Apprentissage par renforcement pour la recherche d'information interactive

Gérard Dupont, Sébastien Adam, Yves Lecourtier

Apprentissage par renforcement pour la recherche d'information interactive

Gérard Dupont[1], Sébastien Adam[2], Yves Lecourtier[3]

1. CASSIDIAN, Elancourt, France, gerard.dupont@cassidian.com
2. LITIS, Université de Rouen, France, <prenom>.<nom>@univ-rouen.fr

Résumé : La recherche d'information dans de grands ensembles documentaires apparaît comme un domaine de recherche en partie résolu par les approches efficaces déployées ; par exemple ; dans les grands moteurs de recherche sur Internet ou Intranet. Si celles-ci résolvent la problématique de base d'accès aux documents, il est clair qu'il existe encore de nombreux cas particuliers de recherche où les outils courants ne sont pas suffisants. Pour cela, de nouvelles approches voient le jour dans le domaine de la recherche d'information interactive.

Cette notion d'interactivité permet donc de replacer l'utilisateur au cœur du système et nécessite la mise en place de système dynamique capable d'adapter les réponses mises à la disposition de l'utilisateur. Dans cet optique, cet article présente une première expérimentation mettant en jeu l'apprentissage par renforcement pour la sélection d'outils de support à la recherche reposant sur une analyse fine des interactions et des comportements de l'utilisateur.

1 Introduction

1.1 Recherche d'information interactive

La recherche d'information dans de grands ensembles documentaires et en particulier sur Internet apparaît aujourd'hui comme un domaine de recherche en partie résolu. Des méthodes efficaces d'indexation plein texte comme le modèle vectoriel de Salton **?** couplé à des méthodes de pondération comme le TF-IDF **?** et à des stratégies d'ordonnancement efficace telle que le "page rank" **?** pour les données Web permettent aujourd'hui d'obtenir des résultats satisfaisant dans de nombreux cas. Si elles ne sont pas encore parvenues à l'étape industrielle, des méthodes encore plus efficaces basées sur des interprétations probabilistes de la langue **?** permettent encore d'obtenir de meilleures performances.

Cependant, il est aussi évident que ces approches globales ne résolvent pas tout les cas de figure complexes de recherche d'information qui peuvent être soumit par les utilisateurs. En particulier, lorsque que ces mêmes utilisateurs n'ont pas une connaissance experte du système de recherche lui-même (manque d'expérience système) ou encore quand ils ont une vision peu précise de leur propre besoin (définition floue du besoin), les réponses sont souvent décevantes. Ce problème est aussi marqué dans les cas où les utilisateurs ont une connaissance précise de leur besoin mais pas forcément du corpus ou du répertoire lexical utilisé dans celui-ci pour décrire un domaine (inadéquation du vocabulaire de l'utilisateur). Dans tout ces cas, les systèmes de recherche simple ne sont plus suffisant et il est nécessaire de basculer dans un système capable de supporter l'utilisateur dans sa recherche.

1.2 Défis actuels

Le domaine de la recherche d'information interactive, qui replace l'utilisateur au centre de la problématique de recherche d'information, tente de résoudre ces cas complexes de recherche. La tâche est de grande ampleur car la recherche d'information est un problème "à longue traîne" : les recherches simples sont aujourd'hui à peu près résolues mais il reste un nombre important de cas spécifiques.

Pour ce faire, de nombreux outils de support à la recherche ont été proposés pour aider les utilisateurs. Ils proposent un échantillon de fonctionnalités supplémentaires qui peuvent être ajoutées à un système de recherche d'information classique. Parmi la multitude d'approches existantes dans la littérature, on peut distinguer les méthodes d'adaptation de la présentation des résultats (ordonnancement personnalisé ou résumé en contexte par exemple) de celles qui vont supporter l'utilisateur dans sa recherche et sa navigation en proposant, plus ou moins directement, de nouvelles requêtes (suggestion de termes, de requêtes, facettes de recherche...).

La grande variété des modes de présentation et des fonctionnalités proposés rendent complexe l'analyse des outils de support à la recherche. En particulier, dans le cas d'un nouveau système, il est difficile de choisir parmi la multitude des approches existantes ou même de comparer leur efficacité. La simple juxtaposition de multiples approches ne permet pas d'obtenir un système satisfaisant (voir par exemple **???**). L'espace d'affichage disponible limite les possibilités mais surtout l'ajout d'un nombre trop important d'outils de support impose aux utilisateurs une surcharge cognitive néfaste à l'interprétation des informations présentées. Par conséquent une intégration intelligente passe par une solution de sélection dynamique des outils de support. Celle-ci va permettre d'adapter la présentation aux besoins de l'utilisateur au cours de sa session de recherche. Si la sélection de la meilleure approche de support au meilleur moment semble évidemment la solution idéale, il est nécessaire de bien comprendre le problème si l'on souhaite proposer une solution efficace et flexible. Deux questions apparaissent alors : Quels sont les critères qui permettent d'identifier ces "meilleurs moments" ? Et surtout, quelles sont les mesures qui permettent d'évaluer l'impact d'une approche sur les performances du système ?

Cet article a pour objectif de présenter une solution novatrice qui tente de résoudre ce problème de combinaison de multiples approches à travers la mise œuvre d'outils de suggestion de requêtes. Celle-ci repose notamment sur une analyse fine des interactions utilisateur ainsi que sur la mise en œuvre de l'apprentissage par renforcement.

