Utilisation Du Service Web Google Dans La Reformulation Requête Par Les Algorithmes Génétiques

FAHSI Mahmoud [1], LEHIRECH Ahmed [2], MIMOUNE Malki[3],

[1,2,3] Laboratoire EEDIS
Département d'informatique
Université Djillali Liabes
Sidi-Bel-Abbès, 22000, Algérie.
[1] Mfahci@gmail.com, [2] Elhir@yahoo.com, [3] malki@gmail.com

Résumé : Avec l'immense volume de données existant sur le web, 84 % des Webnautes utilisent des moteurs de recherche d'ordre général pour trouver les données qu'ils exigent. Mais ces derniers n'adaptent pas leurs stratégies de recherche aux différents utilisateurs, la requête en entrée étant le plus souvent ambiguë. Ces moteurs de recherche retournent donc, des milliers de pages, chacune pertinente dans son contexte. Ce travail vise l'utilisation des services web google pour implémenter une technique automatique de reformulation requêtes utilisateurs, avec les algorithmes génétiques.

Mot clés : recherche d'information, reformulation de Requête, évaluation de Requête, Google service, algorithmes génétiques.

Introduction

Le nombre de documents existants sur le web représente un défi pour les systèmes de recherche d'information (Information Retrieval Systems). Ainsi, la croissance explosive du nombre de documents publiés dans le web, a rendu les Moteurs de recherche le seul moyen d'initier une interaction avec l'Internet. Certe, il y a beaucoup de bons moteurs de recherche, mais les utilisateurs ne sont pas toujours satisfaits des résultats fournis par ses systèmes, car dans plusieurs cas, les résultats retournés ne sont pas significatifs par rapport à leurs besoins, ceci forcera les utilisateurs à examiner de longues listes de résultats afin de localiser le ou les documents désirés.

Cependant, quand l'utilisateur utilise un moteur de recherche, il a besoin de raffiner sa requête quoiqu'il ne soit rarement spécialiste du domaine de recherche pour offrir la reformulation adéquate [1]. Même si un utilisateur moyen a une idée sur ce qu'il

cherche, ce dernier ne possède pas un vocabulaire assez consistant qui peut l'aider à bien rediriger la requête. Les utilisateurs ne peuvent donc pas prédire, quel ensemble de mots clés peut préciser la collection de documents désirés. Afin d'adapter la recherche aux besoins, une bonne reformulation des requêtes de recherche sur le web doit correspondre les termes pertinents dedans les documents aux besoins des utilisateurs.

Récemment, dans la communauté d'Intelligence Artificiel, beaucoup de recherches ont étudié la contribution de l'IA à résoudre ce problème. L'idée de moteurs de recherche personnalisés, et de systèmes de recommandation ont étaient tous largement acceptées parmi les utilisateurs, ces dernier se trouvent à exiger l'aide dans la recherche, le tri, la classification, le filtrage et le partage de la quantité énorme d'information aujourd'hui disponible sur le Web.

Dans ce travail, nous proposerons une approche d'expansion de requête qui s'appuie sur les techniques d'Intelligence Artificielle, précisément les Algorithmes Génétiques qui on la capacité d'explorer l'espace variés de requêtes candidats pour une éventuelle recommandation toute en s'adaptons au profile (déterminé par le feedback) et aux besoins utilisateurs (exprimé dans sa requête) qui sont généralement imprécis. L'approche consiste a utilisé une ontologie lexicale *"Wordnet"* afin d'ajouter tous le contexte possible susceptible à ne pas apparaître dans les réactions de pertinence (feedback), et l'utilisation d'un Algorithme Génétique (GA) [2] qui explore l'espace de recherche afin de déterminé la meilleur requête qui correspond le mieux au besoins a l'aide d'une fonction de fitness floue[17].

1. Travaux Relatifs

Les systèmes de recherche d'information intègrent aujourd'hui des processus actifs *d'adaptation aux besoins des utilisateurs* sous forme d'agents appelés « assistants personnels » de recherche d'information, Ayant comme objectifs intrinsèquement liés, d'assister l'utilisateur dans le parcours d'un ensemble de liens hypertextuels, de l'aider à raffiner et formuler une requête dans un moteur de recherche, filtrer et réordonner la liste des documents retournés par ces moteurs.

