HAL
open science

Regroupement non-supervisé d'utilisateurs par leur comportement pour les systèmes de dialogue parlé

Senthilkumar Chandramohan, Matthieu Geist, Fabrice Lefèvre, Olivier Pietquin

To cite this version:

Senthilkumar Chandramohan, Matthieu Geist, Fabrice Lefèvre, Olivier Pietquin. Regroupement non-supervisé d'utilisateurs par leur comportement pour les systèmes de dialogue parlé. Journées Francophones sur la planification, la décision et l'apprentissage pour le contrôle des systèmes - JFPDA 2012, May 2012, Villers-lès-Nancy, France. 16 p. hal-00736205

HAL Id: hal-00736205

https://inria.hal.science/hal-00736205

Submitted on 27 Sep 2012

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Regroupement non-supervisé d'utilisateurs par leur comportement pour les systèmes de dialogue parlé ⋆

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

[1] Supélec - Equipe IMS (Campus de Metz)
2 rue Edouard Belin, 57070 Metz
nom.prenom@supelec.fr

[2] UMI 2958 (GeorgiaTech - CNRS)
2 rue Edouard Belin, 57070 Metz
[3] LIA-CERI, Université d'Avignon
BP 1228, 84911 Avignon Cedex 9
Fabrice.Lefevre@univ-avignon.fr

Résumé : Les systèmes de dialogue parlé sont des interfaces naturelles pour l'interaction homme-machine. La conception de stratégies d'interaction s'adaptant à l'interlocuteur est essentielle pour assurer le caractère naturel de l'interface. Depuis quelques années, des méthodes d'apprentissage automatique pour l'optimisation de ces stratégies, particulièrement l'apprentissage par renforcement, sont utilisées de plus en plus largement et font maintenant partie de l'état de l'art dans le domaine. Néanmoins, de grandes quantités de données sont souvent nécessaires pour entraîner ces algorithmes menant à une optimisation de l'interaction. Pour palier le déficit de données souvent rencontré en pratique, mais aussi pour évaluer la qualité des systèmes développés, des systèmes de simulation d'utilisateurs sont souvent utilisés. On utilise alors des jeux de données de dialogue annotés plus petits permettant d'apprendre des comportements simulés. Néanmoins l'annotation n'est généralement pas réalisée du point de vue utilisateur mais plutôt du point de vue machine, ainsi le but poursuivi par l'utilisateur est souvent inconnu. Par ailleurs, toute la base de données est la plupart du temps utilisée pour entraîner un simulateur qui produit ainsi des comportements moyens pouvant éventuellement ne correspondre à aucun comportement particulier rencontré dans la base. Cet article propose une méthode de regroupement non-supervisé des données (*clustering*) en fonction du comportement des utilisateurs. Cette segmentation des comportements se base sur l'interprétation des utilisateurs comme étant des processus décisionnels de Markov, de dynamique et/ou récompenses différentes. De cette manière plusieurs groupes d'utilisateurs sont distingués en fonction de dialogues complets et pas de comportements locaux dans des contextes particuliers.

Résumé : 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.

Mots-clés : Apprentissage par renforcement, Systèmes de dialogue parlé, regroupement

1 Introduction

Les systèmes de dialogue parlé (*Spoken Dialogue Systems* - SDS) sont des systèmes permettant à un utilisateur humain d'interagir avec des machines (ordinateurs, robots) par le biais de la parole. Ce sont

⋆. Cette recherche a été partiellement financée par le projet européen FP7 FET OPEN ILHAIRE (bourse n° 270780) et par la Région Lorraine

des systèmes relativement complexes nécessitant le développement de divers sous-systèmes permettant de traiter le signal de parole pour en extraire le sens. Cela se fait généralement en deux macro-étapes : une étape de transcription automatique puis une étape d'analyse sémantique de la transcription. En fonction de ce qui a été compris par la machine, un retour est fourni à l'utilisateur par le biais d'une voix synthèse. Néanmoins, il ne suffit pas de combiner des modules de compréhension de la parole avec des synthétiseurs vocaux pour obtenir un système de dialogue. En effet, un sous-système de gestion de l'interaction vocale est nécessaire pour décider du séquencement du dialogue, c'est à dire de quoi dire étant donné le contexte. Ce gestionnaire de dialogue (*Dialogue Manager* - DM) peut être vu comme un agent mettant en œuvre un processus de décision séquentiel visant à rendre l'interaction avec l'utilisateur naturelle et efficace.