2 Principes de mise en œuvre

2.1 Architecture

L'intégration de multiples outils de support à la recherche dans un système de recherche d'information interactif doit reposer sur une architecture flexible.

Elle doit pouvoir laisser les différents outils proposer des modifications sur l'ensemble de la page de résultats afin de ne pas les limiter à du ré-ordonnancement. Pour cela, on reprendra le principe que l'agent AI^2RS proposé par Jansen dans **?** où chaque module de support propose des adaptations de la présentation des résultats.

Il sera ensuite nécessaire de transmettre l'intégralité des interactions utilisateurs à chacun des modules implémentant les différentes approches de support. Pour cela nous exploiterons un nouveau modèle d'interaction en quadruplet qui se base sur l'analyse des actions utilisateur. Chaque action génère donc une observation $x(t) = (\lambda, U, t, \Delta)$ avec λ l'action en provenance de l'utilisateur, U l'ensemble des unités d'informations considérées (document ou portion de document), t le temps écoulé depuis le début de la session et enfin Δ la durée cette action. Celles-ci pourront permettre une mise à jour des éléments de support à la recherche.

La sélection des outils de support sera réalisée dynamiquement au cours de la session de recherche. Pour cela, il est nécessaire de mettre en œuvre un module spécifique de sélection et de combinaison des approches. Celui-ci exploitera les interactions utilisateurs afin de guider ses décisions et de mesurer les performances des solutions proposées. La figure **??** donne un aperçu global de la proposition.

D'un point de vue conceptuelle, la nouveauté de cette architecture fonctionnelle est le rétablissement des deux cycles dans le processus de dialogue entre le système et l'utilisateur présents dans le modèle de Belkin**?** de la recherche d'information. Ainsi le cycle de gauche correspond au rythme imposé par le dialogue requête/réponse entre l'utilisateur et le système, classiquement implémenté dans les moteurs de recherche courants. Le cycle de droite, plus court, prendra en compte l'ensemble des interactions utilisateurs telles que les clics sur chaque résultat de recherche. Il va ainsi permettre une mise à jour dynamique des informations de support à la recherche. Il s'agit donc

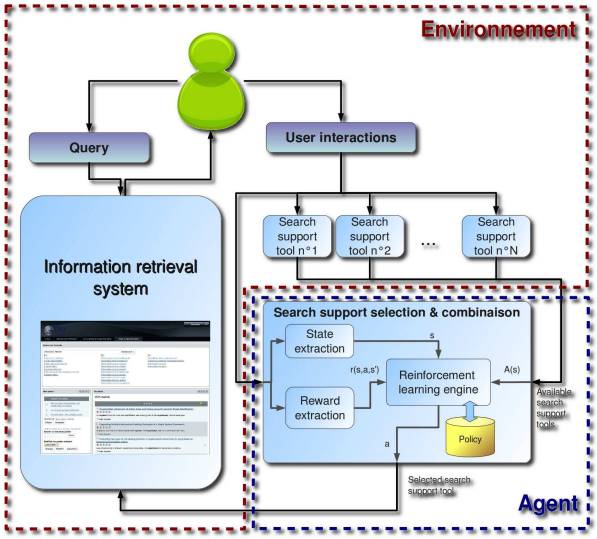
d'une généralisation par rapport au modèle de Belkin en considérant l'ensemble des interactions utilisateur et non seulement les jugements de la pertinence des résultats.

Le cœur de cette proposition réside finalement dans le bloc de sélection et de combinaison des approches de support qui va décider des approches à utiliser à chaque instant par l'intermédiaire d'apprentissage par renforcement. La sous-section suivante va détailler les différents éléments qui le composent.

2.2 Définition d'un processus de décision markovien

La figure **??** donne une illustration de la matérialisation des concepts d'agent et d'environnement sur l'architecture fonctionnelle proposée.

FIGURE 1 – Application du cadre de l'apprentissage par renforcement à la sélection d'outils de support à la recherche.



Le problème posé est donc centré sur l'utilisateur et ses interactions. C'est lui qui constituera la source d'information utile à la construction du modèle de décision et c'est encore lui qui réagira aux décisions effectuées. Par conséquent, l'utilisateur sera naturellement associé à l'environnement tel qu'il est défini dans le cadre de l'apprentissage par renforcement. Dans un souci d'indépendance des composants, le système de recherche ainsi que les différents modules de support seront aussi inclus dans cet environnement. Ainsi le cadre proposé restera indépendant des implémentations de ces différents composants.

L'agent sera alors naturellement associé au module de sélection/combinaison. Il aura pour objectif de sélectionner les éléments de réponse fournit par les outils de support qui tenteront de soutenir l'utilisateur soit directement en répondant à son besoin d'information soit en lui permettant d'avancer dans sa recherche. Ainsi il est simple d'envisager les actions à sa disposition puisque celles-ci correspondront simplement aux différentes combinaisons d'outils à sa disposition et donc aux différentes configurations de support à la recherche qu'il pourra présenter.

L'état de l'environnement sera décrit de manière indirecte à travers les interactions que celui-ci aura avec le système et à travers certains éléments qui pourront provenir du moteur de recherche notamment des données sur le corpus et les ensembles de résultats à chaque nouvelle requête.