Les principaux papiers initiaux sur des réactions de pertinence utilisent tous des modèles vectoriels apparaissent dans [3], y compris la présentation de l'algorithme [4] et la variante de *"Ide Dec-Hi"* avec l'évaluation de plusieurs variantes [5]. Une autre variante prend en considération *tous* documents de la collection, non jugé approprié et le considère comme étant non appropriés, plutôt que prendre seulement en considération ceux que l'on juge explicitement non approprié. Cependant, [6] et [7] décrivent le taux de synonymie et mettent en preuve que de meilleurs résultats

sont obtenus en utilisant seulement des documents évalués par l'utilisateur plutôt que tous les documents.

Dans [8], on observe l'interaction de l'utilisateur avec des logiciels standard quotidiens -comme des navigateurs et des outils de traitement de texte- et sera considéré comme information appropriée de contexte, et produit des requêtes utilisateurs. Une interface a été proposée et par laquelle l'utilisateur peut poser des requêtes explicitement aux moteurs de recherche.

L'approche de Zhang [9] apprend les intérêts de l'utilisateur en observant leur comportement pendant l'interaction avec le système. Le système est alors formé sur les réactions explicites de l'utilisateur. Après cette phase d'étude, le système estime les réactions de pertinence implicitement basées sur les observations des actions d'utilisateur. Les estimations obtenues seront utilisées pour modifier le profil utilisateur qui sera utilisée par la suite, pour la construction de la requête.

Klink [10] Utilise l'approche de réactions de pertinence pour fixer les préférences utilisateur. Le système étend chaque expression avec l'utilisation de concept existant dans les résultats évalués pour produire une nouvelle requête qui est présentée à l'utilisateur pour confirmation. Après l'étape de la confirmation, les requêtes ainsi reformulées, seront soumises aux moteurs de recherche.

Une approche récente est celle de [11], UCAIR [12] et [13], deux systèmes qui utilisent un agent web côté client et qui peut exécuter des réactions implicites désirées, par exemple, l'expansion de requête basée sur des requêtes précédentes et le reclassement immédiat des résultats basés sur le clic pour fournir une recherche personnalisée.

Les techniques d'intelligence artificielle sont introduites pour ajuster le poids des termes par les techniques d'apprentissage : Les réseaux de neurones [14], les probabilités Bayésiennes [15] ou les algorithmes génétiques [16]. La logique Flou est aussi utilisée soit avec les algorithmes génétiques de réaction de pertinence - Local - [17] ou soit avec les ontologies via le contexte-- Global - [18] pour évaluer les poids des termes.

2. Le Moteur de Recherche Google

Google fut créé par Larry Page et Sergey Brin [19], et il est utilisé comme moteur de recherche principal. A cet effet, certaines des notations offertes par Google sur les définitions qui sont utilisées sont décrites ci-dessous :

- Google ne tient pas compte des accents ni des majuscules ou minuscules (la casse), peu importe donc que vous tapiez faites de l'internet ou Faites de l'Internet, réseau ou réseau.
- "OR": Google supporte l'opération logique "OR" pour récupérer les pages qui contiennent l'un des mots de recherche en utilisant la majuscule "OR" entre ces derniers.
- Termes Négatifs : Google utilise le symbole '-' pour assurer que le mot ne sera pas le présent dans des documents retournés.

La dernière innovation de Google est celle qui permettra aux programmeurs de développer leurs propres interfaces et agents compatibles avec les moteurs et les portails. Il s'agit des langages API.

2.1. l'API de Google Web Service

La première API de Google [20] est basé sur les services web et le SOAP a même remplacée en 2006 par Ajax API, elle reste un ensemble fiable d'outils mis à qui permet d'interroger à distance les serveurs du moteur de recherche Google. Elle constitue donc un kit de développement logiciel qui permet de créer de nouvelles applications utilisant directement la base de données des pages indexées par Google, par le biais d'un service web. Fondé sur SOAP (Simple Object Access Protocol), la première API nous a été familière, ce qui nous a incité à l'utiliser pour développer le prototype JAVA de notre expérimentation.

