of ICML*, 2004.

[15] O. Lemon, K. Georgila, J. Henderson, and M. Stuttle, "An ISU dialogue system exhibiting reinforcement learning of dialogue policies: generic slot-filling in the TALK in-car system," in *Proc. of EACL'06*, Morristown, NJ, USA, 2006.

[16] S. Larsson and D. Traum, "Information state and dialogue management in the TRINDI dialogue move engine toolkit," *Natural Language Engineering, vol. 6, pp 323–340*, 2000.

[17] S. Kullback and R. A. Leibler, "On information and sufficiency," *Ann. Math. Statist.*, vol. 22, no. 1, pp. 79–86, 1951.