Le développement d'un tel module de gestion a été transposé dans le cadre de l'apprentissage par renforcement (Sutton & Barto, 1998) depuis la fins des années 90 (Levin & Pieraccini, 1998; Lemon & Pietquin, 2007). Pour ce faire, le dialogue homme-machine est vu comme un Processus Décisionnel de Markov (*Markov Decision Process* - MDP) (Bellman, 1957). Bien qu'il ait été récemment proposé d'utiliser des méthodes d'apprentissage par renforcement efficace pour apprendre des stratégie d'interaction à partir de jeux de données fixes (Pietquin *et al.*, 2011), il est souvent d'usage d'entraîner ces système par le biais d'interactions simulées du fait de la difficulté d'obtenir des corpus de données annotés dans ce domaine (Pietquin & Dutoit, 2006; Schatzmann *et al.*, 2007; Georgila *et al.*, 2005). Les modèles statistiques permettant de simuler l'interaction sont eux-mêmes souvent entraînés sur des corpus de données mais de taille suffisamment petite pour être collectés et annotés. La simulation d'utilisateurs est aussi souvent utilisée dans le but de qualifier les performances d'un système avant de le mettre à disposition de véritables utilisateurs (Eckert *et al.*, 1997).

La plupart du temps, les simulateurs d'utilisateur sont des modèles stochastiques entraînés sur des données dont les annotations sont assez sommaires (Pietquin, 2006; Schatzmann *et al.*, 2006; Keizer *et al.*, 2010). Un corpus de données de dialogue contient un ensemble de dialogues provenant de différents utilisateurs qui peuvent interagir de manière très différentes. Par exemple, il arrive souvent que des utilisateurs habitué à l'interaction avec le système utilise cette connaissance *a priori* pour naviguer plus rapidement et couper court à l'interaction. En revanche, une personne plus novice aura tendance à attendre les instructions de la machine pour savoir comment l'utiliser correctement. En d'autres termes, la distributions des dialogues (et des comportements d'utilisateurs) présents dans un corpus est généralement multimodale. Pourtant, il est très rare (c'est un euphémisme) que les corpora soient annoter pour refléter cette réalité et les comportements différents sont ignorés. Ainsi, le modèle est entraîné sur l'ensemble des données, résultant en un modèle générique moyen qui ne correspond en réalité à aucun comportement réel. Ensuite, l'entraînement d'un gestionnaire de dialogue par simulation avec un tel modèle (ou directement sur les données) revient à réaliser cet entraînement pour un utilisateur non représentatif. La qualité de la simulation influe donc, évidemment, sur la qualité de la stratégie d'interaction apprise (Schatzmann *et al.*, 2005).

Nous pensons donc qu'entraîner un système de gestion de dialogue à partir de simulateurs représentant différents types de comportement conduirait à de meilleures performances et donc à des interactions plus naturelles et efficaces. Néanmoins, annoter les corpus à disposition selon le type d'utilisateur serait un travail fastidieux voire impossible. C'est pourquoi nous proposons dans cet article de réaliser un regroupement automatique non-supervisé des dialogues présents dans les corpus en fonction du type de comportement des utilisateurs. Ce regroupement se fait sans modèle préalable des utilisateurs. La difficulté principale tient au fait que les dialogues que l'on cherche à regrouper sont des séquences d'interactions dont la longueur est variable (alors que la longueur n'est pas forcément représentative du comportement). Ainsi, les séquences elles-mêmes ne peuvent être utilisées directement dans un algorithme de quantification vectorielle standard par exemple (les vecteurs devant être de même taille). Rieser & Lemon (2006) proposent aussi de réaliser un regroupement en classes homogènes. Néanmoins, pour éviter le problème du à la longueur variable des séquences, un comportement spécifique est analysé (les stratégies de clarification) et le contexte est restreint à une longueur fixe.

Chandramohan *et al.* (2011) proposent d'exprimer le comportement d'un utilisateur comme un processus de décision séquentielle pour réaliser un simulateur d'utilisateur basé sur le paradigme d'apprentissage par renforcement inverse (Abbeel & Ng, 2004). Suivant le même raisonnement, nous proposons une méthode de représentation des dialogues sous la forme d'un vecteur tenant compte des trajectoires correspondantes dans l'espace d'états du MDP sous-jacent permettant ainsi de réaliser un regroupement par quantification vectorielle.

