Table des matières

ORTOLANG[1] : une infrastructure de mutualisation de ressources linguistiques écrites et orales

Jean-Marie Pierrel[1, 2]
(1) Université de Lorraine, ATILF, 44 avenue de la Libération 54063 Nancy Cedex
(2) CNRS, ATILF, 44 avenue de la Libération 54063 Nancy Cedex
Jean-Marie.Pierrel@atilf.fr, contact@ortolang.fr

Résumé. Nous proposons une démonstration de la Plateforme de l'Equipex ORTOLANG (Open Resources and Tools for LANGuage : www.ortolang.fr) en cours de mise en place dans le cadre du programme d'investissements d'avenir (PIA) lancé par le gouvernement français. S'appuyant entre autres sur l'existant des centres de ressources CNRTL (Centre National de Ressources Textuelles et Lexicales : www.cnrtl.fr) et SLDR (Speech and Language Data Repository : http://sldr.org/), cette infrastructure a pour objectif d'assurer la gestion, la mutualisation, la diffusion et la pérennisation de ressources linguistiques de type corpus, dictionnaires, lexiques et outils de traitement de la langue, avec une focalisation particulière sur le français et les langues de France.

Mots-clés : Ortolang, plateforme, mutualisation, corpus, ressources linguistiques

1 Pourquoi une telle infrastructure ?

Une analyse de l'évolution des sciences du langage et du traitement automatique des langues montre que la confrontation avec l'informatique a permis de définir de nouvelles approches. Ainsi au-delà d'une simple linguistique descriptive s'est développée une *linguistique formelle* qui propose des modèles s'appuyant sur une double validation, *explicative* d'un point de vue linguistique, *opératoire* d'un point de vue informatique. Une véritable *linguistique de corpus* permet aussi au linguiste d'aller au-delà de l'accumulation de faits de langue et de confronter ses théories à l'usage effectif de la langue. Ainsi l'informatique est devenue un outil indispensable pour :

- étudier la langue et ses propriétés grâce à l'exploitation de corpus de grande ampleur ;
- structurer et normaliser les connaissances linguistiques (de l'acoustique, à la sémantique) ;
- valoriser et partager les résultats de la recherche grâce à la production de ressources et d'outils informatiques.

Dans ce cadre, les aspects de ressources informatisées (corpus annotés, lexiques et outils de traitement) sont particulièrement importants et stratégiques pour servir de support à la fois :

- aux travaux de recherche pour lesquels la notion de corpus d'étude et de ressources est incontournable ;
- à la diffusion des résultats de ces travaux grâce à leur disponibilité sur la toile.

Un équipement d'excellence de mutualisation de ressources et d'outils pour le traitement informatisé et la valorisation de notre langue s'impose aujourd'hui pour les raisons suivantes :

- Le coût de définition et de production de ressources linguistiques de qualité ou d'outils d'analyse est important. Sans une mutualisation de telles ressources, chaque chercheur se verrait dans l'obligation de tout réinventer !
- L'évaluation de nos productions de recherche (modèles, systèmes de traitement) nécessite la disponibilité de ressources de référence (corpus, lexiques, dictionnaires) accessibles, partagées et clairement identifiables.
- Le partage et la patrimonialisation des connaissances sur les langues de France sont nécessaires afin de faciliter des études sociolinguistiques sur les parlers de France et de les faire bénéficier des apports de la recherche.

2 Principales caractéristiques d'ORTOLANG

Le consortium portant le projet ORTOLANG regroupe des compétences complémentaires en

- sciences du langage à travers l'ATILF, le LPL, MoDyCo et le LLL,
- informatique avec le LORIA et l'INIST, mais aussi en partie l'ATILF et le LPL,
- base de données et accès à de l'information scientifique, à travers l'INIST, et à des ressources linguistiques, à travers les deux centres de ressources que sont le CNRTL et le SLDR.

[1] ORTOLANG bénéficie d'une aide de l'Etat gérée par l'Agence Nationale de la Recherche au titre du programme « Investissements d'avenir » portant la référence ANR-11-EQPX-0032

Au-delà de la réunion de ces compétences disciplinaires différentes, notre objectif fut aussi de fédérer pour cet équipement de mutualisation de ressources et d'outils sur la langue des partenaires représentant la diversité des approches d'étude de la langue : modélisation linguistique, linguistique expérimentale et/ou appliquée, production et perception du langage, études diachroniques, sociolinguistiques, traitement automatique des langues, écrit, oral.

Cette proposition s'appuie aussi sur une expérience acquise des équipes proposant cet équipement d'excellence et sur une bonne insertion tant nationale qu'internationale :

- acquis des partenaires, centres de ressources et laboratoires, qui alimentent la version initiale de la plateforme avec un ensemble de ressources et d'outils déjà disponibles en leur sein et dont les compétences recouvrent les trois principaux aspects visés : l'oral, l'écrit et la patrimonialisation des parlers de France.
- implication et cohérence avec la TGIR HumaNum.
- implication et cohérence avec l'infrastructure européenne CLARIN.
- cohérence avec les efforts de la DGLFLF et de la BNF sur les aspects patrimonialisation des parlers de France.

La plateforme ORTOLANG est une infrastructure de mutualisation pour la gestion, la pérennisation et la diffusion de corpus et d'outils sur la langue, ces derniers restant bien entendu propriété des déposants (chercheurs ou laboratoires). Nous avons, de plus, prévu des moyens pour aider des laboratoires à finaliser et normaliser leurs ressources.

Quant aux droits d'accès à ces ressources, ils restent donc définis par leurs propriétaires. Toutefois sur ce point ORTOLANG émet des recommandations fortes :

- respect de la charte éthique Big Data, fruit d'un travail collectif réunissant plusieurs acteurs impliqués dans la création, la diffusion et l'utilisation de données,
- liberté d'usage pour la recherche et tant qu'il n'y a pas de valorisation contractuelle,
- moyennant royalties auprès des propriétaires des ressources dès qu'il y a valorisation contractuelle.

C'est dans cet esprit que divers contacts avec des partenaires ayant déposé ou souhaitant déposer leurs ressources sur ORTOLANG ont déjà été mis en œuvre.

3 Objectifs et missions de cette infrastructure

3.1 Identification et préparation des données

Une des difficultés actuelles pour repérer et accéder à des ressources (corpus, dictionnaires, lexiques et outils de traitement) sur notre langue réside dans leur grande dispersion (il n'est pas aisé de savoir quelles ressources sont disponibles et à quels endroits elles sont accessibles) et leur forte disparité, en particulier en termes de codage. Sans compter que nombre de ressources langagières de qualité, développées dans le cadre de projets de recherche ou de thèses, ont été perdues faute d'une gestion rigoureuse de ce patrimoine. C'est pourquoi l'un des premiers objectifs concerne : le catalogage des ressources et outils existants à travers un ensemble de métadonnées normalisées, le contrôle et la validation des ressources et des outils, avec en particulier un accompagnement de leurs auteurs sur les standards, les normes et les recommandations internationales actuelles, et l'enrichissement de ressources et d'outils.

3.2 Pérennisation des ressources

Afin d'assurer la pérennisation des ressources, nous avons mis en œuvre trois types d'actions : la curation des ressources et des outils ; un stockage sécurisé et une maintenance des ressources ; un archivage pérenne, à travers la solution mise en place par la TGIR HumaNum en lien avec le CINES.

3.3 Diffusion

Enfin, pour assurer la nécessaire diffusion et exploitation de ces ressources nous prévoyons une aide et un accompagnement des utilisateurs pour la mise en place des procédures permettant à des utilisateurs de la plateforme d'exploiter ces ressources et outils mutualisés en nous appuyant sur l'expérience des équipes porteuses de l'Equipex et centres de ressources CNRTL et SLDR appelés à terme à se fondre au sein d'ORTOLANG.

4 Démonstration

Au cours de cette démonstration, après une présentation de l'architecture matérielle et logicielle mise en place pour ORTOLANG, nous présenterons la plateforme accessible aujourd'hui dans sa version 0 à l'adresse www.ortolang.fr, ses développements futurs ainsi que les modes d'interactions et de coopérations que nous souhaitons mettre en place avec les producteurs de ressources et d'outils ainsi qu'avec l'ensemble de la communauté utilisatrice potentielle de cette infrastructure.

Utilisabilité d'une ressource propriétaire riche dans le cadre de la classification de documents

Résumé. Dans ce papier, nous nous intéressons à l'utilisation d'une ressource linguistique propriétaire riche pour une tâche de classification. L'objectif est ici de mesurer l'impact de l'ajout de ces ressources sur cette tâche en termes de performances. Nous montrons que l'utilisation de cette ressource en temps que traits supplémentaires de classification apporte un réel avantage pour un ajout très modéré en termes de nombre de traits.

Abstract. In this paper, we focus on the use of a proprietary resource for a document classification task. The objective is here to measure the impact of the addition of this resource as input for classification features. We show that the use of this resource impacts positively the classification results, for a limited impact on the feature number.

Mots-clés : classification de documents, classification automatique, ressources
Keywords: document level classification, automatic classification, resources

Introduction

1.1 Problématique

La classification de documents selon les thématiques protées par ceux-ci est un problème qui a été très largement étudié par la communauté scientifique. Les approches par classification automatique se sont révélées au cours des dernières années une solution tout à fait adaptée, à la fois en termes de performances et de rapidité/coût de mise en place.

Néanmoins, l'optimisation de ces résultats est toujours possible pour une problématique donnée, notamment via une sélection pertinente des attributs utilisés pour la classification, ainsi que par l'utilisation de primitives linguistiques fines et fiables.

Dans ce papier, nous nous intéressons à comment ces résultats peuvent être optimisés à partir de ressources linguistiques propriétaires génériques sémantiquement riche. L'objectif est ici de mesurer l'impact de l'ajout de ces ressources sur une tâche donnée de classification en termes de performances.

1.2 Travaux existants

La classification automatique thématique de documents est un domaine qui a été très largement étudié. Les travaux sur le sujet peuvent être divisés entre deux approches : la classification sur un jeu de catégorie connu a priori – citons par exemple les travaux de (Chai et al., 2002), et la classification sans jeu de classes, par regroupement de textes proches ou clustering – citons par exemple (Manning et Shütze, 1999), (Pappuswamy et al., 2005).

Notre approche se situe dans la première catégorie, plus simple, mais correspondant à un certain nombre de cas d'application concret (par exemple, la labellisation automatique d'articles de presse suivant une taxinomie d'articles prédéfinie. Notre objectif pour ce travail n'est pas tant d'obtenir un classifieur parfaitement optimisé – les résultats obtenus peuvent très certainement être amélioré – mais de mesurer l'impact de ressources existantes et la progression des résultats avec l'ajout de traits issus de ces ressources.

2 Notre approche

Nous avons choisi de tester l'adaptation des ressources linguistiques propriétaires sur une tâche de classification d'articles de presse. Nous décrivons en section 2.1 la méthodologie que nous avons employée. La section 2.2 décrit en détails les outils et ressources utilisées.

2.1 Méthodologie

Notre approche se déroule en deux phases. Tout d'abord, nous testons, sur un corpus de développement, plusieurs algorithmes de classification automatique. Cette phase nous permet de sélectionner le classifieur le plus adapté a priori pour la tâche, et d'adapter son paramétrage. Cette phase s'effectue sur un corpus plus restreint avec un jeu de traits donnés.

Dans un second temps, nous testons l'ensemble des combinaisons de traits à notre disposition pour le classifieur choisi. Cette approche en deux phases nous permet de limiter la combinatoire des tests effectués. Ceci est d'autant plus important que certains algorithmes testés en phase 1 sont particulièrement longs pour l'apprentissage.

2.2 Ressources linguistiques et outils de classification

Les ressources linguistiques utilisées sont celles qui sont commercialisées par Synapse Développement. Parmi les outils mis à disposition par cette société, nous avons à notre disposition deux outils : l'analyseur syntaxique de Synapse Développement, et l'extracteur de mots-clés, concepts-clés, et noms propres clés de Synapse Développement. Ce dernier est basé sur la taxinomie des concepts générique propriétaire de Synapse Développement.

L'analyseur syntaxique de Synapse Développement a été reconnu comme l'un des meilleurs pour le Français, notamment via plusieurs évaluations (Laurent et al. 2009). Nous utilisons pour notre tâche la version anglaise de l'analyseur, dont les performances sont au niveau de l'état de l'art.

La taxinomie des concepts de Synapse Développement est une ressource lexicale générique qui associe à chacun des termes et syntagme reconnus par l'analyseur syntaxique une ou plusieurs catégories, et ce suivant le sens reconnu si le terme est polysémique. Cette taxinomie se rapproche des synsets de WordNet, à la différence qu'ici le nombre de catégories conceptuelles est borné : la taxinomie est un arbre comptant quatre niveaux, avec 3387 catégories feuilles de rang 4, pour 256 catégories de rang 3. La figure ci-dessous montre un exemple en français de catégorisation pour l'adjectif polysémique « cher ».

