UNIVERSITÉ DE LORRAINE

École doctorale IAEM Lorraine

Apprentissage automatique des paramètres de l'apprentissage par renforcement pour les systèmes de dialogue adaptatifs

THÈSE

présentée et soutenue publiquement le 21 janvier 2016

pour l'obtention du

Doctorat de l'Université de Lorraine

(mention informatique)

par

Layla El Asri

Composition du jury

Rapporteurs :	Wolfgang Minker Fredéric Béchet
Examinateurs :	Christophe Cerisara Fabrice Lefèvre

Laboratoire Lorrain de Recherche en Informatique et ses Applications — UMR 7503
UMI 2958 (CNRS-GeorgiaTech) - Orange Labs

Résumé

Cette thèse s'inscrit dans le cadre de la recherche sur les systèmes de dialogue. Ce document propose d'apprendre le comportement d'un système à partir d'un ensemble de dialogues annotés. Le système apprend un comportement optimal via l'apprentissage par renforcement. Nous montrons qu'il n'est pas nécessaire de définir une représentation de l'espace d'état ni une fonction de récompense. En effet, ces deux paramètres peuvent être appris à partir du corpus de dialogues annotés. Nous montrons qu'il est possible pour un développeur de systèmes de dialogue d'optimiser la gestion du dialogue en définissant seulement la logique du dialogue ainsi qu'un critère à maximiser (par exemple, la satisfaction utilisateur).

La première étape de la méthodologie que nous proposons consiste à prendre en compte un certain nombre de paramètres de dialogue afin de construire une représentation de l'espace d'état permettant d'optimiser le critère spécifié par le développeur. Par exemple, si le critère choisi est la satisfaction utilisateur, il est alors important d'inclure dans la représentation des paramètres tels que la durée du dialogue et le score de confiance de la reconnaissance vocale. L'espace d'état est modélisé par une mémoire sparse distribuée. Notre modèle, *Genetic Sparse Distributed Memory for Reinforcement Learning* (GSDMRL), permet de prendre en compte de nombreux paramètres de dialogue et de sélectionner ceux qui sont importants pour l'apprentissage par évolution génétique. L'espace d'état résultant ainsi que le comportement appris par le système sont aisément interprétables.

Dans un second temps, les dialogues annotés servent à apprendre une fonction de récompense qui apprend au système à optimiser le critère donné par le développeur. A cet effet, nous proposons deux algorithmes, *reward shaping* et *distance minimisation*. Ces deux méthodes interprètent le critère à optimiser comme étant la récompense globale pour chaque dialogue. Nous comparons ces deux fonctions sur un ensemble de dialogues simulés et nous montrons que l'apprentissage est plus rapide avec ces fonctions qu'en utilisant directement le critère comme récompense finale.

Nous avons développé un système de dialogue dédié à la prise de rendez-vous et nous avons collecté un corpus de dialogues annotés avec ce système. Ce corpus permet d'illustrer la capacité de mise à l'échelle de la représentation de l'espace d'état GSDMRL et constitue un bon exemple de système industriel sur lequel la méthodologie que nous proposons pourrait être appliquée.

Chapter 1

Introduction

1.1 Contexte

Les Systèmes de Dialogue (SD) sont des interfaces homme-machine permettant à l'utilisateur de s'exprimer en langue naturelle. Les SD ont connu un essor considérable ces dernières années grâce aux avancées dans le domaine de la reconnaissance vocale et plus globalement, du TALN (Traitement Automatique de la Langue Naturelle) [18]. Leurs applications commerciales sont nombreuses et répandues sur les principales plateformes mobiles. Un exemple fameux est l'assistant d'Apple, Siri, qui permet de chercher un restaurant, envoyer un email ou encore régler une alarme. Les applications académiques sont aussi plurielles, allant de la recherche d'information sur les trajets de bus [31] à la prise de rendez-vous [22] en passant par la recherche d'activités dans une ville [24].

Toutefois, et spécialement dans le secteur industriel, les SD déployés ne disposent que de capacités de dialogue limitées. En effet, ces applications sont souvent construites afin de mener des dialogues courts permettant d'effectuer des tâches simples. Par exemple, lorsqu'un utilisateur demande à Siri de rechercher un type de restaurant à proximité, l'assistant se contente de lister tous les établissements trouvés. L'introduction de plus amples outils de dialogue permettrait à l'assistant de réduire le nombre de restaurants trouvés en demandant, par exemple, le budget de l'utilisateur. La possibilité d'un dialogue plus étendu avec l'utilisateur pose plusieurs difficultés. Un premier obstacle est la gestion des erreurs de reconnaissance vocale. En effet, les systèmes courants affichent toujours un taux d'erreurs proche de 30%. Ainsi, il faut équiper le système de stratégies de récupération en cas d'erreur de reconnaissance. Une deuxième difficulté concerne le dialogue en lui-même. Effectivement, il est nécessaire de définir des stratégies concernant l'ordre des questions, la présentation de l'information, etc.