2 Regroupement de comportements

Dans cette section, nous rappelons les notions de Processus Décisionnels de Markov et nous montrons comment un utilisateur de système de dialogue peut être modélisé comme un MDP. Nous expliquons ensuite comment ceci peut permettre de représenter un dialogue sous la forme d'un vecteur pouvant être utilisé pour réaliser un regroupement non-supervisé.

2.1 Processus Décisionnels de Markov

Les processus décisionnels de Markov (PDM) constituent un cadre fréquemment utilisé pour formaliser les processus de décisions séquentielles impliquant un agent en interaction avec un environnement ou un système dynamique. Un MDP est défini comme un tuple $\{S, A, P, R, \gamma\}$ où S est l'espace d'état, A l'espace d'action, $P : S \times A \rightarrow \mathcal{P}(S)$ un ensemble de probabilités de transitions Markoviennes, $R : S \rightarrow \mathbb{R}$ une fonction de récompense et γ un facteur pénalisant les récompenses à long terme.

Une politique $\pi : S \rightarrow \mathcal{P}(A)$ définit comment choisir une action dans chaque état. A chaque pas de temps t, Les système est supposé dans un état s_t. L'agent choisit alors une action a_t en fonction d'une politique $\pi(.|s_t)$ et le système transite de manière stochastique vers un état s_{t+1} en suivant la probabilité $p(.|s_t, a_t)$. L'agent reçoit une récompense r_t, qui constitue une information locale à propos de la qualité du contrôle. La qualité d'une politique π est évaluée par une fonction de valeur, définie pour chaque état comme étant le gain cumulé espéré en commençant une interaction dans l'état s et en suivant ensuite la politique π :

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

La politique optimale π^* est celle pour laquelle la fonction de valeur est maximale pour chaque état : $V^*(s) \geq V^{\pi}(s), \forall s, \pi$. Le comportement d'un agent, c'est à dire son choix d'action pour chaque état, est donc spécifié par sa politique (et indirectement par sa fonction de valeur).

2.2 Modélisation des utilisateurs par des PDM

Un utilisateur de système de dialogue peut être considéré comme un système de prise de décisions séquentielles qui optimise une récompense inconnue. Il peut donc être modélisé sous la forme d'un PDM. Nous considérons ici un problème de remplissage de formulaire par interaction vocale. La représentation d'état inclut la dernière action du système de gestion du dialogue (question, assertion, demande de confirmation *etc.*) ainsi que les valeurs des différentes informations du formulaire déjà transmises durant le dialogue (ainsi qu'une valeur booléenne pour chaque information indiquant si elle a été confirmée). Les actions de l'utilisateur sont la réponse à l'invite, fermer le dialogue, fournir une information ou la confirmer. La fonction de récompense encore les préférences de l'utilisateur en termes d'interaction. Cette fonction est inconnue. Cette fonction peut être apprise comme cela a été fait dans le cas de la simulation d'utilisateur dans Chandramohan *et al.* (2011) (cette référence donne aussi plus de détail sur la constitution de l'espace d'états). Pour ce faire, le paradigme d'apprentissage par renforcement inverse Abbeel & Ng (2004) est mis en œuvre. Néanmoins, dans le but d'un regroupement non supervisé, il n'est pas nécessaire d'apprendre cette récompense comme nous allons le voir ci-après.

2.3 Quantification du comportement

Grâce à la modélisation sous forme de PDM, chaque utilisateur est caractérisé par sa politique (son comportement). Dans le cadre d'un dialogue ayant pour but le remplissage d'un formulaire, l'état initial s_0 est toujours le même : il n'y a pas d'action précédente du système de dialogue et aucune information n'a été transmise ni confirmée. Supposons qu'il existe K groupes d'utilisateurs différents dans la base de données, et donc K politiques différentes π_k. Nous supposons aussi que tous les utilisateurs dans un groupe optimisent la même récompenses (inconnue) R. A chaque politique présente dans le corpus de donnée est donc associée une fonction de valeur pour l'état initial $v^k = V^{\pi_k}(s_0)$. Cette quantité pourrait être utilisée pour réaliser le regroupement (des valeurs similaires correspondent à des politiques similaires). Néanmoins, ceci n'est pas très pratique puisqu'il est nécessaire d'avoir accès à la fonction de récompense de l'utilisateur.