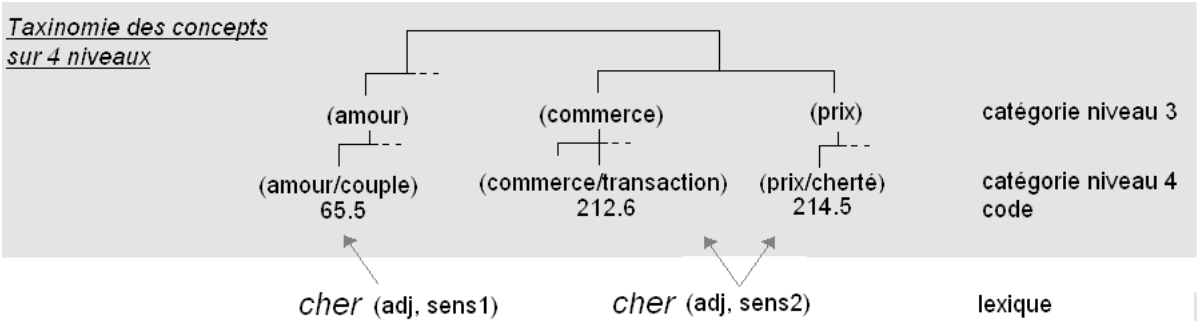


Figure 1 : exemple de catégorisation pour les deux sens de l'adjectif cher

La taxinomie de Synapse Développement étant disponible en multi-lingue (mêmes catégories), nous nous sommes appuyé pour les expérimentations la version anglaise de la ressource. L'extracteur de mots clés, concepts clés et noms propres clés est un composant s'appuyant sur cette taxinomie. À partir d'un texte, le composant extrait les mots clés, les noms propres clés, et les concepts de niveau 3 et 4 associés au document, avec un score de confiance entre 0 et 1 donnant l'importance du terme ou concept clé dans le document.

Enfin, nous avons utilisé pour les expérimentations les implémentations de la plate-forme Weka (Hall et al., 2009) pour la classification automatique.

3 Expérimentations

3.1 Données

Le corpus que nous avons utilisé afin de tester les technologies de Synapse et ce qu'elles peuvent apporter à la classification est une sous partie des nouvelles de l'agence de presse anglaise REUTERS[1] datées entre le 20-08-1996 et le 19-08-1997. Ces nouvelles sont déjà classée par l'agence en 125 catégories dont la plupart peuvent être apparentées à des sous-catégories de la classification IPTC17[2].

Ces documents sont classés en une classe principale et éventuellement plusieurs classes secondaires suivant le ou les thèmes abordés. La tâche que nous avons choisie est la reconnaissance de la classe principale du document. Afin de limiter les problèmes d'affectation d'un document à une classe secondaire, nous nous sommes limité pour la constitution des instances de test aux document ne comportant qu'une seule classe. Parmi les 125 catégories, nous avons choisi de ne garder que les 15 classes les plus occurrentes dans le corpus. Ceci nous permet notamment de disposer de suffisamment d'instances de documents ne possédant qu'une seule classe.

Le corpus de développement est constitué de 660 documents parmi l'ensemble des documents du corpus REUTERS complet. Ce corpus est utilisé dans la phase de sélection et paramétrage du classifieur. Pour la phase d'évaluation des traits le corpus d'entraînement de 2340 documents, et le corpus de test est constitué de 390 documents. Les documents ont été sélectionnés au hasard quant à leur contenu précis, mais de manière à équilibrer les différentes classes présentes.

L'ensemble des documents est rédigé dans un anglais impeccable, ce qui facilite l'analyse syntaxique du texte. Dans le cas contraire, une passe de correction orthographique et grammaticale automatique aurait pu être envisagé en pré-traitement des fichiers.

3.2 Choix et paramétrage du classifieur

Choix du classifieur

Nous avons effectué une première phase de classification sur le corpus de développement dans le but de sélectionner un algorithme adapté à notre tâche. Les algorithmes testés font partie des implémentations classiques de la plate-forme Weka. La version utilisée de l'outil est la version 3.6 (version stable en cours). Les résultats obtenus sont décrits dans le tableau 1 suivant.

[1] http://fr.reuters.com/

[2] http://www.iptc.org/site/Home/

Classifieur	Accuracy
Random Forest	**68.55 %**
J48	52.62 %
IBK	25.60 %
Bagging	46.77 %
Naive Bayes	37.50 %
K-Means	11.90 %
RepTree	53.43 %

Tableau 1 : accuracy sur le corpus de développement (toutes classes) pour chaque algorithme testé

Nous pouvons observé une nette prédominance de l'algorithme Random Forest (Breiman, 2001). C'est donc sur cet algorithme que s'est porté notre choix pour la suite des expérimentations.

Paramétrage du classifieur

L'algorithme Random Forest est basé sur la génération de k arbres se basant sur p traits aléatoires. Une valeur de p est proposée par l'implémentation de l'algorithme que nous avons utilisée. Cette valeur est calculée en fonction du nombre total de traits du corpus.

Nous avons évalué sur ce même corpus de développement les variations de performances de l'algorithme en fonction du nombre d'arbres. En faisant varié entre 10 et 800 le nombre d'arbres, nous avons observé que k=50 arbres était un bon compromis entre rapidité de calcul du modèle et performance, avec une accuracy quasi-maximale. Le maximum global observé pour 775 arbres n'est que très marginalement plus performant pour un temps de calcul très important.

3.3 Comparatifs des différents jeux de traits

Nous nous sommes ensuite basé sur les corpus d'entrainement et de test (décrits en section 3.1) pour effectuer une série de classification, dans le but d'évaluer l'impact de chaque trait sur les performances du classifieur.

Les traits utilisés pour les expérimentations sont les suivants :

- Mots (W) : les mots "bruts" qui composent le document. À chaque mot du corpus correspond un trait, actif s'il est effectivement présent dans le document.

- Lemmes (L) : la forme lemmatisée de ces mots. Comme pour les mots, à chaque lemme distinct présent dans le corpus correspond un trait, actif s'il est effectivement présent dans le document.

- Concepts-clés (CC) : les concepts clés associés au document. Ceux-ci sont extraits par le composant d'extraction des mots, concepts, et noms propres clés décrit en section 2.2. À chaque concept associé à un document est associé un trait booléen. Nous avons utilisé les concepts de niveau 3 et 4 pour la classification, et filtré les concepts clés suivant le niveau de confiance donné par le composant, avec une valeur seuil de 0.5.

- Mots-clés (MC) : les mots-clés associés au document. Ceux-ci sont extraits par le composant d'extraction des mots, concepts, et noms propres clés décrit en section 2.2. À chaque mot-clé d'un document est associé un trait booléen. Comme pour les concepts, nous avons filtré les mots clés suivant le niveau de confiance donné par le composant, avec une valeur seuil de 0.5.

- Noms Propres Clés (NPC) : les noms propres clés associés au document. Ceux-ci sont également extraits par le composant d'extraction décrit en section 2.2. À chaque nom propre clé d'un document est associé un trait booléen. Comme précédemment, nous avons filtré les noms propres clés suivant le niveau de confiance donné par le composant, avec une valeur seuil de 0.5.

Le tableau 2 suivant présente les résultats obtenus en termes de documents correctement classifiés :

Traits utilisés	**Accuracy obtenue (en %)**
W	56 %
L	60 %
W + MC	63.33 %
L + MC	64.66 %
W + MC + CC	71 %
L + MC + CC	**73 %**
W + MC + CC + NPC	69.66 %
L + MC + CC + NPC	**73 %**

Tableau 2 : accuracy sur le corpus de test (toutes classes) pour chaque jeu de traits testé

3.4 Discussion

Nous pouvons retirer plusieurs observations de ces résultats. Quantitativement, les résultats observés ne sont pas excellents quelque soit la méthode employée. Ceci vient très certainement d'une sélection un peu trop restreinte de textes d'entrainement et de test. L'objectif de ce papier n'est néamoins pas tant d'optimiser les résultats bruts que d'observer l'impact de l'ajout des traits de mots clés, concepts clés, et noms propres clés.

Le classifieur le plus performant sur la tâche présente un intérêt certain dans un contexte industriel. En effet, celui-ci présente l'avantage d'être assez performant en termes de vitesse d'apprentissage et de classification, ce qui réduit les coûts d'adaptation à une problématique client. De plus, le modèle généré étant basé sur un arbre de décision, il est tout à fait envisageable de compléter manuellement cet arbre à tout niveau avec des règles symboliques métiers spécifiques à un besoin client, et ce sans ré-entrainement du classifieur.

Côté traits de classification, la lemmatisation apporte globalement un plus important par rapport aux mots bruts. C'était attendu compte tenu de l'état de l'art, mais cela valide l'intérêt et la qualité du composant de Synapse dans le cadre de la classification.

L'ajout des mots clés apporte un gain de performance notable. Ces éléments sont donc intéressants à retenir pour la classification. Les noms propres clés n'apportent quasi aucune amélioration sur notre corpus. Ceci était également plutôt attendu en raison de la taille des corpus utilisés : pour pouvoir tirer parti de la redondance de la mention d'une même entité nommée entre deux textes de catégories similaires, le nombre d'exemples utilisés pour l'entrainement était sans doute trop restreint. Les concepts-clés apportent quant à eux un réel gain en termes de qualité de classification. On remarque également que même seul, ce trait est tout à fait discriminant pour le cas discuté. Ceci valide l'utilisabilité de cette pré-classification générique des documents pour aller vers une classification métier donnée.

Il est également intéressant de noter que le nombre de traits introduits par ces ressources est très réduit par rapport à des traits lexicaux classiques (ici mots ou lemmes, mais également n-grammes), ce qui permet de limiter l'impact sur le temps de calcul du modèle de classification. Ceci est particulièrement important dans le cadre d'une application industrielle.

4 Conclusion et Perspectives

Nous avons testé dans ce papier l'utilisabilité d'une ressource linguistique propriétaire en tant que source de traits de classification. Nous avons pu voir que l'ajout de concepts et mots clés issus du composant d'extraction de Synapse Développement permettait d'améliorer de manière significative les résultats obtenus.

Les résultats en eux-mêmes peuvent être optimisés en complétant le jeu de traits utilisé avec des traits complémentaires, notamment issus de l'analyse syntaxique : nous n'avons utilisé pour ces expérimentations que les lemmes, laissant de côté les groupes syntaxiques ("chunks") et les informations de désambiguïsation sémantique. Des expérimentations concernant ces traits sont en cours avec des résultats prometteurs.

Remerciements

[Paragraphe temporairement supprimé par les auteurs pour anonymisation]

Références (style *Titre sans numéro*)

BREIMAN L. (2001). Random Forests. *Machine Learning*. 45(1) :5-32.

CHAI KMA., CHIEU HL., Ng HT. (2002) Bayesian online classifiers for text classification and filtering. *Proceedings of the 25th annual international ACM SIGIR conference on Research and development in information retrieval* p. 97‑104.

HALL M, FRANK E, HOLMES G, PFAHRINGER B, REUTEMANN P, WITTEN IH. (2009) The WEKA data mining software: an update. *ACM SIGKDD explorations newsletter*. 11(1):10 8.

LAURENT D., NEGRE S., SEGUELA P. (2009) L'analyseur syntaxique cordial dans Passage. *Actes de TALN 2009*.

MANNING, C. AND SCHÜTZE, H. (1999). *Foundations of Statistical Natural Language Processing*. The MIT Press, Cambridge, US.

PAPPUSWAMY U, BHEMBE D, JORDAN PW, VANLEHN K. (2005) A supervised clustering method for text classification. *Computational Linguistics and Intelligent Text Processing*. Springer p. 704‑14.

CFAsT: Content-Finder AssistanT

Romain Laroche
Orange Labs, 38-40 avenue du Général Leclerc, 92130 Issy-les-Moulineaux / France
romain.laroche@orange.com

Résumé. Cette démonstration de CFAsT s'intéresse à "comment concevoir un système de dialogue avec un effort minimal". Cet assistant virtuel repose sur un nouveau modèle pour la génération automatique de système de dialogue construite à partir de contenus. Cette approche utilise un moteur de recherche auquel on a ajouté des fonctionnalités de dialogue : à chaque tour, le système propose trois mots-clefs de manière à optimiser l'espérance de gain d'information.

Abstract. This CFAsT demonstration focuses on "how to design and develop a dialogue system with a minimal effort". This virtual assistant embeds a novel model for automatic generation of dialogue systems built from contents. This approach is similar to and relies on a search engine, but with augmented dialogue capabilities : at each dialogue turn, the system propose three keywords, in order to optimise the information gain expectation.

Mots-clés : Systèmes de dialogue, Traitement automatique des langues naturelles, Assistant virtuel.

Keywords: Dialogue systems, Natural language processing, Virtual assistant.