Une réponse efficace à ces problèmes consiste à modéliser la gestion du dialogue via un Processus de Décision Markovien (PDM) et d'utiliser les techniques dites d'Apprentissage par Renforcement (AR). Ce modèle implique que le système de dialogue est pourvu de différentes stratégies de dialogue qu'il apprend ensuite à utiliser en fonction d'un critère à optimiser, par exemple, la satisfaction utilisateur. Techniquement, le développeur de SD définit un ensemble d'**actions** telles que *demander à l'utilisateur une certaine information*, *demander à l'utilisateur de confirmer ce que le système a compris* ou encore *jouer un message d'aide*. A l'image de ce qu'un humain fait durant une conver-

sation, le système doit conserver une représentation de l'état courant du dialogue afin de savoir quoi dire à chaque étape du dialogue. Dans le vocabulaire de l'AR, ce type de représentation est appelé un **état**. En tout, le développeur doit définir un ensemble d'actions, d'états et un critère à maximiser. Ensuite, le système apprend quelle action choisir en fonction de son état courant. Cet apprentissage s'effectue au moyen d'une **fonction de récompense**, qui indique au système la qualité de chaque choix d'action. Cette fonction est liée au critère à maximiser. Elle est censée être la représentation la plus robuste et la plus compacte de la tâche du système. Par exemple, on peut distribuer une récompense égale à la satisfaction utilisateur à la fin de chaque dialogue. Ainsi, le système apprendra quelle succession d'actions maximise la satisfaction utilisateur. Le SD calcule un **retour** pour chaque couple état-action. Ce retour correspond à la somme réduite des récompenses futures. Les actions sont comparées et choisies sur la base de ce calcul. L'association état-action qui résulte de ce choix est appelée **politique** du système. Le but de l'apprentissage est de déterminer une politique optimale, c'est-à-dire, une association permettant de maximiser l'espérance de retour à chaque état.

Le modèle PDM a de nombreux avantages et répond à toutes les difficultés précédemment évoquées. En effet, la récupération des erreurs ainsi que la stratégie de dialogue peuvent être apprises automatiquement, réduisant ainsi considérablement le temps de développement. L'équipe NADIA (NAtural DIAlogue) d'Orange a été la première a déployer un système commercial intégrant l'AR en 2010 [21]. Depuis, à notre connaissance, il n'y a pas eu de nouvelles exploitations commerciales de l'AR pour les SD. Le principal obstacle à une utilisation plus étendue de l'AR est la difficulté technique de cette méthode. En effet, de nombreux développeurs de solutions vocales ne sont pas experts en *machine learning* et *a fortiori* en AR. Les techniques d'AR ne sont d'ailleurs pas toujours adaptées à un usage industriel et à des contraintes de qualité et de contrôle renforcées. Il manque un modèle d'AR efficace à mettre en place et permettant de respecter les contraintes industrielles. Il s'agit là de l'objectif de cette thèse.

1.2 Motivations

Concernant la facilitation de mise en place d'AR dans les SD, deux tendances se distinguent dans la littérature : la première cherche des méthodes permettant de transférer l'apprentissage d'un système à un autre [41, 3, 15] tandis que la seconde propose des techniques permettant d'apprendre les paramètres de l'AR, c'est-à-dire la fonction de récompense ainsi que la représentation de l'espace d'état [38, 28, 4, 37, 39]. Nous nous intéressons ici à cette deuxième tendance.

Il est difficile de concevoir manuellement la représentation de l'espace d'état. Les états du système représentent le contexte du dialogue. Or, le contexte est déterminé par de nombreux facteurs. Dans un premier temps, il est nécessaire de savoir où l'on se trouve dans la logique du dialogue (par exemple, l'utilisateur a indiqué qu'il souhaitait envoyer un email). A cela, il faut ajouter les paramètres qui permettent de savoir si le dialogue se déroule de manière satisfaisante conformément à la fonction de récompense. Il est en effet nécessaire que l'espace d'état soit adapté à l'apprentissage de la fonction de récompense. Par ailleurs, la représentation de l'espace d'état doit prendre en compte un nombre important de paramètres (durée du dialogue, nombre de fois où l'utilisateur a demandé de l'aide, etc.) et de valeurs possibles de ces paramètres. Afin de remplir ces critères, il a été proposé d'apprendre cette représentation à partir de données [28, 38, 32, 15]. Toutefois, ces propositions soit

ne permettent pas de prendre en compte de nombreux paramètres, soit ne renvoient pas un résultat facilement interprétable. Or l'interprétabiité est très importante pour le développeur de système de dialogue industriel qui doit garder un moyen de suivre l'apprentissage de son système.