Pour s'affranchir de la dépendance à la récompense, nous allons supposer que celle-ci peut s'exprimer comme une combinaison linéaire de p fonction de base pré-définies $\phi(s) = (\phi_1(s), \ldots, \phi_p(s))^T \in \mathbb{R}^p$. Les poids de cette combinaison peuvent être concaténés dans un vecteur de paramètres $\theta = (\theta_1, \ldots, \theta_p)^T$:

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

Les valeurs v^k peuvent être réécrites selon :

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

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

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

La quantité μ^{π_k}, appelée *feature expectation*, ne dépend pas de la fonction de valeur mais seulement de la politiqueπ_k. Nous proposons donc d'utiliser cette quantité pour réaliser une quantification vectorielle de la base de donnée et donc effectuer un regroupement non-supervisé.

Si nous supposons que le corpus de dialogue contient N dialogues, chacun de longueur H_n. Pour $1 \leq n \leq N$, nous pouvons calculer les vecteurs :

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

Chaque vecteur δ_n représente donc de manière compacte une trajectoire dans l'espace d'état (donc un dialogue). De plus, δ_n est une estimation non-biaisée de μ_{π_k} pour une utilisateur quelconque (représenté par la politique π_k). Nous supposons que la variabilité de comportement interne à un groupe d'utilisateur (due au caractère aléatoire des transitions) est plus faible que la variabilité inter-groupes. Enfin, il est possible d'utiliser n'importe quelle méthode de quantification vectorielle pour réaliser le regroupement dans l'espace dans lequel évoluent les vecteurs δ_n.

3 Expériences

Dans cette section le regroupement des comportements d'utilisateurs dans l'espace défini par δ est décrit. À la suite de quoi une expérience simple utilisant un utilisateur simulé défini manuellement pour un système de dialogue d'information sur des restaurants reposant sur 3 critères est proposée pour montrer l'effectivité de la proposition. Finalement le regroupement de comportements utilisateurs est décrit dans le cas de données obtenues auprès d'utilisateurs réels en interaction avec un système d'information sur des restaurants avec 12 critères.

3.1 Regroupement d'utilisateurs par leur comportement base sur les vecteurs d'observations pondérées de trajectoire

Comme évoquer dans la section 2.3 les vecteurs d'observations pondérées de trajectoire fournissent une représentation compacte du comportement de l'utilisateur. Dans la mesure où il s'agit bien de vecteurs, ils représentent des points particuliers dans un espace de grande dimension $\mathcal{F}$. Ainsi une manière directe pour grouper des utilisateurs est de réaliser le regroupement automatiquement dans cet espace $\mathcal{F}$. Dans ce papier nous considérons l'algorithme des K-moyennes pour réaliser le regroupement des vecteurs de trajectoire. Il est important de noter que cette méthode est différente de la comparaison directe de 2 trajectoires dans la mesure où elles peuvent ne pas être de longueur identique, de même l'ordre des états visités et des actions réalisées peut varier d'une trajectoire à l'autre.

TABLE 1 – *Comportements d'utilisateurs définis manuellement*

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)

3.2 Regroupement de comportements utilisateurs pour un problème simple à 3 critères

Afin de montrer la possibilité réelle du regroupement, une expérience simple est d'abord tentée. La tâche correspond à un remplissage de formulaire pour obtenir des informations sur des restaurants, comparable à celle proposée dans Lemon *et al.* (2006). La tâche principale du système de dialogue est de donner des informations au sujet de restaurants, basées sur les préférences de l'utilisateur. Le problème au niveau du dialogue est composé de 3 critères (ou champs) : localisation, type de cuisine et gamme du restaurant. La stratégie de dialogue établie manuellement tente d'obtenir les valeurs de ces critères durant l'interaction avec l'utilisateur.