Approche

Souvent, la recherche sur les systèmes de dialogue s'intéresse à "comment concevoir un système de dialogue avec un effort minimal". C'est le cas notamment des approches dirigées par les données (Levin *et al.*, 1998), basées sur de l'apprentissage par renforcement à partir des dialogues de la machine avec des utilisateurs (réels ou simulés). Ces dernières années, ces approches ont connu une grande quantité d'améliorations, aussi bien au niveau des techniques d'apprentissage (Pietquin *et al.*, 2011) que de la méthodologie (El Asri *et al.*, 2012). Et pourtant, l'apprentissage par renforcement n'a pu être utilisé dans les systèmes de dialogue industriels, que lorsque l'on pouvait le projeter dans un système basé sur des automates (Putois *et al.*, 2010), parce que la plupart des réserves exprimées (Paek & Pieraccini, 2008) sont toujours d'actualité.

Notre approche contourne ces réserves en basant son modèle sur les contenus, plutôt que sur les interactions avec l'utilisateur, pour générer un système de dialogue. Le Content Finder AssistanT (CFAsT) est un outil permettant l'implantation automatique de système de dialogue très simples. Comme l'approche CFAsT se base sur les contenus, les applications générées peuvent être comparées aux moteurs de recherche, mais la méthodologie n'est pas le même : le moteur de recherche cherche les meilleurs contenus étant donnée une requête utilisateur, tandis que le CFAsT aide l'utilisateur à formuler sa requête pour que les réponses soient les plus pertinentes possibles.

Le flux de dialogue est illustré par la figure 1. L'utilisateur entre dans une boucle de dialogue dans laquelle l'historique de dialogue est la concaténation de toutes ses requêtes, augmentée de l'ensemble des contenus qu'il a déjà visionnés (et qui ne l'ont donc pas satisfait). A chaque tour de dialogue, le CFAsT utilise un moteur de recherche standard pour proposer les cinq contenus les plus pertinents et donner des statistiques (nombre de contenus répondant à la requête). Mais il utilise également un algorithme de calcul d'entropie pour évaluer l'espérance de gain d'information suite aux possibles actions de l'utilisateur quand le système lui proposera un ensemble de trois mots-clefs. De cette manière le système choisira les mots-clefs maximisant cette espérance de gain d'information. Si l'utilisateur sélectionne un contenu, il est amené à dire s'il répond à sa requête.

Placer l'utilisateur dans le cadre d'un dialogue permet de recueillir ses retours quant à la pertinence des contenus qu'on lui propose. Cette information est ensuite utilisée en ligne pour optimiser les futures recommandations. Par ailleurs, l'application générée automatiquement peut être surchargée par des branches de dialogue développées manuellement, des stratégies de désambiguïsation, des informations lexicales (synonymie, mots outils) ou encore des comportements standards face aux questions hors périmètre.

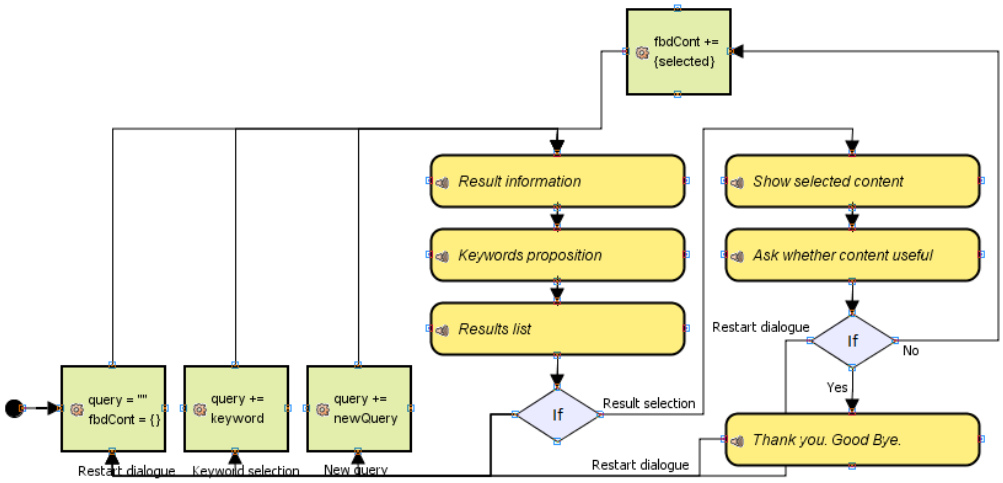


FIGURE 1 – Flux de dialogue dans le CFAsT.

Discussion

A ce jour, une dizaine d'applications ont été générées grâce à cet outil au stade de preuves de concept, en français et en anglais, sur des corpus plus ou moins volumineux (de 160 à 7k contenus et de 818 à 30k termes), et sur des données plus ou moins structurées (pages web, articles de proceedings, FAQ, documents de bureautique, mails, vidéos, fiches de restaurant, ...). Les applications à faible coût ainsi obtenues sont générées après moins d'une heure de travail.

Depuis un moteur de recherche amélioré, surchargé de quelques stratégies de dialogue, jusqu'à un assistant conversationnel utilisant un moteur de recherche comme parachute en cas d'incompréhension de l'entrée utilisateur, tout le spectre est possible. Contrairement aux systèmes de dialogue utilisant de l'apprentissage par renforcement, les outils du CFAsT ne prétendent pas générer des applications optimales, ni même plus performantes que celles développées manuellement, mais simplement de le faire à coût faible voire nul. Il est ensuite à la charge du développeur d'enrichir, ou non, le dialogue avec des stratégies manuelles suivant le besoin.

L'objectif de ce démonstrateur est de réaliser qu'il est possible de générer une application de dialogue simple automatiquement. Nos travaux futurs sur ce sujet porteront sur la saisie de requêtes multi-champs avec une topologie inter-champs optimisée grâce à un système de recommandation (Meyer & Fessant, 2011) et à l'expérience, et sur la proposition automatique de stratégies de dialogue au concepteur humain.

Disserto, la suite logicielle à l'aide de laquelle est développé le CFAsT est avant tout utilisée pour la conception de systèmes vocaux, et notre intention est d'étendre le modèle à du dialogue parlé. Il reste de sérieux obstacles pour atteindre cet objectif : à l'oral le système devra notamment trouver une certaine forme de concision.

Références

EL ASRI L., LAROCHE R. & PIETQUIN O. (2012). Reward function learning for dialogue management. In *Proceedings of STAIRS*, p. 95–106.

LEVIN E., PIERACCINI R. & ECKERT W. (1998). Using markov decision process for learning dialogue strategies. In *Proceedings of ICASSP*.

MEYER F. & FESSANT F. (2011). Reperio : A generic and flexible industrial recommender system. In *Proceedings of IEEE/WIC/ACM WI-IAT*, p. 502–505 : IEEE Computer Society.

PAEK T. & PIERACCINI R. (2008). Automating spoken dialogue management design using machine learning : An industry perspective. *Speech Communication*, **50**, 716–729.

PIETQUIN O., GEIST M. & CHANDRAMOHAN S. (2011). Sample efficient on-line learning of optimal dialogue policies with kalman temporal differences. In *Proceedings of IJCAI*, p. 1878–1883 : AAAI Press.

PUTOIS G., LAROCHE R. & BRETIER P. (2010). Online reinforcement learning for spoken dialogue systems : The story of a commercial deployment success. In *Proceedings of SIGDIAL*, Tokyo (Japan).

Démonstration de *Kawâkib*, outil permettant d'assurer le feedback entre grammaire et corpus arabe pour l'élaboration d'un modèle théorique

André Jaccarini[1], Christian Gaubert[2]
(1) MMSH, CNRS, 5 rue du Château de l'Horloge, 13094 Aix-en-Provence
(2) IFAO, 37 rue Cheikh Aly Yousef, Qasr al Ayni, Le Caire, Egypte
jaccarini@mmsh.univ-aix.fr, cgaubert@ifao.egnet.net

Résumé. Kawâkib est un outil assurant le feedback entre corpus arabe et grammaire. Ce logiciel interactif en ligne démontre le bien fondé de la méthode de variation des grammaires arabes pour l'obtention de l'algorithme optimal tant au niveau de l'analyse morphologique, cruciale étant donnée la structure du système sémitique, que syntaxique ou dans le domaine de la recherche de critères pertinents et discriminants pour le filtrage des textes.

Abstract. Kawâkib is a tool allowing feedback between arabic corpus and grammar. As far as methodology is concerned, this interactive online software implements and illustrates the grammar variation method that aims to determine the optimal algorithm, either for morphology – which is essential in semitic languages - or for syntax. The software also permits the search for criteria for text filtering.

Mots-clés : arabe, automates, analyseurs, opérateurs linguistiques, mots-outils, filtrage de corpus.
Keywords: arabic, automata, parsers, lingistic operators, tool words, corpus filtering.

Kawâkib est un outil assurant le feedback entre corpus arabe et grammaire. Sur le plan méthodologique, ce logiciel interactif en ligne démontre le bien fondé de la méthode de variation des grammaires arabes en vue de l'obtention de l'algorithme optimal tant au niveau de l'analyse morphologique, cruciale étant donnée la structure du système sémitique, que syntaxique ou dans le domaine de la recherche de critères pertinents et discriminants pour le filtrage des textes. Kawâkib contient :

- 1 - une bibliothèque d'automates exprimant des règles morphosyntaxiques et des opérateurs de détection des relations discursives
- 2 - des fonctions de "radiographie linguisitique" attribuant des valeurs numériques à des textes à tout venant en vue du filtrage

Ce logiciel doit permettre au chercheur de spécifier lui-même sa grammaire ou de modifier les grands schèmes d'automates proposés par le système, de les instancier, de les mettre en œuvre pour les modifier ensuite rétroactivement en fonction des résultats.

Cette démonstration a pour objectif de montrer :

- a - la richesse des ressources linguistiques
- b - le souplesse de leur utilisation
- c - la fécondité du feedback entre modèle théorique et implémentation

Les fonctionnalités linguistiques immédiatement accessibles à l'utilisateur sont les suivantes :

- Analyses morphologiques des noms et verbes (trilitères et quadrilitères) par automates à états finis et transducteurs, avec ressources linguistiques minimales
- Mise en évidence des ambiguïtés et de leur hiérarchie
- Analyse/recherche des opérateurs tokens (associés aux mots-outils) organisés en 24 catégories
- Statistiques pour ces tokens et leurs catégories
- Repérage et stockage de combinaisons et séries de tokens
- Recherche de mots dans un texte et analyse du contexte
- Recherche de racines ou motif de racines (R1 X R3 par exemple)
- Recherche des racines les plus fréquentes et calcul du seuil de couverture
- Recherche de répétition

- Recherche d'indice de temporalité
- Recherche de discrours raportés
- Recherche de différentes relations discursives liée aux token
- Recherche de motifs combinant mots, tokens, opérateurs logiques par expressions régulières et automates
- Outils pour la synthèse de fragments de grammaires à partir de ces résultats fournis par nos parseurs de segments de phrases.

Ces fonction peuvent être appliquées à des textes tout venants ou importés dans le corpus du logiciel, qui accepte les documents arabes de tout format (doc, docs, rtf, pdf, etc.) pour en extraire le texte.

La démonstration repose sur le déroulé de scénarios d'interaction grammaire corpus.

Kawâkib, analyse du contexte du token *mā*

OWI.Chat : Assistance sémantique pour un conseiller Chat, grâce à la théorie OWI

Christophe Dany, Ilhème Ghalamallah
OWI, 31 avenue du Général Leclerc, 92340 Bourg-la-Reine
christophe.dany@owi-tech.com

Résumé. La canal chat permet aux entreprises de transformer leur site web en un véritable lieu d'achat et de service. OWI a développé un outil d'assistance aux conversations en ligne (OWI.Chat), qui analyse les messages des internautes et conseille les conseillers en temps réel.

Abstract. Chat channel enables companies to transform their website into a real place of purchase and service. OWI developed an online conversations assistance solution (OWI.Chat). Its main task is to analyze the Chat requests and help agents in real time.

Mots-clés : Analyse sémantique, moteur sémantique Chat, Live Chat, Conversation en ligne, Traitement Automatique du Langage (TAL), Base de connaissance, Relation client
Keywords: Semantic analysis, semantic engine, Chat, Live Chat, Online conversation, Natural Language Procession (NLP), Agent Knowledge Base, Customer relationship

1. Contexte, enjeux et besoins

Le canal chat correspond aux nouveaux usages et comportements (mobilités, interactivité) ; il permet aux entreprises de transformer leur site web en un véritable lieu d'achat et de service, lieu où un conseiller peut à tout moment être sollicité ou proposer son aide de façon proactive. L'échange qui va alors se dérouler entre l'internaute et le conseiller est crucial pour l'entreprise : réussir une vente, résoudre un problème, et laisser à l'internaute le sentiment d'une expérience agréable.

De son côté, le conseiller n'est pas un expert sur tous les sujets possibles, et pas toujours un rédacteur irréprochable. L'entreprise est donc confrontée à des risques sérieux d'échec de la conversation :

- temps d'attente trop long pour l'internaute
- renvoi du conseiller vers une plateforme spécialisée
- informations erronées
- orthographe hasardeuse qui donne une mauvaise image

2. Chaîne de valeur sémantique OWI

2.1. Compréhension

Procédé innovant : à partir d'une connaissance de la langue modélisée sous forme de « schémas de comportement », un graphe qui met en interaction la totalité des signes présents, exploitant l'ensemble des dimensions du langage.