De même, il est difficile et crucial de définir une fonction de récompense adaptée. Comme dit précédemment, la fonction de récompense représente la tâche du système. Or il n'est pas évident de choisir les paramètres à prendre en compte dans le calcul des récompenses. Par exemple, on pourrait distribuer une récompense positive dans le cas où le système a permis à l'utilisateur d'accomplir une tâche. On pourrait aussi distribuer des récompenses négatives dans le cas où le système de reconnaissance vocale n'aurait pas compris l'énoncé de l'utilisateur. La distribution des récompenses requiert donc d'effectuer des choix entre les paramètres à prendre en compte ainsi que ceux à privilégier. Toutefois, il n'est pas facile de transcrire des préférences en récompenses de sorte que le système agisse de la manière souhaitée par le développeur. Pour contrer cette difficulté, il est là aussi possible d'apprendre la fonction de récompense à partir de données [40, 4, 37]. Nous proposons ici deux algorithmes permettant d'apprendre une fonction de récompense et nous montrons que ces algorithmes peuvent être utilisés pour l'apprentissage en-ligne et qu'ils permettent d'apprendre plus vite qu'avec une récompense distribuée seulement à la fin de chaque dialogue.

1.3 Contributions

Nous montrons dans cette thèse qu'il est possible d'apprendre une politique pour un système de dialogue à partir de dialogues annotés, sans passer par la définition d'une représentation de l'espace d'états ni d'une fonction de récompense. Pour ce faire, nous proposons la méthodologie suivante :

- Nous demandons au développeur de système de dialogue de concevoir uniquement la partie dialogique du système, c'est-à-dire de définir les états du dialogue (par exemple, l'utilisateur souhaite envoyer un email) ainsi que les actions disponibles (par exemple, demander à l'utilisateur de confirmer ou d'indiquer l'objet de l'email).

- Il est ensuite nécessaire de collecter des données avec ce système. Il est conseillé d'utiliser une politique purement exploratoire pendant la collecte de données. Cela signifie que le système tire au hasard de manière uniforme chaque action à entreprendre.

- Par la suite, nous demandons à ce que chaque dialogue reçoive un score. Ce score peut être délivré par un expert ou directement par l'utilisateur ayant effectué le dialogue.

- L'étape suivante consiste à collecter des indicateurs clés de performance à partir des logs. Ces indicateurs sont la durée du dialogue, le score de confiance moyen de la reconnaissance vocale, etc. Ils permettent d'estimer la qualité du dialogue.

- Le corpus ainsi constitué permet alors d'apprendre une politique pour le système sans passer par la définition d'une fonction de récompense ni d'un espace d'état. Ces paramètres sont appris, d'abord la représentation de l'espace d'état puis la fonction de récompense.

Cette méthodologie est illustrée sur la figure 1.1.

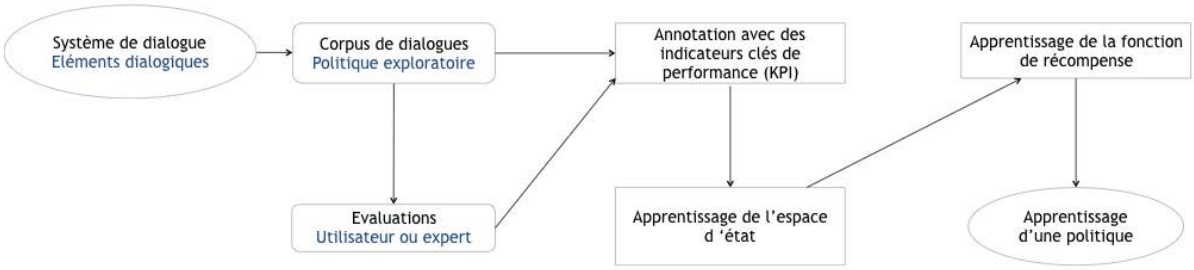

Figure 1.1: Méthodologie pour apprendre une politique automatiquement à partir de données.

Nous avons appliqué cette méthodologie à des systèmes de dialogue dédiés à la prise de rendez-vous. L'un des systèmes, NASTIA, a fait l'objet d'une expérience utilisateur permettant de collecter 1734 dialogues avec un score donné par l'utilisateur.

1.4 Plan de la présentation

Dans un premier temps, nous présenterons l'apprentissage par renforcement pour les systèmes de dialogue puis nous décrirons les diverses méthodes proposées pour l'évaluation des systèmes de dialogue.

Le second chapitre présentera notre solution pour la définition de l'espace d'état. Cette solution permet de prendre en compte de nombreux paramètres tout en apprenant efficacement.