Le comportement de l'utilisateur est modélisé par un MDP (User-MDP) comparable à celui du gestionnaire de dialogue. L'état du MDP est représenté dans le cadre du paradigme de l'état d'information (Information State paradigm Larsson & Traum (2000)). Il est important de noter de dans notre cas l'état est un résumé de la situation courante du dialogue du point de vue de l'utilisateur. En plus de représenter l'état des échanges entre l'utilisateur et le système, l'état utilisateur inclut aussi la dernière action système. La représentation complète de l'état utilisateur dans User-MDP est donc : {System-Act} {Slot1} {Slot2} {Slot3} où le champ action système prend ses valeurs dans 0 :13. Les champs *Slot* prennent des valeurs dans 0 :2 : (0) le critère est vide (non encore fourni par l'utilisateur), (1) le critère a été fourni par l'utilisateur et (2) le critère a été confirmé. L'espace des actions pour User-MDP comprend 10 actions utilisateur : rester silencieux (Silent), fournir tous les critères (AllSlots), fournir une valeur (OneSlot, 3 actions possibles), confirmer une valeur (Confirm, 3 actions possibles), nier une valeur (Negate, 3 actions possibles) et raccrocher (CloseDialogue). Le facteur d'oubli de User-MDP est fixé à 0,95. Deux comportements d'utilisateurs simples sont spécifiés comme indiqués dans le tableau 1. Le comportement 1 correspond à un utilisateur expérimenté qui tente de fournir tous les critères requis dès qu'il en a l'occasion. Tandis que le comportement 2 représente mieux un utilisateur novice qui préfère fournir seulement l'information demandée par le système au fur et à mesure. Les comportements d'utilisateur sont volontairement rendus très distincts afin de permettre une interprétation aisée du résultat du regroupement.

3.3 Regroupement de comportement d'utilisateur pour un problème à 12 critères

Une seconde expérience a pour but de répondre aux questions suivantes : (1) est-ce que la méthode proposée peut fonctionner sur des données collectées avec des utilisateurs réels ? (2) est-ce que la méthode proposée est viable pour des problèmes réels de très grande taille ? (3) et enfin est-ce que les regroupements obtenus sont bien distincts les uns des autres ? Pour cette expérience un système de dialogue d'information sur les restaurants à 12 critères est utilisé (le même que dans Keizer *et al.* (2010)). Les données utilisées pour le regroupement ont été collectées durant les interactions entre des humains et un système suivant un stratégie définie manuellement. Cette tâche est un problème de dialogue du monde réel comparable au problème à 3 critères mais incluant d'autres préférences comme le type de boissons servies, l'ambiance musicale, le nombre de d'étoiles...

La tâche d'interaction utilisateur est une fois de plus modélisé par le biais d'un MDP. L'état de User-MDP est donné par : {System-Act} {Preference} {Goal} {Annoyance} {Correctness} {ChangeIntention}, où l'action système peut prendre des valeurs entre 0 et 10, les critères préférence et goal entre 0 et 2 (0 signifie pas d'information fournie, 1 signifie échange partiel et 2 signifie toutes les contraintes ont été données ou toutes les informations ont été obtenues par le système). Annoyance (gêne) est une valeur booléenne qui indique si l'utilisateur est gêné ou pas (sous-entendu "par le fonctionnement imparfait du système"),

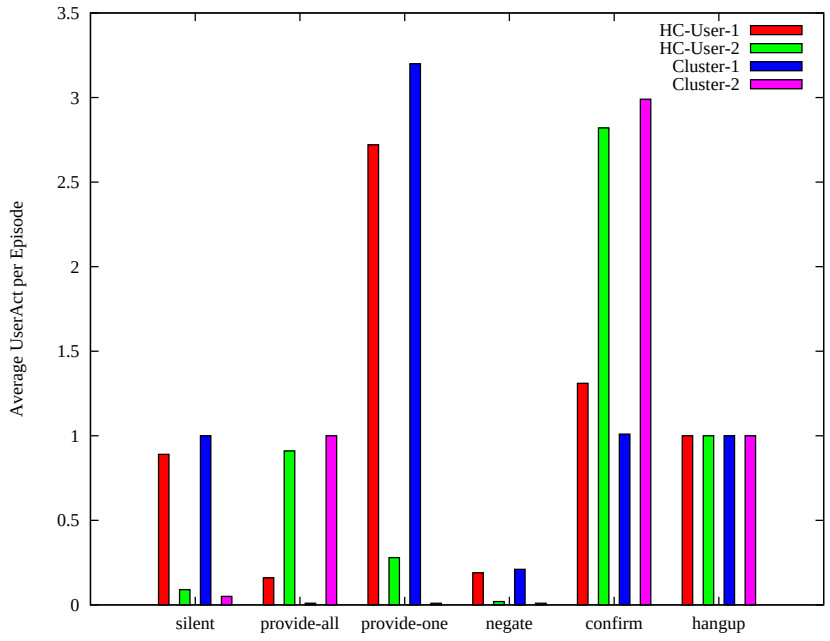

FIGURE 1 – *Fréquences d'actions utilisateurs par épisode pour le problème à 3 critères*

Correctness (validité) indique si l'information présentée par le système est bien ce que l'utilisateur a transmis, enfin changeIntention permet de prendre en compte la capacité du système à satisfaire les contraintes actuelles de l'utilisateur ou non.