3. Conception De Genquery

Le but du système est de donner une mise à jour de la requête utilisateur. La philosophie de base de l'interaction entre l'utilisateur et le système se fonde sur le fait que les utilisateurs sont incapables d'exprimer les critères exacts de leur satisfaction, et donc, le contenu prévu des documents de grand intérêt. C'est en raison de la nature instable des préférences humaines, les changements de l'information disponible et le modèle mental humain qui est fortement complexe et donc difficile à communiquer.

Notre approche vise la conception dans la partie client d'une interface "GenQuery" qui repose sur les services web du fameux moteur de recherche *Google.* En effet, c'est une interface qui prend en entrée la requête initiale formulée par l'utilisateur sans appuis explicite du système. Comme repense à cette requête, l'interface affiche les Dix premiers résultats. Chaque résultat est évalué dans une barre d'outil du navigateur Internet Explorer.

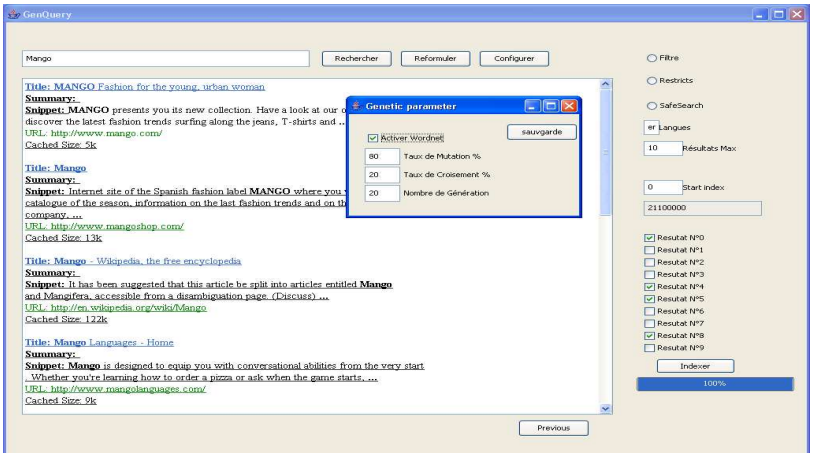


Figure 1. Interface Système Genquery.

Le rôle de l'interface est d'assurer l'interaction avec les utilisateurs, mais il assure d'autres fonctionnalités :

- Il Capte la requête de l'utilisateur communiquée au moteur de recherche, et qui exprime ces besoins d'information sur un sujet spécifique.
- Permettre à l'utilisateur d'accéder via une liste classée, aux documents qui satisferont le mieux cette requête. L'utilisateur pourra aussi évaluer la pertinence des documents retournés (feedback), que le système utilisera ensuite pour s'adapter à ces besoins.
- Communiquer la meilleure requête élue du processus évolutionnaire à l'utilisateur, et lui permettre de lancer une nouvelle recherche.
- En plus, la nouvelle requête enrichie le vocabulaire des utilisateurs.

La phase de prétraitement utilise les documents du Feedback sur les résultats de la requête initiale. On obtient ainsi les mots-clés d'expansion. De plus, cette phase exploite la relation d'hyponymie/hyperonymie des mots-clés de la requête initiale, fournit par *WordNet*, pour extraire les termes de reformulation (modifier la partie initiale de la requête),

Ensuite, Un agent génétique AG explore l'espace de recherche qui est l'ensemble des combinaisons possibles de mots-clés (chaque combinaison de mots-clés est une supposition de requêtes). Chaque requête (chromosome) est associé à une valeur de *fitness* qui représente l'évaluation donnée par le système a la base du processus de Feedback [21], qui récompense les mots-clés (gènes) des documents qui sont évalués comme bons (intéressants), tandis que ceux que l'utilisateur considère sans pertinence sont pénalisés. Le système propose après quelques générations la requête finale afin de la réinjecter à Google.

3.1. Prétraitement Du Feedback

La phase de prétraitement est responsable de la construction du "sac à mots". Elle est implémentée par deux modules :

1- Global agent : qui retient la requête initiale injectée par l'utilisateur, l'analyse, il interroge WordNet par la suite, pour extraire tous les hyponymes, et les hyperonymes de ses mots-clés.