Les algorithmes à l'œuvre permettent de produire, pour une certaines connaissance de la langue, pour un message et pour un contexte sémantique (secteur d'activité), le graphe de qualité maximale.

2.2. Indexation

Suite à l'opération de Compréhension, l'opération d'Indexation extrait du graphe, notamment par linéarisation, toutes les notions qui y sont présentes. Toutes les notions présentes dans le message sont alors enregistrées dans un index.

2.3. Proximité

S'appuyant sur des dictionnaires (sectoriels…), la proximité mesure à quel point une notion attendue est présente dans un message. Catégorisation permet alors de mesurer la présence des données les plus structurantes pour l'activité traitée par les conseillers. Ces données structurantes sont paramétrées par l'administrateur sémantique de la solution.

2.4. Similarité

La mesure de similarité entre un message d'origine et un message analysé répond à deux questions :

- à quel point le message analysé évoque ce dont il s'agit dans le message d'origine ?
- à quel point le message analysé est centré sur ce dont il s'agit dans le message d'origine ?

Cette mesure permet au moteur OWI de s'inspirer des bonnes pratiques de traitement pour inciter à les reproduire.

3. Sémantique et traitement des demandes

3.1. Base de connaissance et conduite de conversation

En plus des documents, la base de connaissance adresse les problématiques propres à la conduite de conversation :

- aide au diagnostic et à la résolution d'incidents, sous forme d'arbre de décisions interactifs
- prise en compte des normes qualité (conduite de conversation, proactivité… etc)
- prise en compte des émotions de l'internaute (empathie, escalade... etc)

Outre le paramétrage de l'administrateur documentaire, elle est enrichie par l'apprentissage automatique des bonnes pratiques (voir ci-dessous) des critères d'accès à l'information.

Ces critères, modifiables par l'administrateur documentaire, combinent les proximités (données structurantes), les similarités (pour s'inspirer des bonnes pratiques), les données contextuelles (ex : la connaissance du client, de son abonnement…), et les informations propres à la conduite de conversation (ex : réponses différentes selon les étapes).

3.2. Accès à la base de connaissance par l'agent

Lorsqu'il reçoit le message de l'internaute, l'agent reçoit en même temps les modèles de réponses et les guides de traitements les plus pertinents, calculés par le moteur OWI. Il peut les consulter ou les utiliser directement pour répondre à l'internaute.

Il peut également accéder à la totalité de la base de connaissance, avec tous les moyens de recherche disponibles (mots clés, sémantique, full text, navigation). Lorsqu'il a trouvé le modèle recherché, il le sélectionne, ce qui a pour effet de :

- le copier pour être collé dans la fenêtre de réponse ;
- informer Pilotage de Conversation de l'orientation qu'il donne à la conversation (il peut, par exemple, déclencher ainsi un arbre de décision de diagnostic interactif qui va alors être piloté par le moteur)
- apprendre au moteur que ce modèle était un choix qui aurait pu être proposé, apprentissage qui sera ensuite utilisé).

3.3. Pilotage de conversation

Pour être en mesure de proposer les documents les plus adaptés, des algorithmes spécifiques tiennent compte des messages déjà échangés dans la conversation, de son degré d'avancement (introduction, traitement, conclusion, escalade), des émotions exprimées par l'internaute, des choix opérés par le conseiller dans la base de connaissance et, évidemment, du message que vient d'envoyer l'internaute pour mettre en œuvre notamment sémantiques et extraire de la base de connaissance les guides et modèles les plus pertinents.

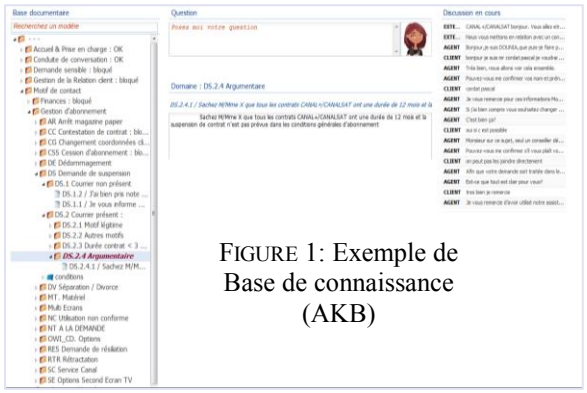


FIGURE 1: Exemple de Base de connaissance (AKB)

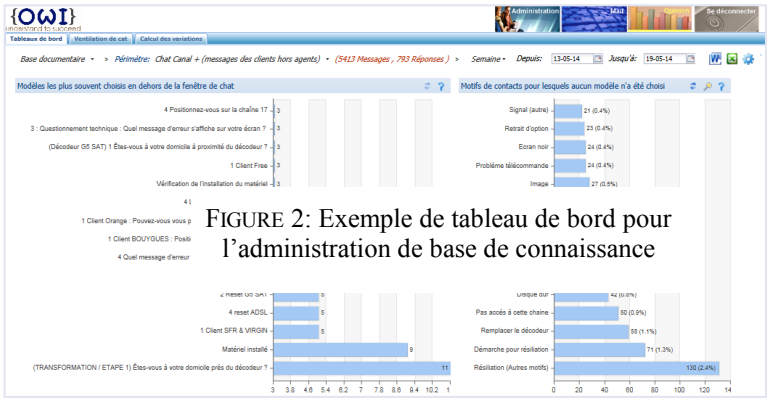


FIGURE 2: Exemple de tableau de bord pour l'administration de base de connaissance

3.4. Administration de la base de connaissance

L'administrateur de la base de connaissance dispose d'un tableau de bord l'informant sur :

- les nouveaux pratiques « choix de conseillers »
- les besoins non couverts (motifs de demandes sans modèles choisis)
- les modèles et guides les plus, ou les moins utilisés
- les taux d'utilisation des éléments de la base documentaire par équipe et par conseiller

Il peut alors compléter et mettre à jour la base de connaissance, ou proposer des actions d'informations pour l'adapter aux évolutions de l'activité et aux équipes de conseillers.

ZOMBILINGO : manger des têtes pour annoter en syntaxe de dépendances

Karën Fort[1] Bruno Guillaume [2] Valentin Stern [1]
(1) LORIA, Université de Lorraine
(2) LORIA, Inria Nancy Grand-Est
karen.fort@loria.fr, bruno.guillaume@loria.fr, valentin.stern@loria.fr

Résumé. Cet article présente ZOMBILINGO un jeu ayant un but (*Game with a purpose*) permettant d'annoter des corpus en syntaxe de dépendances. Les annotations créées sont librement disponibles sur le site du jeu.

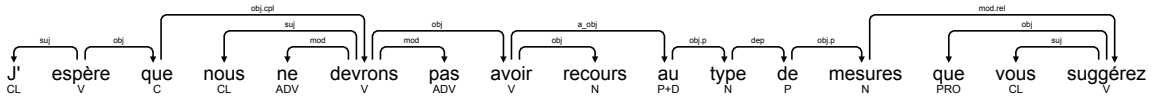
Abstract. This paper presents ZOMBILINGO, a Game With A Purpose (GWAP) that allows for the dependency syntax annotation of French corpora. The created resource is freely available on the game Web site.

Mots-clés : jeux ayant un but, complexité, annotation, syntaxe en dépendances.

Keywords: GWAP, complexity, annotation, dependency syntax.

La production de ressources linguistiques de grande taille est très coûteuse, en particulier en main d'œuvre. Ainsi, le coût d'annotation du Prague Dependency Treebank a été estimé à 600 000 dollars (Böhmová *et al.*, 2001). Une alternative pour produire des ressources est l'utilisation de la myriadisation (*crowdsourcing*), c'est-à-dire le recours à la « foule pour réaliser une tâche. Les jeux ayant un but, par exemple, ont été utilisés pour différentes tâches en TAL : JEUXDEMOTS [1] (Lafourcade, 2007) a pour but de créer un réseau lexical ; PHRASE DETECTIVES [2] (Chamberlain *et al.*, 2008) fait annoter un corpus en anaphores. Ces deux jeux ont eu un succès considérable et ont permis de créer des ressources de qualité raisonnable pour un coût réduit. Le premier fait appel au sens commun et le deuxième à des connaissances scolaires. Dans d'autres domaines, il a été possible d'utiliser un jeu pour des tâches nettement plus complexes et qui nécessitent une formation des personnes qui participent. Ainsi, dans FOLDIT (Cooper *et al.*, 2010) les joueurs doivent manipuler des représentations 3D de protéines pour étudier la façon dont elle peuvent interagir. ZOMBILINGO est inspiré de ces succès et a pour but de faire réaliser à des joueurs une tâche de TAL réputée complexe : annoter des dépendances syntaxiques.

Les données que nous souhaitons produire sont des analyses en dépendances syntaxiques compatibles avec celles utilisées pour le corpus Sequoia (Candito & Seddah, 2012). Elles sont illustrées par l'exemple ci-dessous.



Ce choix nous permet d'utiliser le corpus Sequoia comme amorce pour ZOMBILINGO, notamment pour la phase de formation des joueurs. Le système sera ensuite alimenté par des phrases issues de textes libres de droits, qui seront pré-annotés à l'aide d'analyseurs syntaxiques. Quand une nouvelle phrase est ajoutée dans la base de données, sa pré-annotation est considérée comme correcte ; dans la suite du jeu, si suffisamment de joueurs donnent un avis contraire à la pré-annotation, l'annotation de la phrase considérée est modifiée pour en tenir compte. Il est donc possible à tout moment de faire une extraction de la ressource annotée en syntaxe, qui tient compte de ce que tous les joueurs ont fait précédemment.

L'un des enjeux essentiels de ce jeu est d'être capable de gérer la complexité de la tâche. Il n'est bien entendu pas possible de demander à un joueur de produire l'annotation d'une phrase complète ; il faut décomposer la tâche globale en une série de tâches plus élémentaires qui peuvent être confiées à des joueurs sans les décourager. Dans ZOMBILINGO, cette gestion

1. Voir : `http://www.jeuxdemots.org`.
2. Voir : `http://anawiki.essex.ac.uk/phrasedetectives`.

de la complexité s'appuie sur le découpage de la tâche suivant les différents phénomènes linguistiques présents dans la phrase. Ce découpage permet également de mettre en place des séances de formations pour chacun des phénomènes et donc de ne pas surcharger les joueurs d'informations : le joueur choisit un phénomène, suit la formation correspondante, et peut ensuite commencer à jouer avec ce phénomène.

Un autre élément essentiel à la réussite de ZOMBILINGO est la motivation des joueurs. En effet, la production d'une ressource de grande ampleur de qualité n'est possible que si beaucoup de joueurs utilisent le jeu et si une proportion raisonnable d'entre eux restent longtemps et reviennent régulièrement jouer. Pour attirer les joueurs, le design est un élément essentiel. Nous avons choisi le thème des zombies parce qu'il est fédérateur dans le monde du jeu et par clin d'œil à la notion de tête d'une dépendance linguistique : annoter c'est « manger des têtes », c'est donc une tâche pour les zombies ! La capture d'écran ci-dessous présente l'interface du jeu.

1. profil du joueur
2. progression de la partie
3. aide interactive
4a. mot joué
4b. relation ou phénomène à annoter
4c. « main » pour le choix de la réponse
5. accès aux objets du jeu

Les mécanismes qui encouragent les joueurs à jouer suffisamment longtemps et à revenir régulièrement sont aussi un élément clé de la réussite du jeu. En se basant sur les notions souvent utilisées pour les jeux (sérieux ou non), nous avons prévu différents mécanismes qui correspondent aux différents types de joueurs existants. Ainsi, les mécanismes que nous avons mis en place ont pour but de répondre aux attentes des quatre types de joueurs identifiés par Bartle (1996) : *killers*, *achievers*, *explorers* et *socializers*.

Les données produites par les joueurs permettront de produire un corpus annoté en dépendances de surface qui sera mis à jour en continu en fonction des actions des joueurs. Ce corpus sera mis à disposition librement.

Les auteurs tiennent à remercier Hadrien Chastant pour la première maquette, Charles Ancé pour ses magnifique dessins, Alice Guyot pour les éléments de design et Mathieu Lafourcade pour son aide dans la conception du jeu.

Références

BARTLE R. (1996). Hearts, clubs, diamonds, spades : Players who suit MUDs. *The Journal of Virtual Environments*.

BÖHMOVÁ A., HAJIČ J., HAJIČOVÁ E. & HLADKÁ B. (2001). The prague dependency treebank : Three-level annotation scenario. In A. ABEILLÉ, Ed., *Treebanks : Building and Using Syntactically Annotated Corpora*. Kluwer Academic Publishers.

CANDITO M. & SEDDAH D. (2012). Le corpus Sequoia : annotation syntaxique et exploitation pour l'adaptation d'analyseur par pont lexical. In *Traitement Automatique des Langues Naturelles (TALN)*, Grenoble, France.

CHAMBERLAIN J., POESIO M. & KRUSCHWITZ U. (2008). Phrase Detectives : a web-based collaborative annotation game. In *Proceedings of the International Conference on Semantic Systems (I-Semantics'08)*.