4 Résultats et discussion

Le but premier de l'expérience à 3 critères (voir section 3.2) est de déterminer la possibilité de regroupement d'utilisateurs basé sur les vecteurs d'observations pondérées de trajectoire. L'espace état-action pour ce problème représente 3780 observations possibles. Un ensemble de 3000 trajectoires de dialogue sont générées en choisissant aléatoirement l'un des 2 comportements utilisateur disponibles. Parmi ces 3000 trajectoires, 1544 sont issues du modèle d'expert et 1456 du modèle novice. Les vecteurs pondérés correspondant à chacune des 3000 trajectoires sont calculés. La méthode des K-moyennes, avec K fixé à 2, est appliquée pour le regroupement. La distance euclidienne est utilisée dans cette expérience.

Après le regroupement on peut observer que les 2 ensembles obtenus sont formés de 1549 et 1451 éléments. La fréquence de sélection des actions utilisateur pour les comportements utilisateurs manuels sont donnés dans la figure 1, de même que pour les 2 ensembles obtenus automatiquement. On peut observer que l'histogramme de l'expert est fortement corrélé avec l'ensemble 1 et, de même, celui du novice avec l'ensemble 2. Dans cette expérience préliminaire on observe donc que la procédure a permis de retrouver et de séparer correctement les 2 comportements proposés en entrée.

Notre seconde expérience a été réalisée sur un problème réaliste de grande taille (12 critères) à partir de données collectées auprès d'utilisateurs réels. Un total de 480 dialogues (avec peu d'annotations disponibles du point de vue de l'utilisateur) entre des humains et un système défini manuellement sont disponibles. Les vecteurs d'observation correspondants sont calculés. Considérant que ces vecteurs (i.e. les points dans l'espace $\mathcal{F}$ sont linéairement séparables la méthode des K-moyennes est appliquée.

Dans la mesure où les données ont été générées par des humains, le nombre de types de comportements utilisateurs distincts et donc le nombre de groupes à obtenir n'est pas connu à l'avance. En apprentissage automatique, une manière de choisir le nombre de groupes (la valeur de K) est d'observer l'évolution de la distorsion moyenne cumulée avec la variation du nombre de groupes retenus. En plus de la distorsion

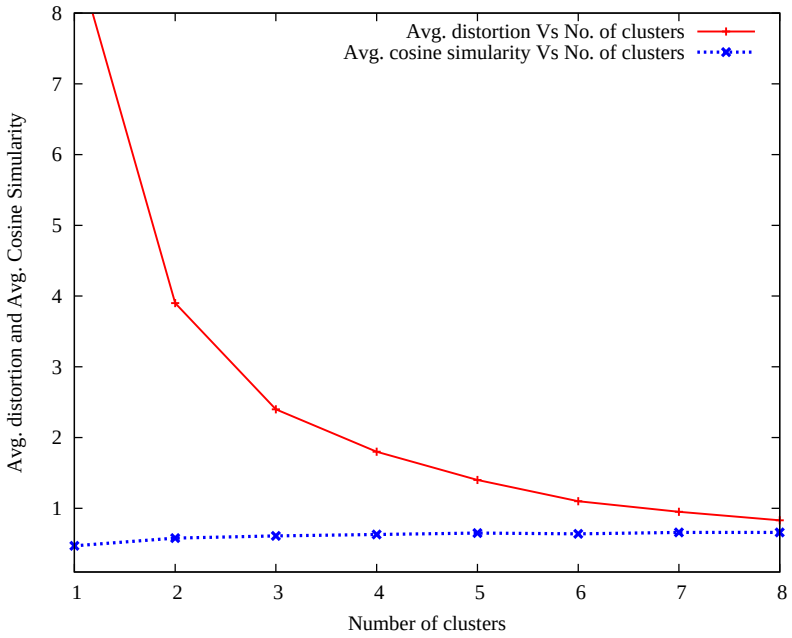

FIGURE 2 – *Distorsion cumulée selon le nombre d'ensembles voulus*

moyenne, une mesure de similarité cosinus intra-groupe (produit scalaire normalisé entre deux vecteurs d'observations pondérées) est aussi retenue pour décider du nombre de groupes : $\cos(A, B) = \frac{A.B}{\|A\|.\|B\|}$.