Le troisième chapitre présentera deux algorithmes apprenant une fonction de récompense à partir de dialogues annotés.

Enfin, nous proposerons des perspectives de travaux futurs.

Chapter 2

Conception et Evaluation des systèmes de dialogue

2.1 Systèmes de dialogue

Les systèmes de dialogue sont conçus selon une architecture générique. En entrée, l'énoncé utilisateur est transmis au module de reconnaissance vocale (ASR). Ce module formule des hypothèses sur cet énoncé. Par exemple, si l'utilisateur dit *je veux prendre rendez-vous mardi*, l'ASR pourrait formuler plusieurs hypothèses parmi lesquelles *je veux prendre rendez-vous mardi*, *je veux prendre rendez-vous lundi*, *je veux prendre rendez-vous à midi*, etc. Chaque hypothèse est accompagnée d'un score de confiance. Les hypothèses sont transmises au module de compréhension du langage naturel (NLU) qui les transforme en actes de dialogue. Un acte de dialogue est une représentation simplifiée de l'hypothèse, par exemple PRENDRE_RENDEZ_VOUS. L'acte est accompagné des paramètres de la requête, dans ce cas mardi. L'acte de dialogue est transmis au module de calcul d'état qui va former l'état courant du dialogue, c'est-à-dire le contexte ainsi que certains paramètres telle que la durée du dialogue, etc. L'état est ensuite transféré au module de gestion du dialogue (DM) qui choisit la prochaine action du système et transmet cette action sous la forme d'un acte de dialogue au module de génération de langage naturel (NLG). Ce module formule un énoncé à partir de l'acte de dialogue et l'énoncé est prononcé oralement par le module de synthèse vocale.

La modélisation sous forme de Processus de Décision Markovien (PDM) ainsi que l'apprentissage par renforcement ont été utilisés pour la gestion du dialogue dès la fin des années 90 [23, 36]. Les premières décisions apprises concernaient le choix entre questions ouvertes et questions fermées ainsi que le moment où demander confirmation de ce que l'ASR et le NLU ont compris. Ces méthodes ont depuis été utilisées pour prendre des décisions variées, allant de la manière optimale de présenter une information [40, 32] au type de prosodie à utiliser citepBretier:10. Des applications récentes s'inscrivent dans le cadre de la négociation [17, 16, 1] où système et utilisateur ont des buts différents. Nous présentons le formalisme des PDM et de l'apprentissage par renforcement dans la section suivante.

2.2 Apprentissage par renforcement pour les systèmes de dialogues

2.2.1 Processus de décision Markoviens

Formalisme

Un PDM est un quintuplet (S, A, P, R, γ) où S est l'espace d'état, A est l'espace d'actions, P sont les probabilités de transition, R est la fonction de récompense et γ est un facteur de réduction, dans $[0, 1[$.

Dans le cas du dialogue, S est l'ensemble des contextes possibles du dialogue, A est l'ensemble des actions disponibles au DM et R est une représentation numérique de la tâche du système (par exemple, la satisfaction utilisateur).

Propriété de Markov

Les PDM reposent sur la *propriété de Markov* : la probabilité d'aller à l'état s_{t+1} au temps $t + 1$ dépend uniquement de l'état précédent s_t et de l'action précédente a_t:

$$P(s_{t+1} \mid s_t, a_t, s_{t-1}, a_{t-1}, ..., s_0, a_0) = P(s_{t+1} \mid s_t, a_t).$$

Il est nécessaire d'inclure dans l'espace d'état un historique du dialogue afin de respecter cette règle [29].

Résolution

Les PDM cherchent à accomplir une tâche représentée par la fonction de récompense. Le but est de maximiser le retour :

$$r_t = \sum_{t' \geq 0} \gamma^{t'} R_{t+t'+1}. \tag{2.1}$$

Le facteur γ permet de donner plus ou moins d'importance aux récompenses reçues après le temps t. Ce modèle est adapté aux systèmes de dialogue où il faut attendre la fin du dialogue pour savoir si celui-ci fut un succès et où chaque décision joue un rôle important dans le succès du dialogue.