2.2.1 Actions

Les actions à disposition de l'agent seront naturellement les différentes possibilités de support à la recherche dont il disposera. Celles-ci peuvent être regroupées dans des catégories distincts. Ainsi il faudra distinguer le cas de la sélection d'un outil parmi l'ensemble d'outils concurrents appartenant à une même catégorie de la combinaison d'outils de catégories différentes.

D'une manière générale, on supposera que la décision se fera dans un ensemble de solutions de dimension N, ce qui correspondra donc à la taille de l'ensemble $\mathcal{A}$ des actions à disposition de l'agent. Ceci correspondra à la totalité des combinaisons possibles de solutions incluant le choix de ne rien présenter. L'évaluation de N apparaît comme un paramètre important de la solution proposée du fait de l'impact sur la complexité de l'apprentissage par renforcement. En posant m le nombre de catégories d'outil de support et k_i le nombre de possibilités concurrentes pour la catégorie i, on a $N = \prod_{i=1}^{m}(k_i + 1)$. On notera que selon les capacités des outils de support, il est possible que certains ne puissent pas proposer de réponse (ou pas de mise à jour possible) avec les dernières interactions. Par conséquent N sera la borne maximale de la taille de l'ensemble de solutions, mais celui-ci sera en général plus réduit.

Dans l'application proposée, nous limiterons l'ensemble des actions à une seule catégorie de support à la recherche, la suggestion de requêtes, et donc aux 4 approches distinctes : **local** qui exploite un modèle de retour de pertinence implicite, **blind** reposant sur un modèle de retour de pertinence aveugle (toutes deux présentées dans **?**), **facet** qui correspond à la présentation de facettes de recherche calculés sur la liste de résultats et **spell** qui propose de la correction orthographique. Ce choix est notamment motivé par le cadre expérimental de cette étude. Nous avons volontairement pris un ensemble d'actions de taille réduite (ici $N = 4$) afin de limiter la complexité du problème d'apprentissage. Ainsi, la quantité de données nécessaire à une convergence vers la solution optimale sera plus réduite.

2.2.2 Récompenses

La fonction de récompense immédiate, notée $r(s, a, s')$, correspond à un signal provenant de l'environnement qui permettre à l'agent de juger de la qualité de sa dernière action vis-à-vis de son objectif au long terme. L'algorithme de résolution du MDP va en effet avoir pour but d'optimiser l'accumulation au long terme de cette récompense.

Dans la cadre du problème considéré et si on se limite à des outils de suggestion de requêtes, une implémentation de la récompense pourrait être de favoriser la sélection des requêtes suggérées par l'utilisateur. On utiliserait donc simplement la fréquence moyenne de sélection d'une suggestion pour chaque combinaison $\{s, a, s'\}$. Cependant du point de vue du problème d'apprentissage, l'objectif glisserait alors vers la présentation de requêtes "attirantes" sans préjuger de leur effet sur le processus de recherche d'information ni même sans prendre en compte l'interprétation des interactions suite à l'exécution de cette requête. On privilégiera donc les implémentations qui ne dépendent pas des outils de support à la recherche afin de conserver la validité du cadre proposé quelle que soit la configuration finale du système.

Les contraintes sur cette fonction sont qu'elle doit être définie pour toutes les combinaisons états/actions et donc après chaque sélection d'une nouvelle action. Si l'utilisation d'une récompense calculée à posteriori est compatible avec le cadre des MDP, elle ne permettra pas un apprentissage en ligne (en cours de session). Par conséquent cette récompense ne doit pas dépendre de paramètres calculés uniquement en fin de session mais seulement au fil de l'eau.

Formulation exploitant le rang moyen des documents manipulés

L'un des objectifs d'un système de recherche d'information est de proposer en haut de la liste de résultats des documents satisfaisants. Il s'agit de favoriser les trajectoires où l'utilisateur interagira avec les documents ordonnés en haut des liste de résultats. Cependant, les utilisateurs ayant naturellement tendance à interagir avec ces mêmes documents, nous utiliserons la différence entre le rang du document courant et le rang moyen des documents classiquement manipulés par les utilisateurs. Ainsi la formulation sera :

$$r(t) = \frac{\omega_{\lambda_t}}{(rank(u) - \hat{rank}} \tag{1}$$

On notera évidemment que cette formulation repose sur l'estimation d'un rang moyen calculé sur une ensemble de sessions passées. Elle ne pourra donc être utilisée que dans les cas où suffisamment de données auront été récupérées.

Formulation basée sur l'entropie des résultats

Dans **?**, Boldareva, De Vries et Hiemstra propose une nouvelle métrique de suivi de performance dans les systèmes de recherche d'information. Celle-ci se base sur la notion d'entropie, issue entre autre des travaux de Shannon sur la théorie de l'information (voir **?**). L'idée présentée dans **?** est qu'il est possible d'interpréter une liste de résultats par l'intermédiaire de cette entropie. Les scores (qui sont des probabilités dans le cas de l'étude) sont donc traités comme un signal sur lequel l'entropie est évaluée. Les grandes valeurs seront interprétées comme de l'incertitude quand à la pertinence réelle des résultats dont les scores ne peuvent discriminer chaque résultat. En revanche une faible entropie signifiera que les scores sont "concentrés" et donc que le système est plus sûr de la réponse qu'il apporte à la requête en cours.