La figure 2, dans laquelle les distorsions moyennes et la similarité cosinus sont données pour plusieurs valeurs de K, permet d'observer que pour K $>$ 4 aucune des deux mesures n'évolue plus réellement. Ainsi pour le regroupement des donnes collectées auprès d'humains K est fixé à 4. Comme il peut être observé au travers des états de User-MDP les données utilisées pour le regroupement sont symboliques, et donc le regroupement par K-moyennes est réalisé en utilisant la mesure de similarité cosinus, plutôt que la distance euclidienne. Les 480 trajectoires sont regroupées dans 4 ensembles avec 35, 124, 114 et 207 éléments.

Le principal problème lorsque l'on s'intéresse au données collectées avec des humains, contrairement à aux comportements d'utilisateurs simulés, est que les comportements regroupés dans les différents ensembles ne sont pas facilement différentiables. C'est pourquoi nous proposons de recourir à la mesure de divergence de Kullback-Leibler (KL) Kullback & Leibler (1951) pour s'assurer de la qualité du regroupement obtenu. La divergence KL mesure la dissemblance entre distributions de probabilités. La divergence entre 2 distributions P et Q est définie par : $D_{KL}(P||Q) = \sum_{i=1}^{M} p_i log(\frac{p_i}{q_i})$, où p_i (resp. q_i) est la fréquence de l'acte de dialogue a_i dans l'histogramme de la distribution P (resp. Q). Une divergence KL proche de 0 indique des comportements très similaires tandis que des valeurs éloignées de 0 correspondent à des divergences marquées. Partant de cela nous faisons une hypothèse : si les comportements d'utilisateurs dans 2 ensembles sont clairement distincts alors il doit y avoir une divergence notable entre leur distribution d'actes utilisateurs. La divergence KL est calculée en utilisant les fréquences de sélection d'actions utilisateurs entre tous les ensembles. Comme on peut le noter au travers des résultats du tableau 2 les ensembles regroupent bien des comportements d'utilisateurs distincts. Ceci tend à montrer que chaque ensemble représente vraiment un comportement utilisateur spécifique, bien marqué par rapport aux autres ensembles.

Un point important à établir est la cohésion au sein des éléments présents dans un même ensemble, c'est à dire montrer qu'il y a une bonne cohésion intra-ensemble. Pour mesure la cohésion intra-ensemble nous utilisons la mesure de similarité cosinus entre le centroide de l'ensemble et les éléments de cet ensemble. La valeur de la similarité cosinus entre 2 vecteurs d'observations peut aller de -1 (signalant des éléments aux comportements tout a fait opposés), en passant par 0 (signalant des éléments indépendants) jusqu'à 1 (signalant des éléments au comportement identique). La mesure cosinus est calculée pour chaque ensemble.

TABLE 2 – *Divergences de Kullback-Leibler inter-ensembles*

Paire testée	Divergence KL
Cluster-1 vs Cluster-2	2.52
Cluster-1 vs Cluster-3	2.51
Cluster-1 vs Cluster-4	1.99

TABLE 3 – *Mesure de similarité cosinus intra-ensemble*

Ensemble	Similarité cosinus
Cluster-1	0.68
Cluster-2	0.60
Cluster-3	0.55
Cluster-4	0.70

Sans regroupement la similarité cosinus du jeu de données complet est de 0,4. Le tableau 3 donne les mesures de similarité pour les 4 ensembles et on peut observer que les valeurs sont toutes proches de 1. Ceci confirme une forte cohésion intra-ensemble et donc une similarité importante entre les comportements des utilisateurs présents dans chaque ensemble.

5 Conclusion

Identifier et/ou simuler les différents comportements utilisateurs possibles face à une tâche de dialogue est un problème important pour apprendre et évaluer des stratégies de dialogue adaptatives. Pourtant jusqu'à présent les moyens de représenter de manière compacte et efficace les comportements utilisateurs au travers des trajectoires de dialogues observées ne sont pas utilisés dans le domaine de la gestion du dialogue. Représenter le comportement utilisateur sous la forme d'un vecteur d'observations pondérées de trajectoire permet d'explorer de nouvelles opportunités, comme le regroupement automatique de comportements utilisateurs. Les résultats expérimentaux dans le cas du problème de dialogue à 3 critères a permis de valider l'approche en permettant de retrouver automatiquement des comportements définis manuellement. Le passage au un problème de grande échelle (12 critères) a permis de monter que l'approche résistait à l'effet d'échelle et permettait de regrouper des comportements humains dans des ensembles bien distincts et cohérents.