Formellement, on apprend une politique π, c'est-à-dire une correspondance entre états et actions :

$$\begin{aligned} \pi : \quad & S \to A \\ & s \mapsto \pi(s) = a. \end{aligned}$$

Le but est d'apprendre une politique optimale. Une politique optimale maximise l'espérance de retour pour chaque état s. Afin de trouver une politique optimale, nous estimons la fonction de valeur Q, ou Q-fonction :

$$Q^\pi(s, a) = E[r_t \mid s_t = s, a_t = a, \pi]. \tag{2.2}$$

2.3 Evaluation des systèmes de dialogue

Les systèmes de dialogue font généralement l'objet de deux types d'évaluation : une évaluation individuelle des composants et une évaluation de l'utilisabilité du système [6, 27]. Certains composants peuvent être évalués individuellement, il s'agit par exemple de l'ASR pour lequel il est possible de calculer simplement le taux d'erreurs, ou encore le NLG qui donne lieu à une évaluation subjective. Toutefois, la gestion du dialogue ne peut pas être évaluée individuellement. En effet, elle est liée à d'autres composants. Par exemple, il est courant d'inclure dans la gestion du dialogue des stratégies de récupération des erreurs commises par l'ASR. Ainsi, il est préférable d'estimer l'utilisabilité du système, qui traduit la capacité du système à satisfaire l'utilisateur.

Il est possible d'évaluer l'utilisabilité d'un système en demandant directement aux utilisateurs de remplir un questionnaire et de donner un score d'utilisabilité après chaque dialogue. Il est aussi possible de faire appel à des experts qui écouteront les dialogues *a posteriori* et leur attribueront une note.

Pour NASTIA [10], nous avons choisi de demander aux utilisateurs de remplir un questionnaire et de donner un score entre 1 et 10 après chaque dialogue. Le corpus DINASTI résultant de cette étude [11] contient 1734 dialogues et nous a permis d'appliquer en partie la méthodologie à des données réelles. Les logs des dialogues du corpus DINASTI nous ont permis de collecter 120 indicateurs de performance (durée du dialogue, score de confiance moyen de la reconnaissance vocale, nombre de rejets de la reconnaissance vocale,...). La description du corpus a été publiée : [11].

Après la constitution du corpus, la première étape de la méthodologie consiste à apprendre une représentation de l'espace d'état. Nous décrivons dans la partie suivante un algorithme permettant de le faire.

Chapter 3

Modélisation de l'espace d'état

3.1 Problème

Comme dit précédemment, l'état du système représente le contexte du dialogue. Deux types de paramètres sont à prendre en compte dans le contexte, d'une part le contexte dialogique (par exemple, l'utilisateur a refusé une proposition de prise de rendez-vous) et d'autre part, les paramètres indiquant si le dialogue se passe bien (durée du dialogue, nombre de fois où l'utilisateur a réclamé de l'aide, etc.). Dans la méthodologie présentée précédemment, la maximisation de la satisfaction utilisateur (ou qualité d'interaction si l'on demande à un expert de noter le dialogue plutôt qu'un utilisateur) est choisie comme tâche. Or, la satisfaction utilisateur dépend de très nombreux paramètres (complétion de tâche, durée du dialogue, etc.) [42]. Par conséquent, pour apprendre une politique maximisant la satisfaction utilisateur, il est nécessaire d'inclure dans l'espace d'état les paramètres qui influent sur cette donnée. Dans le corpus DINASTI, nous avons identifié 120 paramètres pouvant jouer un rôle dans la satisfaction utilisateur. Le fait que le nombre de paramètres soit grand et que ces paramètres peuvent prendre de très nombreuses valeurs (par exemple, la durée du dialogue est un paramètre continu) implique qu'il n'est pas possible de représenter la Q-fonction de manière tabulaire. Dans ce cas, il est nécessaire de faire appel à une paramétrisation.

Nous avons fait le choix de la paramétrisation linéaire suivante :

$$\hat{Q}_\Theta(s,a) = \sum_i^p \theta_i \phi_i(s,a). \tag{3.1}$$

L'espace d'état est ainsi réduit à p fonctions de base ϕ_i et l'apprentissage de la Q-fonction revient à l'apprentissage des poids θ_i. Le problème consiste donc à résumer l'espace d'état en p fonctions de base.

Dans la littérature, trois familles de solutions ont été proposées. La première consiste à adopter une représentation dense, dans le sens où l'espace entier est découpé en régions. Les états sont donc agrégés en fonction de leur proximité [5, 35, 20, 14, 30, 25]. Ce type présente l'inconvénient qu'en discrétisant tout l'espace, le nombre de régions augmente fortement avec le nombre de dimensions. Par conséquent, l'apprentissage par renforcement devient inefficace. Une autre solution propose de se focaliser sur la trajectoire optimale. Les états sont agrégés en fonction de leur action optimale [2, 43].

Toutefois, il a été montré que ce type d'approche ne permet pas de calculer une politique optimale si l'espace d'état n'est pas entièrement observable McCallum [26]. Dans ce cas, il est préférable de choisir le dernier type d'approche, à savoir, l'agrégation d'états en fonction de la fonction de valeur. Nous proposons un algorithme suivant cette approche dans la section suivante.