Dans le cas d'un système non probabiliste, les scores donnés pour chaque résultat ne correspondent pas à des probabilités. Par conséquent il n'est pas possible de calculer une entropie "réelle". Nous proposerons donc une approximation de cette entropie sur les 10 premiers résultats en utilisant les scores fournis par le système. La formulation ne correspond plus à une vraie entropie mais elle va permettre de mesurer la "concentration" des scores dans les résultats ayant un haut rang. Plus que l'entropie absolue nous étudierons l'évolution de celle-ci en calculant la différence d'entropie à chaque étape comme cela est proposé dans **?**. Ainsi la formulation finale de la récompense sera :

$$\begin{aligned} r(t) &= \Delta_{\hat{H}_t} \\ \text{avec : } \Delta_{\hat{H}_t} &= \hat{H}_t \hat{H}_{t1} \\ \text{et : } \hat{H}_t(X) &= -\sum_{i=1}^{10} score(x_i) \log score(x_i) \end{aligned} \tag{2}$$

Avec $score(x_i)$, le score du $i$ème résultat normalisé par le plus grand score de la liste proposée à l'instant t. On notera que conformément au conclusion de Boldareva, cette pseudo-entropie ne mesure pas véritablement la capacité de discrimination du système mais plutôt celle des requêtes qui seront soumises et que celle-ci dépendent évidemment de l'utilisateur. L'idée est donc de favoriser les synergies entre l'utilisateur et le système de façon à faire évoluer la pseudo entropie dans le bon sens - à savoir une réduction de l'entropie et donc de l'incertitude dans les résultats.

2.2.3 États

La définition des états de l'environnement joue un rôle crucial dans le cadre des MDP. Ainsi ils se doivent d'être pertinents vis à vis de l'objectif que l'agent doit atteindre afin qu'il puisse les exploiter convenablement pour construire son modèle de décision.

Pour la plupart des problèmes traités en apprentissage par renforcement, les états de l'environnement découlent naturellement de mesures faites sur le processus considéré. Dans le cadre considéré de la recherche d'information interactive, l'objectif est de satisfaire le besoin d'information utilisateur en le supportant dans son processus de recherche. S'il est simple à comprendre, il est difficile de le lier à des mesures particulières.

Comportement et état cognitif

L'état de l'environnement serait idéalement associé au modèle cognitif de communication proposé par Belkin dans **?** et exploité dans son modèle d'interaction utilisateur présenté dans **?**. Celui-ci repose sur une combinaison du modèle de communication de Shannon **?** et du modèle de recherche d'information de Robertson **?**. Au lieu de centrer l'analyse sur les aspects systèmes (les canaux de transmission pour le modèle de Shannon et système de recherche basé sur un calcul de similarité pour celui de Robertson), il propose un point de vue cognitif. Dans ce contexte, le problème de la recherche d'information est vue comme celui de l'échange d'information entre un auteur (humain) à l'origine de textes et un lecteur (toujours humain).

Si l'auteur est supposé avoir la maîtrise de l'information qu'il diffuse, ce n'est en général pas le cas du lecteur qui est alors face à un manque de connaissances qui le pousse à rechercher de l'information. Belkin associe à ce manque un "état anormal des connaissances" ("*anomalous state of knowledge*" en anglais) qui aura un impact sur la stratégie de recherche du lecteur d'une part et

sur son mode d'interprétation des résultats proposés par le système d'autre part (voir les modèles de stratégie de recherche proposés dans **?**). Au final c'est donc au travers de son comportement et de ses interactions avec l'information présentée que l'utilisateur va révéler les évolutions de son état cognitif. On retrouve ici les aspects d'états implicites qui se manifestent au travers des interactions explicites.

Pour revenir au problème considéré, il est clair qu'il ne sera pas possible d'accéder directement et encore moins d'altérer l'état cognitif de l'auteur. Cependant "l'état anormal des connaissances" du lecteur et utilisateur du système de recherche va être accessible et utilisable à travers les interactions avec le système. Être capable, par exemple, de distinguer un utilisateur qui souhaite découvrir globalement une thématique ou bien approfondir un sujet précis permettrait de sélectionner dans un cas un outil de suggestion de requêtes exploratoires et dans un autre un outil de suggestion de requêtes discriminantes.

Ces états n'étant pas accessibles directement et il sera nécessaire de les inférer à partir des données d'interactions. Ce sont ces mêmes états que nous proposons d'exploiter pour le MDP afin de construire le modèle de sélection d'outil de support à la recherche.

Partitionnement des données d'interaction

Afin de découvrir ces différents schémas de comportement, nous incluons donc dans notre système un certain nombre de capteurs menant à la récupération de plus 24 caractéristiques liées aux différentes actions de l'utilisateur. On notera que celles-ci ne reposent que sur 4 actions de bases (requête, lecture des résultats, ouverture d'un document et sélection d'un document) qui se retrouve dans la majorité des systèmes et qui sont indépendantes du média de support des documents. À travers les mesures effectuées sur les interactions utilisateurs, nous proposons de mettre en œuvre un algorithme de partitionnement modélisant (basée sur l'algorithme espérance-maximisation, voir **?**). Celui-ci a pour objectif de regrouper ensemble les mesures ayant une forte corrélation afin de détecter des ensembles cohérents.