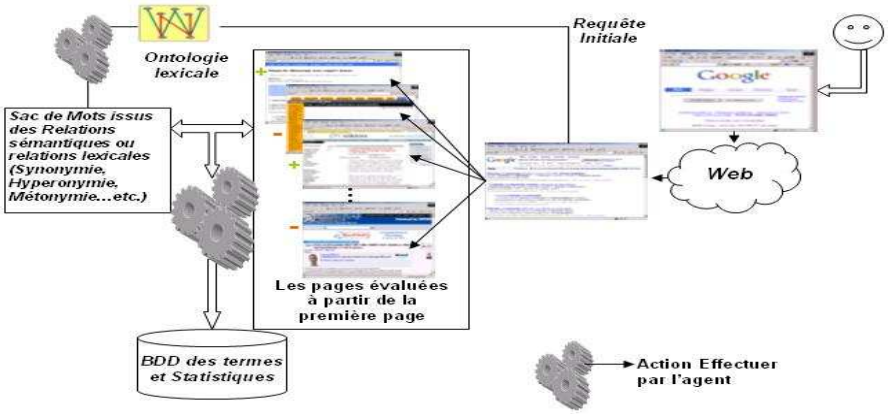


Figure 2. Génération des termes et création des statistiques sur les réactions de pertinence.

2- Feedback agent : Il contrôle le cache d'Internet Explorer. Crée un index des termes qui se trouvent dans les documents atteints depuis la page des résultats de recherche, ainsi que leurs fréquences. L'index est ensuite nettoyé en supprimant les *Stop Words* correspondant à la langue de recherche.

3.2. L'Algorithme génétique

L'implémentation du feedback par les Algorithmes génétiques est l'élément central de l'architecture du système, puisqu'il assure l'exploration de l'espace de recherche.

L'agent génétique utilisé implémente le modèle classique des Algorithmes génétique pris de la littérature. Les sections suivantes contiennent des détails d'implémentation.

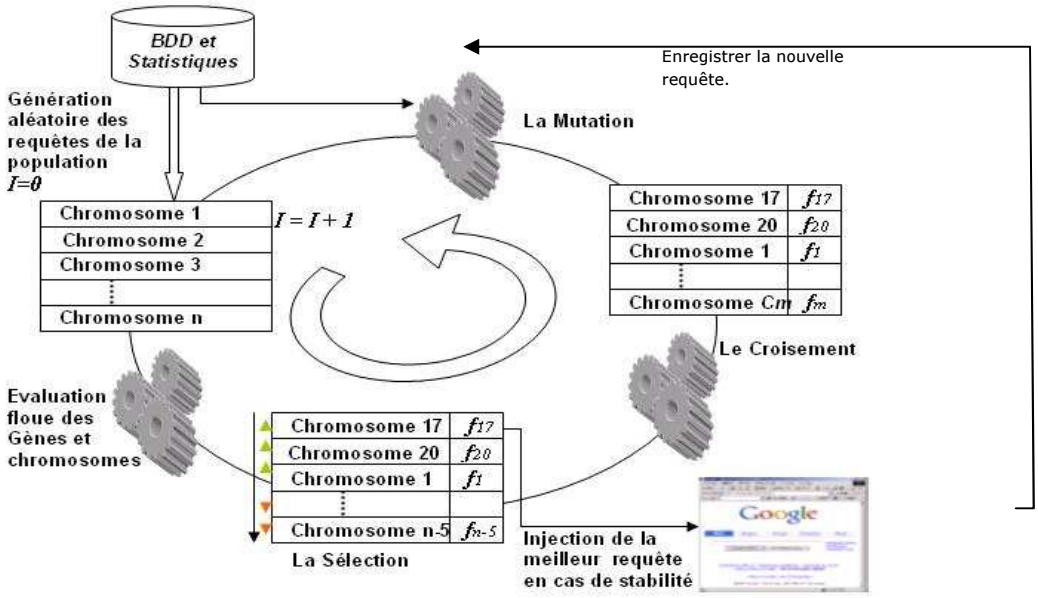


Figure 3. Cycle d'apprentissage.

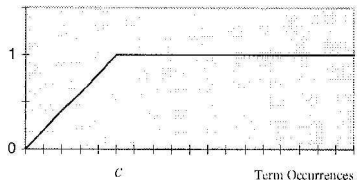
4. Les types de gènes et leurs évaluations

Le pré traitement effectué par le feedback agent et le global agent a pour résultat la création d'un index de mots-clés significatives. Cet indexe est complété par l'attribution de poids à chaque terme pour construire les gènes.