3.2 Mémoire Parcimonieuse Distribuée Génétique pour l'Apprentissage par Renforcement

Afin de réduire l'espace d'état à un petit nombre de paramètres, nous avons choisi une approche *instance-based* [14, 30], c'est-à-dire que nous maintenons en mémoire un petit nombre d'états, dit prototypes. La deuxième difficulté vient du fait que les paramètres à prendre en compte sont disparates, ils peuvent être en effet discrets, continus ou catégoriques. Afin de répondre à cette difficulté, nous avons choisi d'utiliser une mémoire parcimonieuse distribuée (SDM) [19]. Les SDM représentent les données sous forme binaire. Ainsi, il est uniquement nécessaire de choisir une représentation binaire des données, puis on peut inclure tous les paramètres de la même manière dans le modèle. Enfin, les états prototypes doivent permettre de calculer une politique optimale. Pour ce faire, nous optons pour la famille de solutions proposant d'agréger les états en fonction de leur Q-valeurs. Nous nous sommes inspirés des travaux de Rogers [33, 34] et avons ajouté une composante génétique à la SDM afin de ré-agencer les prototypes en fonction de leurs valeurs. L'algorithme résultant, GSDMRL a été publié [12]. Les innovations de GSDMRL consistent en la construction de la base d'états prototypes, l'architecture permettant de ré-agencer les états et enfin, une adaptation au problème de régression de la fonction Q. La paramétrisation linéaire ainsi que l'approche *instance-based* permettent d'interpréter aisément la politique apprise.

Nous avons montré que l'apprentissage avec GSDMRL était très efficace dans le sens où le nombre d'états prototypes créés était faible par rapport au nombre d'états nécessaire pour une représentation dense de l'espace (jusqu'à 4 fois moins) alors que la politique apprise était de qualité similaire. Nous avons testé GSDMRL sur une tâche classique de l'apprentissage par renforcement et dans le domaine du dialogue, sur un simulateur de prise de rendez-vous.

Enfin nous avons appliqué GSDMRL au corpus DINASTI et nous avons montré qu'il était possible de réduire l'espace d'état (à 120 paramètres) à seulement 899 prototypes tout en apprenant une politique dont l'interprétation donne des indications précieuses sur le système.

Le dernière étape de la méthodologie consiste à apprendre une fonction de récompense à partir des dialogues. Nous proposons deux algorithmes permettant d'effectuer cela.

Chapter 4

Apprentissage d'une fonction de récompense

4.1 Problème

Après avoir appris une représentation efficace de l'espace d'état, il est nécessaire d'apprendre une fonction de récompense. Cette fonction représente la tâche du système. Ici, nous basons les récompenses sur les scores donnés aux dialogues par les utilisateurs ou les experts. Nous avons proposé deux algorithmes permettant de récompenser le système de sorte à maximiser la satisfaction utilisateur : les algorithmes de *reward shaping* et de minimisation de la distance. Ces deux algorithmes interprètent la satisfaction utilisateur comme récompense totale pour le dialogue, c'est-à-dire, comme le retour au temps 0 r_{t_0}. Dans la littérature, le score est majoritairement interprété comme étant la valeur de l'état final. Nous avons montré sur différentes tâches (la prise de rendez-vous et la recherche de restaurant) que notre interprétation permettait d'apprendre plus vite. Les deux algorithmes ont été publiés [7, 8].

4.2 Approche

L'algorithme de *reward shaping* consiste à propager le score de satisfaction utilisateur à travers le PDM. Cette propagation permet d'apprendre plus vite qu'en donnant le score uniquement à la fin du dialogue. Cet algorithme nécessite un estimateur des scores de satisfaction utilisateur afin que l'apprentissage du système puisse être poursuivi en-ligne. Nous avons proposé d'utiliser une régression ordinale pour ce faire. Sur un corpus de 200 dialogues dans le domaine de la recherche de bus, nous avons montré que la régression ordinale donnait de meilleurs résultats que les méthodes de régression et de classification usuelles sur de nombreuses métriques (erreur euclidienne, coefficient de Spearman,...). Cette étude a fait l'objet d'une publication [9]. Une fois l'estimateur des scores appris, il est possible de propager ce score à travers le PDM. Nous avons proposé un algorithme original pour ce faire [7].

L'algorithme de minimisation de la distance repose sur la même interprétation du score de satisfaction utilisateur mais effectue une régression linéaire afin de calculer les récompenses intermédiaires. Cet algorithme ne nécessite pas d'estimateur des scores, ceux-ci sont directement dispersés à travers le PDM de sorte que le retour au temps t_0 corresponde au score du dialogue. Cet algorithme a aussi fait l'objet d'une étude théorique [13].

Chapter 5

Conclusion et travaux futurs

Lors de cette thèse, nous avons, dans le contexte des systèmes de dialogue, proposé une méthodologie complète pour apprendre une politique de gestion du dialogue sans passer par la définition d'un espace d'état ni d'une fonction de récompense.