Dans la littérature, les méthodes supervisées, précédées d'une analyse manuelle, sont souvent privilégiées pour l'analyse de comportements utilisateurs. Ici, le choix d'une méthode de partitionnement (donc non supervisée) pour l'analyse des interactions se justifie par l'objectif qui est de découvrir des comportements ou des états de comportement très fins et/ou non identifiables par un expert. Certains travaux issus de la littérature ont tout de même proposé une méthodologie similaire de partitionnement. On peut notamment citer **?** ou **?** qui exploitent les résultats de partitionnement sur des session de navigation Web pour le premier et sur des sessions de recherche pour le second. Cependant ces approches sont différentes du fait que les sessions sont prises dans leur globalité et qu'elles ne font pas d'analyse "interne" à la session n'est faite. A ce jour aucune étude de partitionnement n'a donc été porté à notre connaissance et c'est ce qui constitue l'une des originalités de cette approche.

Cette architecture propose donc de s'attaquer au problème de sélection d'outils de support à la recherche tout en restant indépendante à la fois des outils eux-mêmes amis aussi de la manière dont l'environnement est décrit. L'objectif étant de mettre en œuvre cette proposition dans un système de recherche d'information interactif, nous avons opté pour un algorithme de résolution de MDP permettant à al fois un apprentissage de la dynamique de l'environnement et de la politique de décision. Pour cela, nous avons utilisé la librairie open source PIQLE [1](issu des travaux de F. De Comité du LIFL) qui propose une variante du Q-learning issu de **?** et auquel certaines adaptations ont été effectuées afin de pour permettre la récupération du modèle appris et le calcul de prédiction sous contrainte d'une action choisie.

3 Expérimentation utilisateur interactive

L'apprentissage par renforcement repose sur le principe d'expériences face à un environement qui dans le cadre considéré est un utilisateur en recherche d'information. Dans ce contexte, la

1. http://piqle.sourceforge.net/

principale difficulté de l'évaluation réside donc dans la nécessité de tester le système face à des utilisateurs dans une situation réaliste de recherche d'information.

3.1 Principes de l'évaluation

Le principe retenu pour l'évaluation repose sur deux idées maîtresses :
- Il est plus important de tester le système sur des cas proches de la réalité (utilisateurs réalistes mis en situation de recherche interactive) que d'augmenter le volume des données de façon artificielle (utilisation d'utilisateurs simulés) ;
- Il est inutile de faire une évaluation du système proposé s'il n'est pas suffisamment entraîné.

Pour exploiter au maximum les données d'interaction récupérées, les expérimentations ne seront donc effectuées que sur le système unique de référence (sans apprentissage) qui servira donc à la génération d'un maximum de données d'interactions. Ces données seront alors exploitées par le système apprenant, d'une part pour l'apprentissage, et d'autre part pour le test. Ainsi le principe d'évaluation retenue est une solution hybride entre les techniques dévaluations de systèmes de recherche d'information interactive, dont on conserve le protocole, et les évaluations classiques des méthodes d'apprentissage statistiques dont on utilise les méthodes de mesures de performances.

3.1.1 Métrique

Sachant qu'il ne sera alors pas possible de comparer les deux systèmes, l'objectif de l'évaluation sera de démontrer que le système proposé est capable "d'apprendre". Dans le contexte des algorithmes d'apprentissage par renforcement il s'agit d'être capable de donner des estimations fiables des récompenses que l'on peut atteindre à chaque décision du système. Par conséquent, il est nécessaire de tester le système sur des sessions réelles afin de comparer, à chaque étape, les prédictions du système à la récompense effectivement obtenue. Pour une session d'interaction X, on calculera donc l'erreur absolue de prédiction donnée par :

$$\zeta = \sum_{t}^{[0,T]} |r^*(s(t), a(t), s(t+1)) - r(t+1)| \tag{3}$$

Avec $r^*(s(t), a(t), s(t+1))$, la prédiction de la récompense faite par le système sachant l'état $s(t)$, l'action $a(t)$ choisie pendant l'expérimentation et l'état $s(t+1)$ auquel le système est alors parvenu. La valeur réelle de la récompense obtenue au cours de la session est donnée par $r(t+1)$.

3.1.2 Méthode de mesure de performance

Afin d'obtenir une estimation plus fiable de l'erreur moyenne du système, on procédera à une évaluation croisée sur l'ensemble des données disponibles. Ainsi, parmi les N sessions d'interactions dont nous disposerons à l'issue de l'expérimentation sur le système de référence, nous sélectionnerons une session particulière qui sera utilisé pour le test, les autres sessions servant à l'entraînement préalable du système. Ce processus sera être répété autant de fois qu'il y a des sessions soit N.

Enfin, pour visualiser l'amélioration des performances et donc évaluer l'impact de l'apprentissage sur les performances, nous ferrons une comparaison de la métrique d'erreur de prédiction sur chaque session de tests mais selon différentes configurations d'apprentissage. Ainsi, on fera varier le nombre m de sessions utilisées pour l'apprentissage de 1 à $N-1$ et nous comparerons les performances pour chaque cas. L'hypothèse étant évidemment que les performances s'améliorent lorsque le système apprend plus. Il existe cependant un nombre important de façons de sélectionner les m sessions utilisées pour l'apprentissage (correspondant à la sélection de m sessions parmi $N-1$ soit $\frac{(N-1)!}{m!(N-1-m)!}$ possibilités) et l'apprentissage sur des sessions différentes peut potentiellement avoir un impact sur les performances à nombre de sessions d'apprentissage égal. Par conséquent nous moyennerons pour chaque valeur de m, les performances obtenues pour k sélection différentes.

Au final, l'évaluation du système sera effectué k fois pour un nombre m de sessions d'apprentissage (sachant $m \in [1; N-1]$) et sera testé sur la session n (sachant $n \in [1; N]$).