COOPER S., TREUILLE A., BARBERO J., LEAVER-FAY A., TUITE K., KHATIB F., SNYDER A. C., BEENEN M., SALESIN D., BAKER D. & POPOVIĆ Z. (2010). The challenge of designing scientific discovery games. In *Proceedings of the Fifth International Conference on the Foundations of Digital Games*, FDG '10, p. 40–47.

LAFOURCADE M. (2007). Making people play for lexical acquisition. In *Proceedings of the 7th Symposium on Natural Language Processing (SNLP 2007)*.

Ubiq : une plateforme de collecte, analyse et valorisation des corpus

François-Régis Chaumartin[1]
(1) Proxem, 19 boulevard de Magenta, 75010 Paris
frc@proxem.com

Résumé. Proxem édite Ubiq, une plateforme de collecte de documents et d'analyse sémantique, capable d'extraire des informations pertinentes à partir du contenu de vastes corpus. Les documents analysés sont d'une grande diversité : opinions collectées sur des sites web, emails de réclamation ou de demande d'information, réponse à des questions ouvertes dans des sondages, offres ou demandes d'emploi, etc. La reconnaissance des entités nommées joue un rôle central car c'est un préalable à d'autres traitements sémantiques. La conception d'un module de reconnaissance d'entités nommées nécessite généralement un investissement important en amont, avec une adaptation de domaine. Ubiq propose une approche d'apprentissage faiblement supervisé de l'extraction d'entités nommées qui tient compte du corpus collecté et de ressources externes (Wikipédia). La méthode et l'outillage développés permettent de déterminer à la volée, en interaction avec l'utilisateur, la granularité des types d'entités adaptée à un corpus de texte tout-venant.

Abstract. Proxem publishes Ubiq, a platform for web crawling and semantic analysis, which can extract relevant information from large corpus. Documents are of great variety: reviews crawled from websites, emails about complaints or requests for information, answers to open questions in surveys, employment offers or job applications, etc. Named Entity Recognition plays a key role since it is a prerequisite to further semantic processing. The design of a NER module generally requires a significant upfront investment with some domain adaptation. Ubiq proposes a semi-supervised approach to NER that takes into account the crawled corpus and external resources (Wikipedia). The proposed method and tools allow to get on the fly, with some user interaction, the type granularity of entities suitable for a given corpus.

Mots-clés : entités nommées, désambiguïsation, apprentissage, Wikipédia, catégorisation.
Keywords: named entities, disambiguation, machine learning, Wikipedia, categorization.

1 Objectif : amorcer, à la volée, l'extraction d'entités nommées d'un corpus

La multiplication d'avis de consommateurs sur le web permet aujourd'hui d'effectuer des enquêtes dans divers domaines applicatifs en allant chercher des documents à analyser sur Internet. Ce type d'enquête est généralement mené avec des phases (i) de collecte de documents textuels à partir de sources web, (ii) d'analyse sémantique des contenus et (iii) de présentation de ces informations. La phase d'analyse des contenus débute par des tâches comme le découpage de chaque page web en zones, l'identification de la langue du texte et éventuellement une correction orthographique. L'extraction des entités nommées est réalisée après ces prétraitements ; elle associe un type à chaque instance, dont certaines peuvent être ambiguës (Orange$_{[Fruit]}$ et Orange$_{[Marque\ télécom]}$ par exemple). D'autres traitements sémantiques peuvent suivre : extraction de relations entre entités, analyse d'opinions, classification, priorisation… La qualité de ces traitements dépend directement de celle obtenue lors de la reconnaissance des entités nommées. Sur des projets du domaine de la grande distribution, nous avons atteint une F-mesure de 97% dans la détection des marques et produits ; néanmoins, ce résultat n'a été possible qu'au prix d'un effort manuel significatif ; des semaines de travail ont été nécessaires pour créer les ressources linguistiques et résoudre les principales ambiguïtés présentes. La variété des sujets abordés est le sujet le plus épineux dans ce cadre traité. Lors de nos projets, nous avons été confrontés à la difficulté de constitution des ressources lexicales dans des domaines applicatifs multiples : banque de détail, assurance, automobile, recrutement, télécommunications, mode, bricolage, cosmétique, pétrole, industrie du vin, pathologies médicales… La récurrence de cette problématique nous a poussé à développer différentes approches permettant d'être opérationnel rapidement sur une thématique nouvelle. Ubiq permet la mise en œuvre simultanée de deux approches complémentaires :

– La première a pour objectif la rapidité de mise en œuvre, et consiste à faciliter l'utilisation d'un annotateur générique préexistant comme le système de classification thématique générique décrit dans (Chaumartin, 2013) ou Open Calais : un tel annotateur est prévu pour reconnaître un jeu d'entités prédéfini apparaissant dans des documents d'un certain type (par exemple personnes, lieux et organisations au sein d'articles de presse). Son intérêt est d'être opérationnel immédiatement… sous réserve d'être adapté au corpus à traiter (faible rappel sur un corpus arbitraire).

– La seconde vise à améliorer la finesse d'analyse, en réalisant un annotateur sur mesure, spécifique à un domaine ou à un corpus particulier. Il devient alors possible d'identifier les entités pertinentes d'une façon arbitrairement fine ; dans le domaine du vin, on peut par exemple regrouper toutes les boissons alcoolisées ensemble, ou au contraire choisir de distinguer les vins, bières ou apéritifs cités dans le corpus. Mais l'investissement préalable (pour écrire des règles, annoter manuellement un corpus d'apprentissage ou valider des ressources) est généralement important.

Proxem a innové sur ce dernier point en développant, grâce à la plate-forme Antelope (Chaumartin, 2012), une méthode qui fournit rapidement un annotateur capable d'extraire des entités nommées adapté à tout corpus monolingue (sous réserve qu'il soit relativement homogène). Le système est suffisamment simple à administrer pour être mis en œuvre par un utilisateur sans compétence linguistique forte. Le processus d'amorce d'un tel annotateur repose sur un apprentissage faiblement supervisé qui vise à déterminer à la volée la granularité des types d'entités, avec les interactions suivantes :

1. L'utilisateur constitue un corpus homogène (par exemple par collecte ciblée sur le web).
2. Le système calcule un graphe de thématiques associées à ce corpus, en prenant comme référence les catégories de la Wikipédia dans la langue du corpus.
3. L'utilisateur valide au sein du graphe les catégories proposées qui semblent pertinentes et supprime les autres.
4. Le système restreint les entités nommées candidates aux articles de la Wikipédia rattachés à ces catégories et susceptibles de définir une entité. Ces données alimentent un composant de REN fonctionnant avec des listes d'entités et capable de gérer l'éventuelle homonymie avec des informations de désambiguïsation.
5. Le système effectue une première REN sur le corpus, et compte le nombre effectif d'occurrences de chaque entité candidate. Par ailleurs, l'appartenance d'une entité à une classe et les relations d'hyperonymie entre classes ont été précalculées. Cette double information permet d'afficher à l'utilisateur une arborescence de classes d'entités potentiellement pertinentes dans le contexte du corpus.
6. L'utilisateur amorce une taxonomie ad-hoc en sélectionnant les classes qu'il souhaite reconnaître. Il peut éventuellement en fusionner (pour regrouper des classes Marque, Société et Entreprise, par exemple). Il peut aussi choisir unitairement les entités à conserver ou à exclure.
7. Le système annote à nouveau le corpus, en tenant compte des choix de l'utilisateur. Des post-traitements aident à identifier de nouvelles entités candidates (absentes de la Wikipédia, ou non liées à l'une des catégories choisies lors de la première interaction, ou dont l'hyperonyme n'a pas été précalculé correctement).
8. L'utilisateur valide parmi ces nouvelles entités proposées celles qu'il estime pertinentes.
9. Le système entraîne un composant d'apprentissage (par CRF) sur ce corpus annoté.

2 Résultats préliminaires

Nous avons évalué ce système sur un corpus de 15 000 avis de consommateurs sur l'univers mode & chaussure, le système propose à l'utilisateur les classes d'entités suivantes : 73 entreprises, 69 textiles, 55 photographes, 53 accessoires, 38 types de chaussures. L'utilisateur peut corriger d'éventuelles erreurs et préciser les classes que le système devra réellement reconnaître. La table 1 compare les résultats obtenus en quelques heures par ce système avec ceux d'un analyseur dont la création manuelle a nécessité deux semaines de travail avec une démarche classique. Les résultats expérimentaux sont encourageants ; les entités extraites sont de plus liées aux entrées correspondantes de Wikipédia. Il reste toutefois à améliorer le rappel de cette méthode et à comparer les résultats obtenus sur des jeux de données standard. L'approche proposée est en principe facilement généralisable à de nouvelles langues.

	Analyseur créé manuellement en 10 jours	Analyseur créé avec la méthode décrite ici
Classe Chaussure	71 entités créées manuellement et repérées dans les documents (63000 occurrences)	145 entités proposées par le système dont 72 présentes dans le corpus (36 000 occurrences)
Entités spécifiques	boots (3 878), spartiate (612), sneaker (534)…	babies (431), motarde (245), brogue (61)…
Entités communes	chaussure (11 110), bottine (3 906), tong (1 563), rangers (540), chaussure de sécurité (81)…	

TABLE 1 : Comparaison entre un analyseur créé manuellement et avec la méthode décrite ici

Références

CHAUMARTIN F.-R. (2012). *Antelope, une plate-forme de TAL permettant d'extraire les sens du texte : théorie et applications de l'ISS*. Thèse de doctorat, Université Paris Diderot.

CHAUMARTIN F.-R. (2013). Apprentissage d'une classification thématique générique et cross-langue à partir des catégories de la Wikipédia. Actes de *TALN*, 659–666.

Zodiac : Insertion automatique des signes diacritiques du français

Fabrizio Gotti et Guy Lapalme
RALI, Université de Montréal, CP 6128 Succursale Centre-Ville, Montréal, Canada, H3C 3J7
{gottif,lapalme}@iro.umontreal.ca

Résumé. Nous proposons dans cette démonstration de présenter le logiciel Zodiac, permettant l'insertion automatique de diacritiques (accents, cédilles, etc.) dans un texte français. Zodiac prend la forme d'un complément Microsoft Word sous Windows permettant des corrections automatiques du texte au cours de la frappe. Sous Linux et Mac OS X, il est implémenté comme un programme sur ligne de commande, se prêtant naturellement à lire ses entrées sur un « pipeline » et écrire ses sorties sur la sortie standard. Implémenté en UTF-8, il met en œuvre diverses librairies C++ utiles à certaines tâches du TAL, incluant la manipulation de modèles de langue statistiques.

Abstract. In this demo session, we propose to show how the software module Zodiac works. It allows the automatic insertion of diacritical marks (accents, cedillas, etc.) in text written in French. Zodiac is implemented as a Microsoft Word add-in under Windows, allowing automatic corrections as the user is typing. Under Linux and Mac OS X, it is implemented as a command-line utility, lending itself naturally to be used in a text-processing pipeline. Zodiac handles UTF-8, and showcases some useful C++ libraries for natural language processing, including statistical language modeling.

Mots-clés : aide à la rédaction, diacritiques, modèles de langue probabilistes.

Keywords: text editing, diacritical marks, statistical language models.

1 Contexte

Les *signes diacritiques* sont des symboles graphiques (accents, cédilles) combinés à des lettres déjà existantes afin d'en modifier la phonétique ou d'éviter la confusion entre des mots homographes (p.ex. « cote », « côte » et « côté »). Il en existe une vingtaine dans l'alphabet romain, et le français en utilise couramment cinq, soit la cédille, le tréma, les accents aigu, grave et circonflexe. Critiqués, on rencontre rarement le tilde (« cañon ») ou le rond en chef (« ångström »).

Malgré leur caractère indispensable à la sémantique du texte (« interne » ou « interné » dans un hôpital psychiatrique) et à la justesse orthographique, les diacritiques sont omis dans des situations relativement fréquentes. Les communications électroniques (courriels, textos, forums de discussion, etc.) présentent ces problèmes, que ce soit à cause de leur caractère informel, ou simplement parce que les méthodes de saisie rendent difficile l'insertion des diacritiques. Traditionnellement, ce dernier problème était surtout dû à des dispositions de clavier peu compatibles avec le français, mais l'avènement des tablettes et téléphones intelligents accentue désormais la difficulté à cause d'entraves ergonomiques. Par ailleurs, les apprenants de la langue française peuvent éprouver de la difficulté avec ces symboles.

Il peut donc s'avérer utile dans ces contextes de disposer d'un système d'insertion automatique de diacritiques à partir d'un texte qui en est dépourvu. La tâche n'est pas triviale, car un mot sans accents peut présenter plusieurs candidats avec diacritiques (voir l'exemple de « cote » plus haut). Il y a plus de dix ans, notre laboratoire proposait le logiciel Réacc, qui faisait l'insertion automatique des diacritiques (Simard & Deslauriers, 2001). Fondé sur un modèle de Markov caché, le logiciel offrait de bonnes performances, mais nécessitait le difficile maintien de librairies C faites sur mesure, dont l'instabilité a fini par condamner le logiciel. Une solution de rechange a donc été conçue.