Un gène *G (t, g)* est le vecteur qui caractérise l'évaluation du terme *t* dans la requête. Les poids *g* des termes *t* sont assignés selon le type de gène.

4.1. Feedback gène

Il s'agit des mots-clés extraits des documents du feedback. Les termes des documents non pertinents pour l'utilisateur (hors sujet) son indexés avec le suffixe *NOT* qui désigne dans une requête Google que le mot-clé ne doit pas apparaître dans le résultat de recherche. Ainsi, tous les gènes du feedback seront évalués par la fonction suivante



$$\mu_{\geq}(x) = \begin{cases} \frac{x}{c} & x < c \\ 1 & x \geq c \end{cases}$$
$$\forall x \in Z_0^+$$

Figure 4. La fonction appliquée pour l'augmentation de rappel.

Les termes de la requête initiale peuvent ainsi être pénalisés s'ils n'occurrent pas assez dans les documents évalués pertinents, ce qui donne la possibilité de les remplacer par d'autre qui représentent mieux les besoins de l'utilisateur.

4.2. Wordnet gène

La relation exploitée dans notre expérimentation est l'hyperonymie et l'hyponymie (*is-a*), dans la taxonomie des Noms. La figure suivante représente un fragment de

```
tumor, tumour, neoplasm
      => growth
           => illness, malady, sickness
                => ill health, unhealthiness, health problem
                     => physiological state
                          => condition, status
                               => state
```

Wordnet ou la relation "is–a" est matérialiser par le symbole "=>"

Figure 5. Un fragment de la taxonomie des noms de Wordnet

Puisque ces mots-clés sont extraits de la requête initiale, leur évaluation primaire (sans prendre en compte s'ils appartiennent au Feedback document) est définis comme le plus cours chemin entre *a* et *b* [22] :

$$sim_{ab} = max\left[-log\left(\frac{N_p}{2D}\right)\right] \quad (1)$$

Où N_p est le nombre de liens dans le parcours *p* de *a* jusqu'a *b* et *D* est la profondeur maximale de la taxonomie. On normalise la valeur de cette similitude pour qu'elle soit comprise entre 0 et 1 (La mise à jour de poids d'un gène se fait par la moyenne de ces poids)..

4.3. La Population

La population de chromosomes est traitée par l'algorithme génétique. Chaque chromosome est une hypothèse sur la façon de reformuler la requête initiale afin de mieux représenter les besoins de l'utilisateur. L'évaluation du chromosome est appelée fitness. Le même gène est appliqué une unique fois dans le même chromosome, aussi bien que dans des chromosomes différents. La population initiale et générée depuis l'index (sac a mots) à l'aide d'un Random.

4.4. Les Opérateurs

Les opérateurs des Algorithmes génétiques sont *la sélection* qui est le choix des chromosomes de la population à reproduire. *Le croisement* qui prend une séquence de gènes de chacun de deux chromosomes parents choisis et les combine pour créer un nouveau chromosome en résultat, Et la *mutation* qui est le changement aléatoire d'un gène dans le chromosome choisi. Le croisement et la mutation sont nécessaires pour l'exploitation, l'exploration de l'espace de recherche [23].

a) **La Sélection :** On classe en premier temps les chromosomes dans l'ordre décroissent de leur fitness, et on choisi un chromosome dans une position aléatoire dans l'ensemble ordonné. Ensuite on sélectionne pour la nouvelle population un chromosome dans une position aléatoire entre la position du chromosome fittest (de fitness max) et la position du premier, jusqu'à atteindre le taux de survie généralement fixé a 80%. Selon cette stratégie de Sélection, nous obtenons que les chromosomes avec un haut fitness vont devenir plus probablement des parents que les chromosomes avec une fitness plus bas, comme dans la plupart de stratégies de Sélection [24].

b) **Le Croisement :** La stratégie de croisement choisis est one-point croisement. Le croisement est achevé comme suit : en commençant à la position de premier gène, jusqu'au premier point de croisement, on copie tous les gènes du premier parent à la même position dans le chromosome résultat [25].

c) **La Mutation :** On choisit aléatoirement un chromosome de la population, et une position de gène pour la mutation. Le gène *G (t, g)* dans la position choisie est remplacé par un autre gène *G (t', g')*, où *t'* est choisi aléatoirement parmi les termes d'un document évalué bon de l'ensemble de tous les termes à considérer dans la base de données, *g'* est l'évaluation du nouveaux gène *t'*.

d) **Paramètres De Contrôle :** Les valeurs des paramètres de contrôle sont respectivement la probabilité de croisement p_c =0,8, la probabilité de mutation p_m=0,2 et le nombre de générations fixé à 20 générations [26].