3.2 Protocole d'évaluation interactive

L'expérimentation a placé des utilisateurs dans une situation réaliste de recherche d'information. Les utilisateurs envisagés sont des veilleurs ou des analystes du renseignement en charge de compléter un dossier sur un sujet à priori peu connu à l'aide de documents collectés sur Internet. Pour cela, nous avons choisi le thème de l'intervention américaine en Afghanistan qui est important d'un point de vue médiatique.

3.2.1 Corpus

Le corpus a été constitué par une collecte de documents issus des deux internet (communiqué de presse de l'ISAF [2] ; un site de diffusion d'information sur les opérations des marines américains [3]). La collecte a donc permis de récupérer 3518 sur le site ISAF et 7724 pour marines corps news soit un total de plus de 10 000 documents indexés sur la plate-forme WebLab.

3.2.2 Tâches de recherche

Les tâches de recherche ont été définies en suivant la même approche que pour l'expérimentation de la partie précédente à savoir en suivant le protocole expérimental "hybride" défini par Borlund **?**. Elles constituent donc l'expression de besoins d'information vis à vis du domaine considéré sous la forme d'une ou plusieurs questions. Pour faire varier le niveau de difficulté, nous avons distingués 3 types de tâches selon la présence ou l'absence des mots clés de la question dans les documents contenant la ou les réponses. Au final, 18 tâches réparties dans les trois niveaux de difficultés mentionnées ont été exploitées pour les besoins de l'expérimentation.

3.2.3 Utilisateurs

Deux critères ont donc été choisis pour sélectionner les volontaires : [a)]

un niveau de formation universitaire de deuxième cycle à minima ;

un niveau de compétence en anglais et des outils de recherche sur l'internet supérieur à la moyenne. L'idée bien sûr que ces critères correspondent à des veilleurs ou analystes du renseignement. On notera de plus qu'il a été porté une attention particulière au domaine de spécialité des personnes recrutées afin d'éviter les profils trop "informaticiens". Ils sont en général les sujets les plus couramment recrutés pour des expérimentations de recherche d'information mais ne sont, en réalité, pas représentatifs des utilisateurs attendus. Le recrutement des utilisateurs s'est fait par l'envoi d'e-mails auprès de listes fermées comportant des professionnels de divers pays d'Europe.

3.2.4 Déroulement d'une session d'expérimentation

L'expérimentation ayant eu lieu à distance, le déroulement des étapes était contrôlé par un module de gestion spécifique et présenté à tout les onglets. A chaque étape, celui-ci présentait donc un message particulier relatif aux instructions à suivre. Après un court entraînement au système, l'utilisateur devait répondre à 3 tâches successives présentées sous la forme d'une question ouverte et d'un formulaire demandant la réponse trouvée ainsi que les références des documents ayant servi à trouver la réponse. Un temps indicatif de 15 minutes était suggéré et sans réponse au-delà de ce délais, l'utilisateur était invité à passer à la tâche suivante. Dans la pratique l'investissement des utilisateurs a été assez fort, certains ayant passés plus d'une heure sur une tâche complexe. Entre chaque tâche, un questionnaire intermédiaire, très court, permettait de récupérer des informations sur la session de recherche effectuée. Un questionnaire final terminait ensuite l'expérimentation.

4 Résultats

L'expérimentation a permis de récupérer plus de 63 sessions de recherche exploitables($N = 63$). Le système a donc été entraîné progressivement sur m sessions (avec $m \in [1; 62]$) selon la

2. Force internationale d'assistance et de sécurité de l'OTAN : http://www.nato.int/isaf/docu/pressreleases/
3. http://www.marine-corps-news.com/

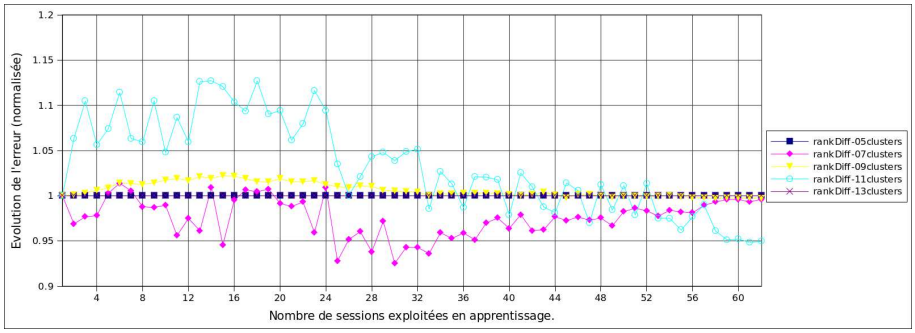
méthode “leave-one-out” puis testé sur l’une des sessions restantes. Afin de masquer les variations qui pourraient résulter du choix des m sessions d’apprentissage, nous avons répété 1000 fois ($k = 1000$) les tests puis moyenné les performances obtenues.