Nous proposons ainsi une démonstration du logiciel Zodiac, un module effectuant la restauration de diacritiques automatiquement, et fondé sur un modèle statistique de langue et sur un lexique du français à large couverture. Notre démonstration montrera l'interface web de Zodiac (basée sur Linux), l'utilitaire ligne de commande, ainsi qu'un complément Zodiac à Microsoft Word permettant de faire l'insertion des diacritiques au cours de la frappe, de façon interactive, sous Windows.

2 Zodiac

2.1 Mode de fonctionnement

Zodiac commence par segmenter le texte entré en phrases et en mots, à l'aide de la librairie C++ ICU (`http://icu-project.org/`). ICU supprime ensuite tous les diacritiques déjà présents. Chaque jeton sans diacritiques est recherché dans une liste précompilée afin de trouver les candidats avec diacritiques (p.ex. : mais → mais ou maïs). Un modèle de langue statistique parcourt la phrase et utilise le contexte pour choisir une substitution candidate. Ainsi, « du maïs » sera préféré à « du mais ». Pour gérer l'explosion combinatoire lorsque des mots ambigus se suivent, une recherche en faisceau est utilisée. Le modèle de langue utilisé est un modèle trigramme entraîné avec la librairie SRILM (Stolcke, 2002). Le corpus d'entraînement consiste en 1 M de phrases des domaines politique et journalistique. Le modèle est compilé sous forme binaire avec la librairie C++ KenLM (Heafield, 2011) afin d'être chargé en 0,1 s en RAM.

2.2 Interface

La démonstration montrera les modules logiciels dans lesquels Zodiac est intégré. Une démo web est également disponible[1]. Le module est entièrement mis en œuvre en Unicode, grâce à la librairie ICU. **Sous Windows**, Zodiac est un complément Word à deux modes de fonctionnement : l'utilisateur peut activer l'insertion de diacritiques au fur et à mesure de la frappe ou il peut décider de sélectionner un passage d'un texte et d'en corriger les diacritiques avec Zodiac. **Sous Linux et Mac OS X**, la démonstration de Zodiac illustrera qu'il peut se comporter comme un filtre de texte typique, c'est-à-dire un programme lisant du texte UTF-8 et produisant une sortie textuelle corrigée. Il se prête donc à une utilisation par « pipeline », permettant le chaînage de plusieurs programmes.

2.3 Évaluation

L'évaluation d'un système d'insertion de diacritiques comme Zodiac est relativement simple. Il suffit de choisir un texte où les diacritiques figurent, et de le soumettre à Zodiac. Puisque celui-ci commence par supprimer les diacritiques pour ensuite les réinsérer selon l'algorithme décrit plus haut, les diacritiques produits en sortie sont exclusivement l'œuvre de Zodiac. L'évaluation consiste alors à comparer le texte original (la référence) avec la sortie de Zodiac (le candidat). Sur un corpus constitué de textes juridiques et littéraires, on constate que le système commet des erreurs en moyenne sur 0,54 % du nombre total de mots de corpus. La démonstration montrera certains aspects du logiciel affectant sa performance. Ainsi, des ambiguïtés subsistent pour l'insertion d'accents sur la première majuscule des noms propres (p.ex. « Eric » ou « Éric »). Cette difficulté est épineuse, à cause des variantes orthographiques et culturelles dans la rédaction des prénoms. Il reste également à déterminer s'il est constructif de supprimer les diacritiques préexistants avant leur insertion.

3 Perspectives

Une extension naturelle de Zodiac serait de s'intéresser à d'autres langues pour lesquelles des ambiguïtés comparables existent. Ainsi, le vietnamien utilise une orthographe appelée quốc ngữ, usant de nombreux diacritiques parfois combinés sur la même lettre, rendant la saisie malaisée. Zodiac utilise justement des librairies C++ capables de traiter du texte dans plusieurs langues, sans règles *ad hoc* spécifiques au français, donc cette voie nous paraît prometteuse.

Références

HEAFIELD K. (2011). KenLM : faster and smaller language model queries. In *Proceedings of the EMNLP 2011 Sixth Workshop on Statistical Machine Translation*, p. 187–197, Edinburgh, Scotland, United Kingdom.

SIMARD M. & DESLAURIERS A. (2001). Real-time automatic insertion of accents in French text. *NLE*, **7**, 149–165.

STOLCKE A. (2002). SRILM - an extensible language modeling toolkit. In *In Proceedings Of The 7th International Conference On Spoken Language Processing (ICSLP 2002)*, p. 901–904.

[1] `http://rali.iro.umontreal.ca/rali/?q=fr/projet-zodiac`

Le système STAM

Mehdi Embarek[1]
(1) MK SOFT, 11 rue des fossés St Marcel, 75005 Paris
embarekm@gmail.com

Résumé. Le projet STAM aborde la problématique de la transcription automatique du langage texto (SMS) et plus particulièrement la traduction des messages écrits en arabe dialectal. L'objectif du système STAM est de traduire automatiquement des textes rédigés en langage SMS dans un dialecte parlé dans le monde arabe (langue source) en un texte facilement interprétable, compréhensible et en bon français (langue cible).

Abstract. The STAM project addresses the problem of automatic transcription of SMS language and especially the translation of messages written in Arabic dialect. The objective of STAM system is to automatically translate texts written in SMS language in a dialect spoken in the Arab World (source language) into a French text (target language), interpretable and understandable.

Mots-clés : Dialecte, SMS, Transcription, STAM.
Keywords: Dialect, SMS, Transcription, STAM.

L'évolution des technologies de communication a permis de développer différentes formes de langage. En plus du langage naturel utilisé dans tous les documents écrits, un autre langage, que l'on retrouve dans des documents ou des textes informels, a vu le jour. Il s'agit du langage SMS[1] (ou langage texto) qu'il faudra dorénavant considérer comme un nouveau langage à part entière. Aussi, depuis l'expansion des réseaux sociaux, des blogs et des forums de discussions, les internautes utilisent de plus en plus ce langage pour exprimer leur avis concernant une actualité ou commenter un évènement. Et pour certaines communautés (maghrébine par exemple), on emploie même dans les messages des termes issus de leur dialecte local. Ce qui rend la compréhension du texte presque impossible pour les personnes ne parlant pas le dialecte employé. Ces textes non structurés peuvent être considérés comme des sources d'informations et il serait intéressant de pouvoir les exploiter et les analyser afin d'en extraire le contenu informationnel en se basant sur des outils et techniques adaptés. De nombreux travaux ont été menés dans ce sens afin d'étudier plus particulièrement les spécificités des dialectes (Guella, 2011) (Vanhove, 1999) ou encore d'élaborer des terminologies (dictionnaires) propre à chaque pays (www.speakmoroccan.com pour le Maroc, www.arabetunisien.com pour la Tunisie). Cependant, peu de travaux se sont penchés sur le développement ou la proposition d'un système de traduction automatique.

Le projet STAM (Système de Transcription AutoMatique) (http://www.stam-dz.com) est un projet de recherche industrielle soutenu conjointement par la société Med Point Dz et la société MK Soft. Le projet aborde la problématique de la transcription (traduction) automatique du langage texto (SMS) et plus particulièrement la traduction des messages écrits en arabe dialectal. L'objectif du système STAM est de traduire automatiquement des textes rédigés en langage SMS dans un dialecte parlé dans le monde arabe (langue source) en un texte facilement interprétable, compréhensible et en bon français (langue cible).

Dans notre développement, nous nous sommes particulièrement intéressés au dialecte algérien, c'est-à-dire à tous les dialectes parlés en Algérie (l'algérois, l'oranais, le constantinois, le kabyle, etc.). Le dialecte algérien est un langage assez particulier, un langage fondé sur un mélange de plusieurs langues dont la plupart des termes ont été repris à l'origine de la langue arabe littéraire et le français. Pour mieux illustrer certaines de ses caractéristiques dans le langage SMS, prenons cet exemple : « nakol fel restaurant » (comprendre : je mange au restaurant). L'exemple montre qu'un message écrit en dialecte algérien peut contenir un terme issu d'une langue étrangère, ici du français (restaurant). De plus, étant donné qu'il n'existe aucune règle d'écriture, un mot peut éventuellement être écrit de plusieurs manières. Par exemple, le mot «restaurant» (resto, mat3am, …) ou encore le mot «nakol» (nacol, nakoul, …) (comprendre : manger).

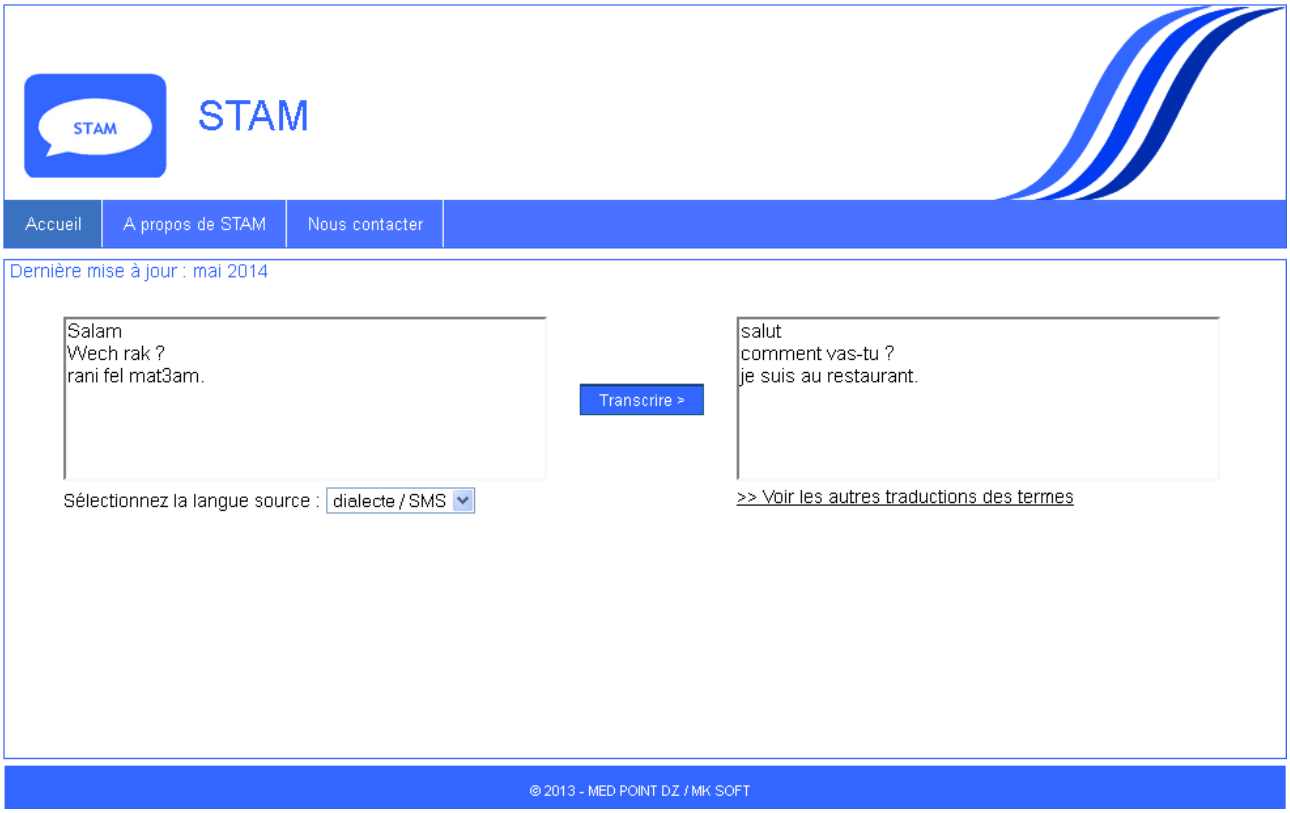
[1] Short Message Service

Une autre caractéristique du dialecte algérien (valable également pour la langue arabe) concerne la présence de chiffres dans certains mots (mat3am) pour exprimer principalement une certaine prononciation (problème phonétique) spécifique aux mots arabes.

Enfin, STAM est un système souple et facilement paramétrable permettant d'intégrer d'autres dialectes parlés dans d'autres pays. On parle ici d'une solution multi-dialectale. En effet, le but du projet STAM est de ne pas se limiter uniquement au dialecte local (algérien) mais aussi de prendre en compte, par la suite et dans la mesure du possible, les autres dialectes disponibles en commençant par les pays du Maghreb (Tunisie, Maroc, Libye, Mauritanie), puis les pays du Moyen Orient (Egypte, Syrie, Liban, Arabie Saoudite, etc.). La réalisation d'un tel outil passe par la constitution d'une importante base terminologique.

Actuellement, le système STAM s'appuie sur une terminologie multi-dialectale évolutive. Cette terminologie regroupe les termes et expressions employés dans les pays suivants : Algérie, Maroc et Tunisie. Pour les transcriptions, STAM repose également sur un ensemble de règles d'écriture et d'algorithmes. On peut citer l'algorithme « STAM_Align » qui permet d'effectuer des alignements entre la requête utilisateur et le contenu de la terminologie.