La probabilité p_c et p_m son considérablement grand pour éviter une convergence prématurée vers les extremums locaux. Le paramètre c de taux de pertinence est 25, tous ces paramètres sont ajustables. Il est aussi primordiale de fixé le nombre de gènes *n* par chromosome, car une requête courte est source d'ambiguïté, aussi bien qu'une longue requête qui ne retourne aucun résultat (Google n'accepte pas au delà de 20 par requête). *n* est fixé dans notre expérimentation a $n = 7$.

La fonction de Fitness (1) mesure l'adaptation de chaque chromosome de la population dans chaque itération du processus d'évolution. On suppose une population de taille *m*, formé par les chromosomes ou requêtes susceptibles à être injectés dans le moteur de recherche *Google*. Cette Fitness est calculée par la fonction suivante :

$$F(g_{j=1..m}) = \frac{1}{n}(\sum_{i=1..n} g_i), \tag{2}$$

5. Expérimentation et Evaluation

La forme normale conjonctive de la requête optimale est réinjecté dans GenQuery. Puisque *Google* accepte les requêtes booléennes, dix nouveaux résultats seront affichés aux utilisateurs.

L'utilisation de la métrique Rappel *R* ou la Précision *P* à part n'est pas suffisante. Donc il est préférable d'utiliser la F1 (**2**) mesure suivante :

$$F = \frac{2 \times P \times R}{P + R} \tag{3}$$

Cette mesure d'efficacité de chaque reformulation sera comparée avec celle de la première recherche, et a une reformulation manuelle.

On a prévu de tester le système avec un groupe d'utilisateur, ainsi il leur sera demandé d'utiliser notre système pour rechercher une collection de sujets proposées et d'autres sujets qui leur sont propres.

Mot de sujet	F1 mesure	Meilleur recommandation (après nettoyage manuel)	F1 mesure	Apport
Angle	0,37	angle parallel line point note	0,41	10%
Ashoka	0,28	ashoka Buddha land Indian	0,32	14%
Cobra	0,34	cobra snake venom yellow hood	0,47	38%
Eucalyptus	0,31	eucalyptus fruit tall gum white alcohol evergreen blue	0,51	65%
Graphite	0,33	graphite lead luster black soft maximum carbon	0,41	24%
Grasshopper	0,30	grasshopper adult soil stage brown	0,43	43%
Igloo	0,30	igloo snow air make shelter find eskimo	0,63	110%
Kangaroo	0,33	kangaroo tail pouch feet found user grass	0,35	6%
Kohinoor	0,26	Kohinoor diamond shah found	0,54	108%
Konark	0,28	Konark chariot Bhubaneswar sun	0,39	39%
Lave	0,33	lava ground earth volcano	0,49	48%
Leopard	0,35	leopard panther black coat long short east	0,67	91%
Lotus	0,35	lotus domain support list user search	0,61	74%
Mango	0,34	mango indica green tree	0,35	3%
Nil	0,33	nile Egypt river ancient	0,41	24%
Ostrich	0,32	ostrich bird larg brown ground	0,53	66%
Ozone	0,33	ozone ultraviolet earth Canada layer hole ground	0,61	85%

Tableau 1 : évaluation avec la métrique F1-mesure.

Pour confirmer le succès de l'algorithme, une enquête a été conduite pour tester la qualité des requêtes synthétisées. Chacun requête cible est une formulation automatique des requêtes initiales fournies par l'utilisateur. Nous avons constaté que les requêtes issues du système contiennent des mots-clés sans relation avec le besoin d'information, d'où la nécessité d'une étape de nettoyage manuel, ce qui rend le processus semi-automatique.

La précision de la requête initiale est calculée en fonction du nombre de résultats pertinents de première recherche par rapport à la totalité des documents retournés. Après l'acquisition du feedback utilisateur.