4.1 Choix de la récompense et des états

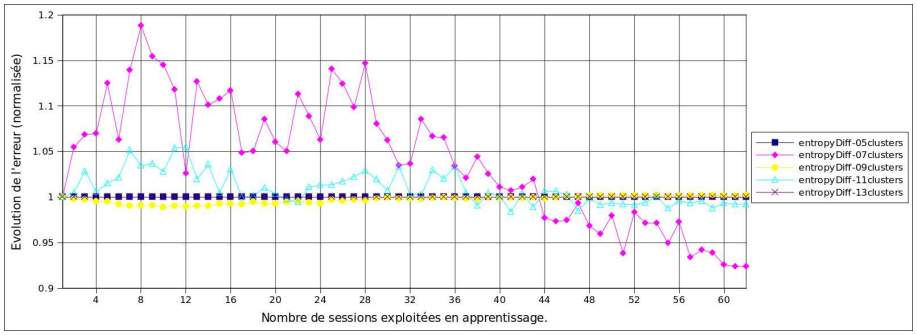
Nous avons tout d’abord étudié d’une part le choix d’implémentation de récompense et d’autre part le nombre de partitions utilisés pour décrire les états de comportements. Les différentes formulations de récompense proposées sont toutes, en théorie, bénéfiques pour le système. Cependant, la maximisation de celles-ci va nécessairement dépendre de l’adéquation des variations de la récompense et de la succession des états. Si les états de comportements définis de correspondent en rien aux variations de récompense, alors le système par renforcement ne pourra apprendre. L’objectif était donc de voir si certaines configurations étaient plus favorables que d’autres au système d’apprentissage par renforcement en montrant que l’erreur de prédiction obtenue lorsque le système dispose de plus de données d’apprentissage.

Les deux graphes suivants (figure **??**) montrent l’évolution de l’erreur de prédiction pour les deux propositions de récompense. On remarque que l’évolution apparaît plus positive sur certaines configurations des états.

FIGURE 2 – Évolution de l’erreur de prédiction de la récompense en fonction du nombre de sessions apprises, des états de comportement détectés et de l’implémentation de la récompense.



(a) Récompense basée sur le rang des documents manipulés “rankDiff".



(b) Récompense basé sur la différence d’entropie “entropyDiff”.

La récompense “rankDiff" ne semble pas permettre une réduction trés efficace de l’erreur d’estimation hormis pour le cas exploitant 11 partitions pour décrire les états de comportement. Dans ce cas, l’entraînement du système sur 62 de sessions (plus une pour le test) permet une erreur d’estimation de 5% inférieure à la valeur initiale et presque 20% par rapport à la valeur maximale. La récompense basée sur la différence d’entropie (“entropyDiff”) semblent elle être mieux apprise dans le cas de la définition des états de comportement sur 7 partitions. Si pour un apprentissage restreint à peu de sessions, l’erreur a encore une fois tendance à augmenter, elle diminue ensuite progressivement de façon quasi monotone pour atteindre une réduction de plus de 25% par rapport

à l'erreur maximale.

On remarquera que l'apprentissage est plus efficace pour un nombre variable de partitions pour chaque implémentation de la récompense. Chaque récompense ayant un mode de variation propre, il est normal que le découpage des états de comportement soit à adapter pour chacune d'entre elle. Cependant, deux observations sont intéressantes à faire à ce niveau :

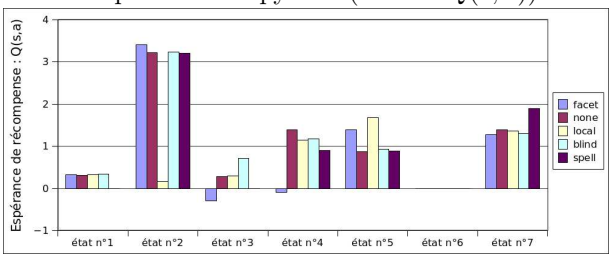
- le meilleur partitionnement au sens du partitionnement (5 partitions) n'est pas celui qui permet d'atteindre les meilleures performances ;
- certaines configurations d'états de comportement semblent tout de même permettre une meilleure interprétation de la dynamique des récompenses.

Ceci permet de confirmer que l'approche de partitionnements sur les données d'interaction est intéressante, même s'il serait nécessaire d'approfondir la manière de mieux adapter ceux-i à chaque récompense.

4.2 Résultats de l'apprentissage

Si la prédiction de la récompense semble possible, il est important de vérifier que les valeurs obtenues de $Q(s, a)$ permettent une discrimination intéressante de chaque action pour chaque état afin de vérifier que les actions proposées permettent bien d'influencer l'environnement. Pour cela, nous avons ré-entraîné le système sur l'ensemble des sessions sur la meilleure configuration, à savoir celle utilisant la récompense basée sur la différence d'entropie et la segmentation en 7 états de comportements. Au final, nous obtenons donc l'histogramme présenté à la figure **??**.

FIGURE 3 – Espérance de récompense "entropyDiff" (valeurs $Q(s, a)$) obtenue après apprentissage.



On observe que les estimations de récompense au long terme, donnée par les valeurs $Q(s, a)$, sont variées pour chaque action. Ainsi, il est clair que l'algorithme est parvenu à faire la distinction entre les effets des différentes actions. Or la section précédente a montré que ces valeurs correspondent à des prédictions de récompenses qui s'améliorent avec l'arrivée de nouvelles données d'apprentissage. Le système sera donc capable de prendre des décisions discriminantes sur des prédictions de plus en plus précise. L'apprentissage par renforcement permettra donc d'optimiser la présentation de outils de suggestion de requête au sens de la récompense exploitée.