Ci-dessous une figure représentant un exemple de transcription dans le système STAM :



Références

GUELLA N. (2011). Emprunts lexicaux dans des dialectes arabes algériens. *Synergies Monde arabe 8*, 81-88.

VANHOVE M. (1999). Les dialectes arabes des régions sud, centre et est du Yémen : perspectives et recherche. *Chroniques Yéménite*, 95-100.

DictaNum : système de dialogue incrémental pour la dictée de numéros.

Hatim Khouzaimi[1,2] Romain Laroche[1] Fabrice Lefèvre[2]
(1) Orange Labs, 38-40 Avenue du Général Leclerc, 92794 Issy-les-Moulineaux, France
(2) Laboratoire Informatique d'Avignon, 339 Chemin des Meinajaries, 84911 Avignon, France
hatim.khouzaimi@orange.com, romain.laroche@orange.com, fabrice.lefevre@univ-avignon.fr

Résumé. Les stratégies de dialogue incrémentales offrent une meilleure réactivité, une expérience utilisateur plus aboutie et une réduction du risque de désynchronisation. Cependant, les systèmes de dialogue incrémentaux sont basés sur une architecture logicielle dont l'implantation est longue, difficile et donc coûteuse. Pour faciliter cette évolution d'architecture, nous proposons de simuler un comportement incrémental en ajoutant une surcouche à un service de dialogue traditionnel existant. DictaNum est un démonstrateur de dialogue incrémental mettant en œuvre cette démarche. Sa tâche consiste à recueillir des numéros auprès des utilisateurs. Grâce à son fonctionnement incrémental, il autorise une correction rapide des erreurs au fil de la dictée.

Abstract. Incremental dialogue strategies are more reactive, offer a better user experience and reduce desynchronisation risks. However, incremental dialogue systems are based on architectures that are long, difficult and hence costly to implement. In order to make this architecture evolution easier, we suggest to simulate incremental behavior by adding a new layer to an existing traditional service. DictaNum is an incremental dialogue demonstrator that uses this approach. It collects numbers dictated by the user. Thanks to its incremental behavior, it makes it possible to rapidly correct errors on the fly.

Mots-clés : Systèmes de Dialogue, Traitement Incrémental, Architecture des Systèmes de Dialogue.

Keywords: Dialogue Systems, Incremental Processing, Dialogue Systems Architectures.

1 Introduction

Les systèmes de dialogue incrémentaux traitent les paroles de l'utilisateur au fur et à mesure qu'elles sont prononcées, sans attendre la fin de la requête comme c'est le cas des systèmes traditionnels. Les architectures incrémentales proposées dans la littérature (Dohsaka & Shimazu, 1997; Allen *et al.*, 2001; Schlangen & Skantze, 2011) nécessitent une construction intégrale du système. Afin d'éviter cela et les coûts que cela engendre, nous proposons de rajouter un module intermédiaire entre le client et le service appelé *Scheduler* (Khouzaimi *et al.*, 2014) (fonctionnement similaire à celui de l'Incremental Interaction Manager dans (Selfridge *et al.*, 2012) et du Micro-turn Interaction Manager dans (Hastie *et al.*, 2013)). Celui-ci permet de simuler un comportement incrémental vu du client sans devoir modifier le comportement interne du service (des modifications à l'échelle applicative sont cependant nécessaires). DictaNum est un service de collecte de numéros fonctionnant suivant le même principe. Ce système est inspiré de NUMBERS (Skantze & Schlangen, 2009) qui est basé sur une architecture complètement incrémentale.

2 Description du fonctionnement

Un service de dialogue recueillant des numéros est déployé sur un serveur distant. Celui-ci a été développé à l'aide de la suite logicielle Disserto d'Orange Labs. Le *Scheduler* est également déployé sur le même serveur (ce qui n'est pas obligatoire, les deux entités pouvant êtres déployées sur des machines différentes). Le client se présente sous la forme d'une page HTML communiquant en AJAX avec le *Scheduler*. Par ailleurs, celle-ci est dotée d'un module JavaScript utilisant la web API de Google pour effectuer les tâches de reconnaissance et de synthèse vocales.

La sortie incrémentale de la reconnaissance vocale est envoyée à intervalles réguliers au *Scheduler*. Chacun de ces intervalles est appelé *micro-tour* dont la durée doit être spécifiée sur l'interface du démonstrateur. Si pendant n micro-tours, la sortie du module de reconnaissance vocale reste inchangée, on dit qu'un *micro-silence* est détecté. De même, on dit qu'un *silence* est détecté après $m > n$ micro-tours sans aucun changement de la phrase courante de l'utilisateur. Les paramètres n et m doivent également être spécifiés dans l'interface.

Le *Scheduler* envoie toutes les réponses intermédiaires au client. Cependant, seules certaines sont désignées pour être prononcées par le module de synthèse vocale (dans le cas de DictaNum, seules les réponses suivant un micro-silence ou un silence sont prononcées). Le *Scheduler* a aussi pour tâche de décider des moments de prise de parole par le système.

Une autre caractéristique importante du dialogue incrémental est la possiblité pour l'utilisateur d'interrompre le système pendant qu'il parle. Dans le cadre de notre architecture, cela est possible à certains endroits du dialogue définis au niveau du service. Celui-ci renvoie des messages en plusieurs blocks reconnaissables à l'aide d'un séparateur : *block1 [sep] block2 [sep] block3...*. Une fois reçu par le client, celui-ci le prononce fragment par fragment, et en cas d'interruption de l'utilisateur, le couple {block durant lequel l'interruption a eu lieu, contenu de l'interruption} est renvoyé au service qui réagit en conséquence.

3 Usage du service

L'utilisateur est invité à dicter son numéro suite à un message vocal. Au cours de sa dictée, si un micro-silence est détecté après un fragment de numéro, le système prend la parole et répète ce dernier fragment. Ensuite, l'utilisateur peut continuer sa dictée normalement ou bien corriger le dernier fragment en commençant sa prochaine phrase par 'Non'. Par exemple :

– **Système :** Bonjour. Bienvenue dans le service DictaNum. Veuillez entrer votre numéro.
– **Utilisateur :** 01 45 (suivi d'un micro-silence)
– **Système :** 01 45
– **Utilisateur :** 28 (suivi d'un micro-silence)
– **Système :** 38 (erreur de reconnaissance vocale)
– **Utilisateur :** Non, 28 (suivi d'un micro-silence)
– **Système :** Désolé, 28
– **Utilisateur :** 79 88 (suivi d'un micro-silence)
– **Système :** 79 88
– **Utilisateur :** (silence)
– **Système :** Le numéro dicté est le : 01 45 28 79 88. Est-ce bien cela ?
– **Utilisateur :** Oui
– **Système :** Merci d'avoir utilisé DictaNum. A bientôt.

Le système comprend que la dictée est terminée quand il détecte un long silence. Par la suite, le numéro complet est récapitulé et l'utilisateur peut intervenir durant ce feedback pour effectuer une correction locale. Les corrections peuvent être faites par lots de deux chiffres par exemple :

– **Système :** Le numéro dicté est : 01 45 28 72
– **Utilisateur :** Non, 62
– **Système :** Désolé : 01 45 28 62 59. Est-ce bien cela ?
– **Utilisateur :** Oui.
– **Système :** Merci d'avoir utilisé DictaNum. A bientôt.

4 Conclusion et travaux futurs

DictaNum montre qu'en utilisant un module intermédiaire entre le client et le service (le *Scheduler*), il est possible de simuler un comportement incrémental à moidre coût. Cependant la version du *Scheduler* utilisée ici est élémentaire et nous projetons d'en faire un réel module de décision intégrant un fonctionnement plus élaboré (analyse du contexte de dialogue, détection de problèmes, correction d'erreurs...).

Références

ALLEN J., FERGUSON G. & STENT A. (2001). An architecture for more realistic conversational systems. In *6th international conference on Intelligent user interfaces*.

DOHSAKA K. & SHIMAZU A. (1997). A system architecture for spoken utterance production in collaborative dialogue. In *IJCAI*.

HASTIE H., AUFAURE M.-A., ALEXOPOULOS P. & AUTRES (2013). Demonstration of the parlance system : a data-driven incremental, spoken dialogue system for interactive search. In *Proceedings of the SIGDIAL 2013 Conference*.

KHOUZAIMI H., LAROCHE R. & LEFÈVRE F. (2014). Vers une approche simplifiée pour introduire le caractère incrémental dans les systèmes de dialogue. In *Proceedings of the TALN 2014 Conference*.

SCHLANGEN D. & SKANTZE G. (2011). A general, abstract model of incremental dialogue processing. *Dialogue and Discourse*, **2**, 83–111.

SELFRIDGE E. O., ARIZMENDI I., HEEMAN P. A. & WILLIAMS J. D. (2012). Integrating incremental speech recognition and pomdp-based dialogue systems. In *Proceedings of the 13th Annual Meeting of the Special Interest Group on Discourse and Dialogue*.

SKANTZE G. & SCHLANGEN D. (2009). Incremental dialogue processing in a micro-domain. In *ACL*.

Construction (très) rapide de tables de traduction à partir de grands bi-textes

Li Gong[1,2] Aurélien Max[1,2] François Yvon[1]
(1) LIMSI-CNRS, Orsay, France (2) Univ. Paris Sud, Orsay, France
{prénom.nom}@limsi.fr

Résumé. Dans cet article de démonstration, nous introduisons un logiciel permettant de construire des tables de traduction de manière beaucoup plus rapide que ne le font les techniques à l'état de l'art. Cette accélération notable est obtenue par le biais d'un double échantillonnage : l'un permet la sélection d'un nombre limité de bi-phrases contenant les segments à traduire, l'autre réalise un alignement à la volée de ces bi-phrases pour extraire des exemples de traduction.

Mots-clés : traduction automatique statistique ; développement efficace ; temps de calcul.

Keywords: statistical machine translation ; efficient development ; computation time.

1 Introduction et motivations

De très grands bi-textes sont désormais disponibles pour l'apprentissage des systèmes de traduction automatique statistique. Cependant, la grande majorité des situations d'utilisation de ces systèmes n'imposera en pratique l'exploitation que d'une petite partie des données disponibles. Les approches standard, telles que celle du système à l'état de l'art `Moses` [1], reposent sur l'alignement au niveau des mots de l'intégralité des bi-textes, étape très couteuse en temps. Afin de réduire les temps de construction des tables de traduction, des travaux ont déjà proposé de recourir à un échantillonnage des exemples de traduction focalisé sur les seuls segments à traduire, permettant d'obtenir directement des tables de traduction très compactes dont la performance rivalise avec des tables apprises sur l'intégralité du bi-texte (Callison-Burch *et al.*, 2005). Toutefois, le gain en temps obtenu par cette approche ne correspond qu'à une réduction d'environ 15% du temps de traitement.

Nous présentons un logiciel permettant de considérablement modifier cette situation : non seulement les bi-phrases contenant des exemples de traduction utiles sont échantillonnées, mais l'alignement de ces bi-phrases est également construit à la demande à l'aide de la technique décrite dans (Gong *et al.*, 2013). Ce résultat peut par exemple être comparé à celui décrit dans (Zens *et al.*, 2012), où des tables de traduction sont filtrées *a posteriori*, soit en ajoutant à la procédure d'apprentissage standard un calcul (lui-même couteux) de détection de bi-segments redondants selon un critère d'entropie. Zens *et al.* (2012) décrivent par exemple, pour la paire de langue anglais-français, qu'un filtrage ne retenant que 8,1% des entrées d'une table de traduction ne mène pas à une perte supérieure à 1 point BLEU. Dans nos expériences ci-dessous, le même résultat est obtenu en n'effectuant que 6,9% du temps de calcul utilisé par un système de référence pour l'estimation des tables de traduction de développement et de test.

2 Description du logiciel

L'architecture de notre logiciel d'estimation de tables de traduction [2] est décrite sur la Figure 1. Pour l'ensemble des segments du texte à traduire, un échantillonnage aléatoire [3] (à taille constante) est effectué dans l'intégralité du bi-texte de manière efficace à l'aide d'un tableau de suffixes. Pour les bi-phrases trouvées et ne se trouvant pas déjà dans un cache d'alignements, un alignement est construit par une procédure d'alignement ciblé par échantillonnage proposée par Gong *et al.* (2013). Une fois ces phrases alignées, les comptes nécessaires au calcul d'un ensemble standard de traits pour des bi-segments sont extraits, et les tables de traduction sont finalement écrites sur disque pour être utilisé par un décodeur

1. `http://www.statmt.org/moses`
2. Nous regroupons sous le nom générique de "tables de traduction" les tables de bi-segments et tables de réordonnancement lexicalisées.
3. D'autres stratégies d'échantillonnage possibles ne seront pas décrites ici.