En ce qui concerne le travail futur, une fois que les comportements d'utilisateurs sont regroupés il est envisageable de construire des simulateurs d'utilisateurs permettant de reproduire différents types d'utilisateurs. Il sera alors possible d'utiliser ces simulateurs aux comportements paramétrables pour l'optimisation et l'évaluation des stratégies de dialogue (adaptives). Une autre voie à poursuivre en relation avec ce point est la qualification des comportements utilisateurs. Dotés de mesures automatiques il devient pertinent de cherche à qualifier plus finement les comportements utilisateurs discriminés. L'intérêt serait alors, ayant compris les vrais traits distinctifs des comportements, de pouvoir générer les données manquantes dans les corpus de données. Enfin, dans ce papier, nous avons utilisé l'algorithme des K-moyennes qui est sensible au fait que certains ensembles soient de petites tailles. D'autres algorithmes de quantification doivent être explorés, moins sensibles aux distributions non uniformes des données dans les ensembles (comme l'algorithme du gaz neuronal, par exemple).

Références

ABBEEL P. & NG A. (2004). Apprenticeship learning via inverse reinforcement learning. In *Proc. of ICML.*

BELLMAN R. (1957). A markovian decision process. *Journal of Mathematics and Mechanics, vol. 6, pp. 679–684.*

CHANDRAMOHAN S., GEIST M., LEFÈVRE F. & PIETQUIN O. (2011). User Simulation in Dialogue Systems using Inverse Reinforcement Learning. In *Proc. of Interspeech 2011*, Florence (Italy).

ECKERT W., LEVIN E. & PIERACCINI R. (1997). User Modeling for Spoken Dialogue System Evaluation. In *Proc. of ASRU*, p. 80–87.

GEORGILA K., HENDERSON J. & LEMON O. (2005). Learning User Simulations for Information State Update Dialogue Systems. In *Eurospeech*.

KEIZER S., GASIC M., JURCICEK F., MAIRESSE F., THOMSON B., YU K. & YOUNG S. (2010). Parameter estimation for agenda-based user simulation. In *Proc. of SigDial 2010*, p. 116–123.

KULLBACK S. & LEIBLER R. A. (1951). On information and sufficiency. *Ann. Math. Statist.*, **22**(1), 79–86.

LARSSON S. & TRAUM D. (2000). Information state and dialogue management in the TRINDI dialogue move engine toolkit. *Natural Language Engineering, vol. 6, pp 323–340*.

LEMON O., GEORGILA K., HENDERSON J. & STUTTLE M. (2006). 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.

LEMON O. & PIETQUIN O. (2007). Machine learning for spoken dialogue systems. In *Proc. of InterSpeech'07*, Belgium.

LEVIN E. & PIERACCINI R. (1998). Using markov decision process for learning dialogue strategies. In *Proc. ICASSP'98, Seattle (USA)*.

PIETQUIN O. (2006). Consistent goal-directed user model for realistic man-machine task-oriented spoken dialogue simulation. In *Proc. of ICME'06*, p. 425–428, Toronto (Canada).

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

PIETQUIN O., GEIST M., CHANDRAMOHAN S. & FREZZA-BUET H. (2011). Sample-Efficient Batch Reinforcement Learning for Dialogue Management Optimization. *ACM Transactions on Speech and Language Processing*, **7**(3), 7 :1–7 :21.

RIESER V. & LEMON O. (2006). Cluster-based User Simulations for Learning Dialogue Strategies. In *Proc. of Interspeech/ICSLP 2006*.

SCHATZMANN J., STUTTLE M. N., WEILHAMMER K. & YOUNG S. (2005). Effects of the user model on simulation-based learning of dialogue strategies. In *Proc. of ASRU'05, Puerto Rico*.

SCHATZMANN J., THOMSON B., WEILHAMMER K., YE H. & YOUNG. S. (2007). Agenda-based User Simulation for Bootstrapping a POMDP Dialogue System. In *Proc. of HLT/NAACL*.

SCHATZMANN J., WEILHAMMER K., STUTTLE M. & YOUNG S. (2006). A survey of statistical user simulation techniques for reinforcement-learning of dialogue management strategies. *Knowledge Engineering Review, vol. 21(2), pp. 97–126*.

SUTTON R. & BARTO A. (1998). *Reinforcement Learning : An Introduction*. The MIT Press, 3rd edition.