Nous avons mis au point un algorithme d'apprentissage de l'espace d'état permettant de prendre en compte de nombreux paramètres et d'interpréter aisément la politique apprise. Nous avons aussi proposé deux algorithmes permettant de calculer une fonction de récompense permettant d'accélérer l'apprentissage en propageant les scores de satisfaction utilisateur.

Les perspectives de travaux futurs concernent dans un premier temps l'apprentissage actif : il serait intéressant de pourvoir le système de la capacité de demander qu'un dialogue soit annoté si ce dialogue correspond à une situation inconnue ou peu connue. Cela permettrait d'améliorer la représentation de l'espace d'état et la fonction de récompense. Une seconde perspective se situe dans le domaine du dialogue incrémental. Le dialogue incrémental est un modèle d'interaction très fluide, où l'utilisateur ou le système peut intervenir à tout moment. Ce type de dialogue implique une architecture particulière de la gestion du dialogue et une perspective prometteuse serait d'adapter nos travaux à ce type de système.

Bibliography

[1] Merwan Barlier, Julien Perolat, Romain Laroche, and Olivier Pietquin. Human-machine dialogue as a stochastic game. In *Proc. of SIGDIAL*, 2015.

[2] Andrey Bernstein and Nahum Shimkin. Adaptive aggregation for reinforcement learning with efficient exploration: Deterministic domains. In *Proc. of COLT*, 2008.

[3] Dan Bohus and Alexander I. Rudnicky. Error handling in the RavenClaw dialog management architecture. In *Proc. of HLT-EMNLP*, 2005.

[4] Abdeslam Boularias, Hamid R. Chinaei, and Brahim Chaib-draa. Learning the reward model of dialogue pomdps from data. In *Proc. of NIPS*, 2010.

[5] D.S. Broomhead and David Lowe. Multivariable functional interpolation and adaptive networks. *Complex Systems*, 1988.

[6] Laila Dybkjaer, Niels O. Bernsen, and Wolfgang Minker. Evaluation and usability of multimodal spoken language dialogue systems. *Speech Communication*, 43:33–54, 2004.

[7] Layla El Asri, Romain Laroche, and Olivier Pietquin. Reward function learning for dialogue management. In *Proc. of STAIRS*, 2012.

[8] Layla El Asri, Romain Laroche, and Olivier Pietquin. Reward shaping for statistical optimisation of dialogue management. In *Proc. of SLSP*, 2013.

[9] Layla El Asri, Hatim Khouzaimi, Romain Laroche, and Olivier Pietquin. Ordinal Regression for Interaction Quality Prediction. In *Proc. of ICASSP*, 2014.

[10] Layla El Asri, Romain Laroche, and Olivier Pietquin. DINASTI: Dialogues with a Negotiating Appointment Setting Interface. In *Proc. of LREC*, 2014.

[11] Layla El Asri, Rémi Lemonnier, Romain Laroche, Olivier Pietquin, and Hatim Khouzaimi. NASTIA: Negotiating Appointment Setting Interface. In *Proc. of LREC*, 2014.

[12] Layla El Asri, Romain Laroche, and Olivier Pietquin. Compact and interpretable dialogue state representation with genetic sparse distributed memory. In *Accepted to IWSDS*, 2016.

[13] Layla El Asri, Bilal Piot, Mathieu Geist, Romain Laroche, and Olivier Pietquin. Score-based inverse reinforcement learning. In *Submitted to AAMAS*, 2016.

[14] Jeffrey R.N. Forbes. *Reinforcement Learning for Autonomous Vehicles*. PhD thesis, University of California at Berkeley, 2002.

[15] Milica Gašić, Catherine Breslin, Matthew Henderson, Dongho Kim, Martin Szummer, Blaise Thomson, Pirros Tsiakoulis, and Steve Young. POMDP-based dialogue manager adaptation to extended domains. In *Proc. of SIGDIAL*, 2013.

[16] Kallirroi Georgila and David Traum. Reinforcement learning of argumentation dialogue policies in negotiation. In *Proc. of InterSpeech*, 2011.

[17] P.A. Heeman. Representing the reinforcement learning state in a negotiation dialogue. In *Proc. of ASRU*, 2009.

[18] Geoffrey E. Hinton, Li Deng, Dong Yu, George E. Dahl, Abdel rahman Mohamed, Navdeep Jaitly, Andrew Senior, Vincent Vanhoucke, Patrick Nguyen, Tara N. Sainath, and Brian Kingsbury. Deep Neural Networks for Acoustic Modeling in Speech Recognition: The Shared Views of Four Research Groups. *IEEE Signal Process. Mag.*, 2012.