Pour chaque sujet de recherche, Le mot utiliser n'a pas de résultats par rapport à Wordnet. On remarque bien que Genquery a un très bon rapport précision/rappel, comparé avec une requête de l'utilisateur.

Dans les résultats (figure 6), on peut voir que les requêtes synthétisées ont été capables d'enrichir les efforts humains en ce qui concerne le vocabulaire désignant le BI. La valeur de précision et la mesure de F1, qui favorise une performance équilibrée de valeurs de couverture et précision.

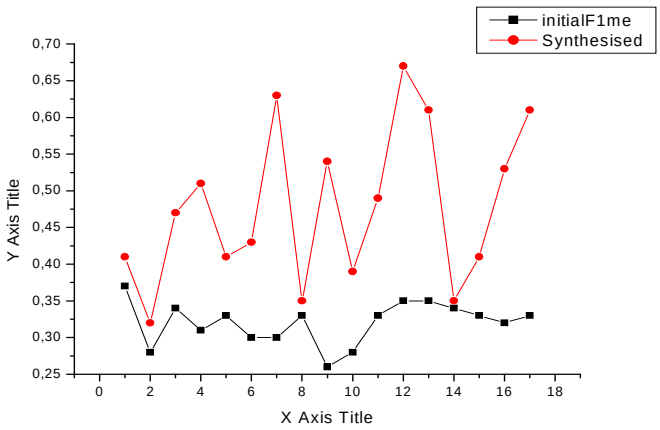


Figure 6 : Représentation graphique du tableau d'évaluation avec la métrique F1-mesure.

De façon générale, notre expérience qui utilise l'API de Google mise à disposition de tout développeur nous permet de tester directement de nouvelles applications en lien direct avec les indexes Google. Puis confirmer les caractéristiques fondamentales bien connues des chercheurs humains, pour proposer un système de recommandation qui a présente les propriétés suivantes :

- Les résultats trouvés restes loin de concurrencé les résultats de G. Salton[3] qui présente une amélioration de 140%.
- La Précision de la requête apparaît comme un des critères principaux utilisés pour avoir accès à la qualité de la requête. Elle est une métrique si intuitive que facile a mesurée.
- Le temps général d'exécution qui est la somme de temps de téléchargement

des pages plus le temps de traitement, pourra causer un inconvénient qui incite les utilisateurs a abandonné les méthodes basées sur les réinjections de pertinences.

- L'expérimentation a confirmé que les utilisateurs seront satisfaits par la première page des résultats si la précision relative à cette collection de 10 documents seulement est supérieure à 50%, c'est-à-dire cinq documents pertinents. Alors, ils ne chercheront pas faire une nouvelle itération.
- Mais, il est toutefois plus facile d'évaluer un processus de reformulation automatique de requêtes qu'un processus piloté par l'utilisateur, car on contrôle mieux l'environnement dans le premier cas que dans le second, où l'évaluation de la reformulation ne tiendrait pas seulement compte de l'algorithme, mais également des spécificités de l'utilisateur.

Conclusion

On a présenté dans ce papier une conception basée sur les techniques de relevance feedback par les algorithmes génétiques, tout en diminuant l'effort fournis par les utilisateurs et en augmentant aussi la précision. Les résultats expérimentaux prouvent que les utilisateurs ne donnent pas d'importance au rappel de leurs requêtes, par contre on considère la précision comme un facteur déterminant pour mesurer la qualité des requêtes du système. On plus, Les APIs Google nous ont été très utile pour faire les tests, mais il reste toujours des insuffisances concernant surtout dans la partie évaluation, qui serait plus subjective avec les compagnies de tests tels que TREC. Mais les évaluations prises en compte restent crédibles pour prouver l'efficacité de l'intégration de l'intelligence artificielle dans le contexte de recherche informationnel sur le web.

Références

[1] Rocchio, J. J. (1971). Relevance feedback in information retrieval. In Gerard Salton (ed.), *The SMART Retrieval System – Experiments in Automatic Document Processing*, pp. 313–323. Englewood Cliffs, NJ: Prentice-Hall. 139, 150, 244

[2] Ide, E. (1971). New experiments in relevance feedback. In Gerard Salton (ed.), *The SMART Retrieval System – Experiments in Automatic Document Processing*, pp. 337–354. Englewood Cliffs, NJ: Prentice-Hall. 150

[3] Claude de Loupy, (2002). Évaluation des taux de synonymie et de polysémie dans un texteTALN 2002, Nancy, 24-27juin.