4.3 Analyse des séquences apprises

Des observations supplémentaires peuvent être faite pour chaque état en exploitant le modèle de transition des séquences d'interaction collectées. Ce réseau, illustré à la figure **??**, présente donc les transitions dominantes entres les états (en flèches pleines) associées à leur probabilité de transition. Les transitions secondaires (dont la probabilité est inférieur à 10%) sont représentées en gris pointillé.

D'après l'histogramme de la figure **??**, seule l'action de suggestion "spell" paraît intéressante pour l'état (7). Or, le réseau des transitions nous indique que cet état est toujours le premier dans les séquence d'interaction. Par conséquent, l'impact positive de l'action qui propose de corrections orthographiques devient évident : c'est à la première requête que l'utilisateur aura tendance à faire le plus de fautes de frappes puisqu'il aborde un thème nouveau et construit souvent une requête complètement nouvelle (c'est-à-dire sans réutiliser des mots-clés précédents), de ce fait, la suggestion des variantes orthographiques ont plus de chance d'avoir un effet bénéfique.

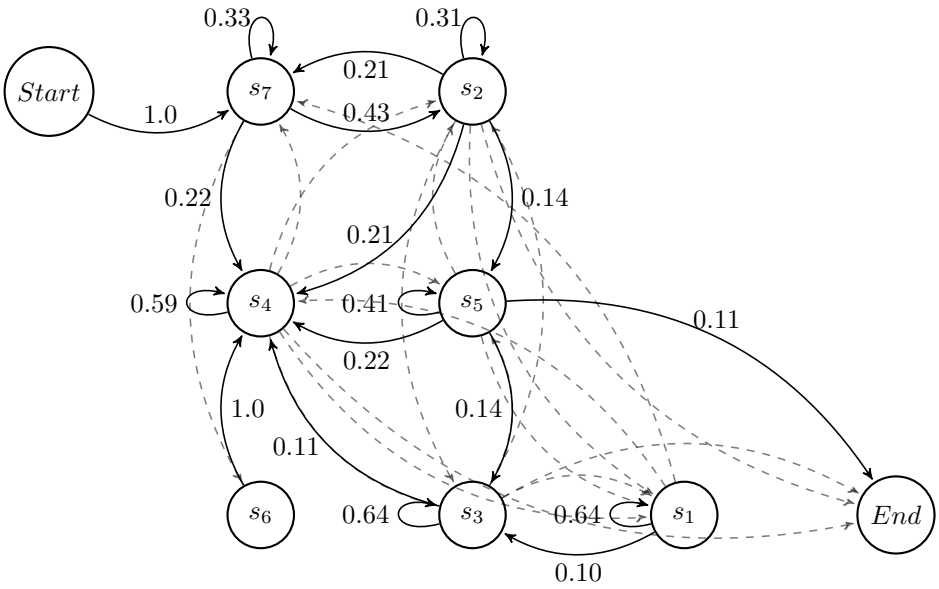


FIGURE 4 – Chaine de markov donnant les transitions entre 7 états découverts en ignorant les actions (les transitions en pointillé ont un probabilité d'occurance inférieure à 10%).

Inversement, le mode "local" est le plus performant pour l'état (5) qui est aussi l'état qui a le plus de chance d'être suivit par une fin de session (avec une probabilité de 11% soit plus de deux fois plus que le cas suivant qui est la transition de l'état (4) vers la fin de session avec une probabilité légèrement à 5%). Or ce mode exploitant les interactions utilisateurs, il est normal qu'il soit plus pertinent lorsqu'il a bénéficié de plus de données, c'est-à-dire en fin de session. C'est le cas inverse qui est constaté pour l'état (2) qui a de grandes chances d'être en début de session au vu des probabilités de transitions en provenance et vers l'état (7).

Néanmoins, l'analyse ne permet pas d'interpréter tout les résultats et notamment les valeurs nulles pour l'action "spell" aux états (1) et (3) (sans doute dut à l'absence de suggestion de reformulation orthographique et donc l'impossibilité d'exploiter l'action pour ces cas) ni les valeurs négatives pour les suggestions "facet" aux états (3) et (4).

5 Conclusions et perspectives

L'expérimentation interactive qui a été menée a permis de valider l'approche proposée pour le problème de sélection dynamique d'outils de support à la recherche. La définition des états de comportements basée sur un partitionnement s'est montrée efficace pour l'apprentissage les variations de la fonction de récompense basée sur la différence d'entropie. Le système est donc capable d'apprendre correctement les variations de la récompense et d'améliorer son estimation de celle-ci pour des nouvelles séquences. D'autre part, l'apprentissage permet de faire une distinction convenable entre les différentes actions et leur impact sur l'environnement. De ce fait, il est clair que cette configuration doit permettre une amélioration au long terme des performances des utilisateurs sur ce système et donc une utilisation optimisée de chaque mode de suggestion.

Bien sûr les performances réelles du système n'ont pu être complètement analysées et des expérimentations complémentaires seraient sans doute nécessaire. Dans un premier temps, on pourrait tenter d'améliorer la méthode de description des comportements en adaptant la solution de partitionnements pour qu'elle s'adapte plus en profondeur à la récompense sélectionnée. D'autres algorithmes d'apprentissage pourraient aussi être testé afin notamment de réduire la courbe d'apprentissage relativement longue du Q-learning qui impose d'avoir accès à une quantité importante de données. Enfin une autre expérimentation interactive doit être menée afin de comparer le système avec apprentissage avec un système de référence sans apprentissage afin de mesurer l'impact

sur les performances des utilisateurs dans leur tâche de recherche d'information.