[19] Pentti Kanerva. *Sparse Distributed Memory*. Cambridge, Mass.: Bradford/MIT Press., 1988.

[20] Kostas Kostiadis and Huosheng Hu. KaBaGe-RL: Kanerva Based Generalisation and Reinforcement Learning for Possession Football. In *Proc. of IEEE IROS*, 2001.

[21] Romain Laroche, Ghislain Putois, and Philippe Bretier. Optimising a handcrafted dialogue system design. In *Proc. of InterSpeech*, 2010.

[22] Romain Laroche, Ghislain Putois, Philippe Bretier, Martin Aranguren, Julia Velkovska, Helen Hastie, Simon Keizer, Kai Yu, Filip Jurčíček, Oliver Lemon, and Steve Young. Report D6.4 : Final evaluation of classic towninfo and appointment scheduling systems. Technical report, CLASSIC Project, 2011.

[23] Esther Levin, Roberto Pieraccini, and Wieland Eckert. Learning dialogue strategies within the markov decision process framework. In *Proc. of IEEE ASRU*, 1997.

[24] Diane Litman, Satinder Singh, Michael Kearns, and Marilyn Walker. Njfun: A reinforcement learning spoken dialogue system. In *In Proc. of the ANLP/NAACL Workshop on Conversational Systems*, 2000.

[25] Sridhar Mahadevan, Mauro Maggioni, and Carlos Guestrin. Proto-value functions: A laplacian framework for learning representation and control in markov decision processes. *Journal of Machine Learning Research*, 2006.

[26] Andrew K. McCallum. *Reinforcement Learning with Selective Perception and Hidden State*. PhD thesis, University of Rochester, 1995.

[27] Sebastian Möller. *Assessment and Prediction of Speech Quality in Telecommunications*. Kluwer, 2000.

[28] Tim Paek and David M. Chickering. The markov assumption in spoken dialogue management. In *Proc. of SIGdial Workshop on Discourse and Dialogue*, pages 35–44, 2005.

[29] Olivier Pietquin. Machine Learning Methods for Spoken Dialogue Simulation and Optimization. In *Machine Learning*, pages 167–184. 2009.

[30] Bohdana Ratitch and Doina Precup. Sparse distributed memories for on-line value-based reinforcement learning. In *Proc. of ECML*, 2004.

[31] Antoine Raux, Brian Langner, Allan Black, and Maxine Eskenazi. LET'S GO: Improving Spoken Dialog Systems for the Elderly and Non-natives. In *Proc. of Eurospeech*, 2003.

[32] Verena Rieser and Oliver Lemon. Learning and evaluation of dialogue strategies for new applications: Empirical methods for optimization from small data sets. *Computational Linguistics*, 37, 2011.

[33] David Rogers. Statistical prediction with kanerva's sparse distributed memory. Technical report, NASA, 1989.

[34] David Rogers. Weather prediction using a genetic memory. Technical report, NASA, 1990.

[35] Satinder Singh and Richard S. Sutton. Reinforcement learning with replacing eligibility traces. In *Machine Learning*, 1996.

[36] Satinder Singh, Michael Kearns, Diane Litman, and Marilyn Walker. Reinforcement learning for spoken dialogue systems. In *Proc. of NIPS*, 1999.

[37] Hiroaki Sugiyama, Toyomi Meguro, and Yasuhiro Minami. Preference-learning based Inverse Reinforcement Learning for Dialog Control. In *Proc. of Interspeech*, 2012.

[38] Dave Toney, Johanna Moore, and Oliver Lemon. Evolving optimal inspectable strategies for spoken dialogue systems. In *Proc. of NAACL-HLT*, 2006.

[39] Stefan Ultes, Alexander Schmitt, and Wolfgang Minker. Towards quality-adaptive spoken dialogue management. In *Proc. of NAACL-HLT SDCTD*, 2012.

[40] Marilyn A. Walker. An application of reinforcement learning to dialogue strategy selection in a spoken dialogue system for email. *Journal of Artificial Intelligence Research*, 12:387–416, 2000.

[41] Marilyn A. Walker, Diane J. Litman, Candace A. Kamm, and Alicia Abella. PARADISE: a framework for evaluating spoken dialogue agents. In *Proc. of EACL*, pages 271–280, 1997.

[42] Marilyn A. Walker, Irene Langkilde-Geary, Helen Wright Hastie, Jerry Wright, and Allen Gorin. Automatically training a problematic dialogue predictor for a spoken dialogue system. *Journal of Artificial Intelligence Research*, 16:293–319, 2002.

[43] Cheng Wu and Waleed Meleis. Adaptive fuzzy function approximation for multi-agent reinforcement learning. In *Proc. of IEEE/WIC/ACM IAT*, 2009.