[4] Singhal, Amit, Mandar Mitra, and Chris Buckley. 1997. Learning routing queries in a query zone. In *Proc. of SIGIR '97*, pp. 25–32. 150.

[5] Budzik, J. and K. Hammond. Watson. (1999). Anticipating and Contextualizing Information Needs. In American Society for Information Science..

[6] Zhang, B.T. and Y.-W. Seo. (2001). Personalized Web document Filtering Using Reinforcement Learning. In Applied Artificial Intelligence.

[7] Klink, S., (2002) Collaborative Learning of Term-Based Concepts for Automatic Query Expansion. In 13th European Conference on Machine Learning. 2002. Helsinki, Finland.

[8] Joachims, Thorsten. (2002). Optimizing search engines using clickthrough data. In *Proceedings of the ACM Conference on Knowledge Discovery in Data (KDD)*, pp. 133–142. 132, 150.

[9] Shen, X., B. Tan, and C. Zhai, (2005). Implicit User Modeling for Personalized Search. In 14th ACM international conference on Information and knowledge management.. Bremen, Germany: ACM Press

[10] Joachims, T., L. Granka, B. Pang, H. Hembrooke, and G. Gay. (2005). Accurately interpreting clickthrough data as implicit feedback. In *Proceedings of the Conference on Research and Development in Information Retrieval (SIGIR)*, pp. 154–161. 132, 150

[11] Shavlik J, Calcari S., Eliassi-Rad T., Solock J., (1999). An Instructable, Adaptive Interface for Discovering and Monitoring Information on the World-Wide Web, *Procs of the 1999 International Conference on Intelligent User Interfaces, ACM*, pp257-260.

[12] Pohl W., Nick A., (1999). Machine Learning and Knowledge-Based User Modeling in the LaboUr approach, *User Modeling: Proceedings of the 7th International Conference, UM99 , Edited by Judy Kay, Springer Wien New York*, pp. 179-188.

[13] Nick Z. Z., Themis P., (2001).Web Search Using a Genetic Algorithm, *IEEE Internet Computing, vol. 5, n° 2*, pp. 18-26, March-April

[14] Maria J. Martin-Bautista, Henrik Legind Larsen and Maria-Amparo Vila (1999), A Fuzzy Genetic Algorithm Approach to an Adaptive Information Retrieval Agent, JOURNAL OF THE AMERICAN SOCIETY FOR INFORMATION SCIENCE—July,760-771.

[15] Ph. Mylonas, D. Vallet, P. Castells, M. Fernandez,Y. Avrithis,(2007),Personalized information retrieval based on context and ontological knowledge.*The Knowledge Engineering Review*, Vol. 00:0, 1–24.

[16] http://www.google.com/

[17] Google Java API. Télécharger depuis http://www.google.com/apis ..

[18] Salton, G., & Buckley, C. (1990). Improving retrieval performance by relevance feedback. Journal of the American Society for Information Science, 41, 288–297.

[19] Resnik, P., 1995. Using information content to evaluate semantic similarity in a taxonomy. In: Mellish, C. Editor, , 1995. *Proceedings of the 14th International Joint Conference on Artificial Intelligence (IJCAI-95), Montreal, Canada* Morgan Kaufmann, San Francisco, CA, pp. 448–453 20–25 August.

[20] Goldberg, D.E. (1989). Genetic algorithms in search, optimization and machine learning. Addison-Wesley.

[21] Goldberg, D.E., & Deb, K. (1991). A comparative analysis of selection schemes used in genetic algorithms. In G.J.E. Rawlins (Ed.), Foundations of Genetic Algorithms (pp. 69–93). California: Morgan Kaufmann.

[22] DeJong, K.A., & Spears, W.M. (1992). A formal analysis of the role of multi-point crossover in genetic algorithms. Annals of Mathematics and Artificial Intelligence Journal, 5, 1–26.

[23] H. Chen, 1995.Machine learning for information retrieval: neural networks, symbolic learning, and genetic algorithms. *Journal of the American Society for Information Science* 46 3 (1995), pp. 194–216.