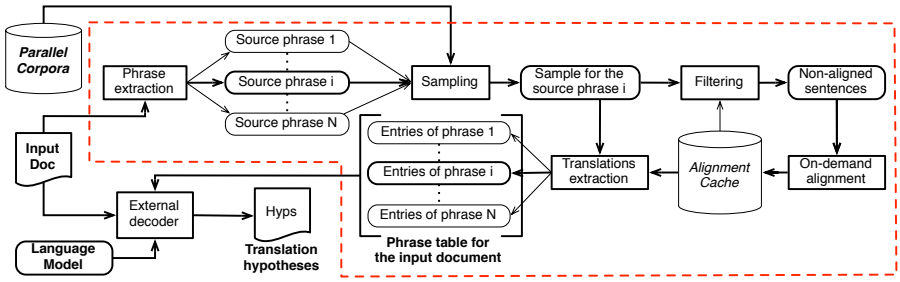


FIGURE 1 – Architecture de notre système (encadré rouge) permettant l'estimation rapide de tables de traduction.

fondé sur les segments. Le logiciel, intégralement développé en langage Java, permet une exécution *multi-thread*, et sera disponible en téléchargement libre à des fins de recherche.

3 Exemple de validation expérimentale

Systèmes		**Performance traduction**		**Performance temps**				
	α	BLEU	TER	TabSuff.	dev	test	tuning	total
`Moses`	-	34,12±0,10	48,59±0,22	-	1 212h		21h	1 233h
`otf`	1 000	33,35±0,02	49,62±0,05	2h	72h	81h	20h	175h
	500	33,05±0,10	50,03±0,11	2h	40h	44h	20h	106h
	250	32,81±0,03	49,87±0,07	2h	24h	25h	20h	71h
	100	32,52±0,08	50,49±0,15	2h	14h	14h	20h	50h
	50	32,73±0,06	49,46±0,06	2h	10h	11h	20h	43h

TABLE 1 – Résultats pour nos expériences de traduction anglais-français des documents Cochrane. La performance en temps est donnée en *user CPU time* tel que donné par la commande UNIX `time`.

Pour illustrer la performance de notre approche, nous avons exécuté les expériences suivantes. Nous avons utilisé un grand corpus de 16,6 millions de bi-phrases anglais-français de la tâche de traduction médicale de WMT'14 [4]. Nous avons choisi comme domaine d'évaluation des documents destinés à des spécialistes en médecine produit par la collaboration Cochrane [5], en utilisant 743 phrases pour le développement, et 1 800 pour l'évaluation. Nous avons suivi les procédures standard du logiciel `Moses`, utilisant notamment MERT pour l'optimisation des paramètres.

La Table 1 présente les résultats obtenus pour un système `Moses` standard, ainsi que pour différentes valeurs du paramètre d'échantillonnage (α) de calcul des scores d'association utilisé par notre procédure d'alignement (voir (Gong *et al.*, 2013)). On constate une perte en BLEU relativement au système `Moses` allant de 0,77 à 1,38 point BLEU, pour un gain en temps pour l'estimation des tables nécessaires de respectivement 87% à 98%.

Références

CALLISON-BURCH C., BANNARD C. & SCHROEDER J. (2005). Scaling phrase-based statistical machine translation to larger corpora and longer phrases. In *Proceedings of ACL*, Ann Arbor, USA.

GONG L., MAX A. & YVON F. (2013). Improving bilingual sub-sentential alignment by sampling-based transpotting. In *Proceedings of IWSLT*, Heidelberg, Germany.

ZENS R., STANTON D. & XU P. (2012). A Systematic Comparison of Phrase Table Pruning Techniques. In *Proceedings of EMNLP*, p. 972–983, Jeju Island, Korea.

4. `http://www.statmt.org/wmt14`
5. `http://www.cochrane.org`

Un assistant vocal personnalisable

Tatiana Ekeinhor-Komi [1, 3] Hajar Falih [2] Christine Chardenon [1]
Romain Laroche [2] Fabrice Lefevre [3]

(1) Orange Labs, 2 Avenue Pierre Marzin, 22300 Lannion
(2) Orange Labs, 38-40 Rue du Général Leclerc, 92130 Issy les Moulineaux
(3) LIA-CERI, Université d'Avignon, France

prenom.nomsanstiret@orange.com, fabrice.lefevre@univ-avignon.fr

Résumé. Nous proposons la démonstration d'un assistant personnel basé sur une architecture distribuée. Un portail vocal relie l'utilisateur à des applications. Celles-ci sont installées par l'utilisateur qui compose de ce fait son propre assistant personnel selon ses besoins.

Abstract. We introduce a personal assistant based on a distributed architecture. A portal connects user to applications. Applications are installed by a user who compose his own assistant according to his needs.

Mots-clés : Système de dialogue, applications du traitement automatique du langage naturel, assistant personnel.

Keywords: Dialogue system, natural language processing applications, personal assistant.

Présentation du démonstrateur

Un assistant vocal est un système de dialogue qui converse avec son utilisateur en langage naturel, afin de répondre aux divers besoins de celui-ci. Avec Google Now et Siri, les assistants vocaux se sont démocratisés dans la vie de tous les jours. Leur utilité croit avec les maisons intelligentes, objets connectés, etc. Plus que jamais, on attend de ces systèmes qu'ils soient capables de gérer des dialogues portant sur des sujets divers et variés.

Dans le but de concevoir des systèmes de dialogue multi-domaines (Hsu *et al.*, 2002; Lee *et al.*, 2012; Planells *et al.*, 2013), l'approche dominante a été d'étendre les domaines d'un système existant (Gašic *et al.*, 2013). Ceci fonctionne bien pour une extension d'un voire deux domaines. Mais cela ne permet pas de gérer la dynamique de dizaines d'applications et encore moins de réaliser l'extension automatiquement. C'est pourquoi, le modèle du démonstrateur s'inspire plutôt du modèle distribué de (Lin *et al.*, 1999) car celui-ci permet d'ajouter ou de supprimer un domaine sans perturber le fonctionnement de l'existant. Ce modèle considère le système comme un ensemble constitué d'un module central et de sous-modules correspondant aux domaines possibles du système de dialogue. Le module central assure la liaison entre l'utilisateur et les sous-modules. Dans le cas du démonstrateur, les domaines correspondent à des applications de dialogue. C'est la modularité du système qui permet la composition des applications sur demande de l'utilisateur. De plus cette personnalisation permet à l'utilisateur de n'installer que des applications qui lui seront utiles. Nous proposons la démonstration d'un assistant personnel basé sur une architecture distribuée. Un portail vocal relie l'utilisateur à des applications. Celles-ci sont installées par l'utilisateur qui compose de ce fait son propre assistant personnel selon ses besoins.

Le démonstrateur est une mise en œuvre préliminaire du modèle distribué, où chaque domaine est une application de dialogue indépendante, appelée à partir d'une application centrale, dénommée *Portail*. Le démonstrateur est donc constitué du *Portail* et d'une application spéciale : le *Store*. Il sert à installer, mettre à jour ou désinstaller des applications. A titre d'illustration trois applications sont actuellement proposées sur le *Store* : *MeteoOrange* est une application de météo qui se connecte au site de météo d'Orange ; *NewsAFP* est une application d'actualités catégorisée en plusieurs thèmes (politique, sport, affaires), il se connecte au site de l'AFP (Agence France-Presse) ; *ShoppingList* est une application qui permet de gérer sa liste de course. A chaque nouvelle mise à jour (installation, désinstallation), le portail doit intégrer les données sémantiques fournies par l'application. Ces données permettent au portail d'interpréter une requête et de sélectionner

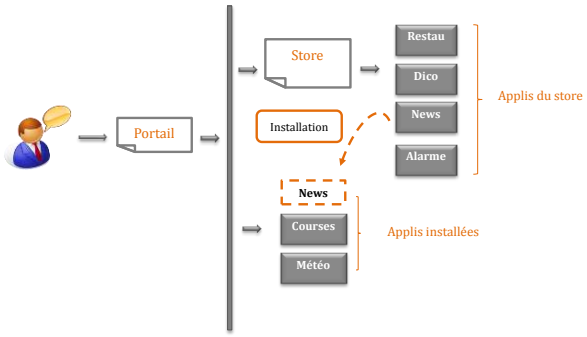


FIGURE 1 – Architecture générale

l'application correspondante. L'architecture est présentée dans la figure 1. Les applications ont été développées avec l'outil interne d'Orange : Disserto (Laroche *et al.*, 2010). Une interface web a été développée pour réaliser la démonstration. L'API de Google est utilisée pour la reconnaissance vocale, la transcription textuelle et la synthèse vocale. Le tout est déployé sur un serveur d'application.

Exemple de dialogue démontrant l'installation à la volée d'une application (comme illustré dans la figure 1)

U : Quels sont les actualités politiques du jour ?
S : Désolé, nous ne pouvons donner suite à votre demande. Voulez-vous rechercher sur le Store une application à installer ?
U : Oui.
S : Voulez-vous faire une recherche par nom ou par description ?
U : Par nom.
S : Donnez le nom de l'application à installer
U : Je veux installer NewsAFP.
S : Confirmez-vous l'installation de NewsAFP ?
U : Oui.
S : L'application NewsAFP est désormais installée sur votre portail.
U : Quelles sont les actualités politiques ?
S : [Résulats]

Ce sujet embrasse les problématiques de conception de systèmes de dialogue multi-domaines par l'intégration de systèmes existants d'une part mais aussi de la gestion de dialogues complexes d'autre part. L'étape suivante de notre travail sera de permettre la transition cohérente entre les différentes applications au cours de l'interaction ce qui supposera d'améliorer les algorithmes de fusion et de désambiguïsation qui sont implantés pour le moment sous une forme simple dans un modèle à base de règles.

Références

GAŠIC M., BRESLIN C., HENDERSON M., KIM D., SZUMMER M., THOMSON B., TSIAKOULIS P. & YOUNG S. (2013). Pomdp-based dialogue manager adaptation to extended domains.

HSU W.-T., WANG H.-M. & LIN Y.-C. (2002). The design of a multi-domain chinese dialogue system. In *International Symposium on Chinese Spoken Language Processing*.

LAROCHE R., BRETIER P. & PUTOIS G. (2010). Enhanced monitoring tools and online dialogue optimisation merged into a new spoken dialogue system design experience. In *Proceedings of Interspeech*, Chiba (Japan).

LEE S. R. D. L. I., LEE S. H. G. G. & KIM M. K. K. (2012). A hierarchical domain model-based multi-domain selection framework for multi-domain dialog systems.

LIN B.-S., WANG H.-M. & LEE L.-S. (1999). A distributed architecture for cooperative spoken dialogue agents with coherent dialogue state and history. In *Asru*, volume 99, p.4̃ : Citeseer.

PLANELLS J., HURTADO L.-F., SEGARRA E. & SANCHIS E. (2013). A multi-domain dialog system to integrate heterogeneous spoken dialog systems.

Mesurer la similarité structurelle entre réseaux lexicaux

Bruno Gaume[1] Emmanuel Navarro[2] Yann Desalle[3] Benoît Gaillard[1]
(1) CLLE-ERSS, CNRS, Universié de Toulouse
(2) IRIT, CNRS, Université de Toulouse
(3) ATILF, CNRS, Université de Lorraine
gaume@univ-tlse2.fr, navarro@irit.fr, yann.desalle@gmail.com, benoit.gd@gmail.com,

Résumé. Dans cet article, nous comparons la structure topologique des réseaux lexicaux avec une méthode fondée sur des marches aléatoires. Au lieu de caractériser les paires de sommets selon un critère binaire de connectivité, nous mesurons leur proximité structurelle par la probabilité relative d'atteindre un sommet depuis l'autre par une courte marche aléatoire. Parce que cette proximité rapproche les sommets d'une même zone dense en arêtes, elle permet de comparer la structure topologique des réseaux lexicaux.

Abstract. In this paper, we compare the topological structure of lexical networks with a method based on random walks. Instead of characterising pairs of vertices according only to whether they are connected or not, we measure their structural proximity by evaluating the relative probability of reaching one vertex from the other via a short random walk. This proximity between vertices is the basis on which we can compare the topological structure of lexical networks because it outlines the similar dense zones of the graphs.

Mots-clés : Réseaux lexicaux, réseaux petits mondes, comparaison de graphes, marches aléatoires.

Keywords: Lexical networks, small worlds, comparison graphs, random walks.

1 Contexte

Une ressource lexicale peut être modélisée sous la forme d'un graphe $G = (V, E)$ dans lequel un ensemble de n sommets V représente des entités lexicales (lemmes, contextes syntaxiques ...) et un ensemble de m arêtes $E \subseteq \mathbf{P}_2^V$ représente une relation lexicale entre ces entités. Un des problèmes majeurs concernant ces réseaux lexicaux porte sur leurs désaccords apparents : par exemple, si $G_1 = (V, E_1)$ et $G_2 = (V, E_2)$ sont deux graphes de synonymie standards d'une langue donnée, alors une grande proportion de paires $\{x, y\} \in \mathbf{P}_2^V$ sont liées dans G_1 ($\{x, y\} \in E_1$) mais ne le sont pas dans G_2 ($\{x, y\} \notin E_2$) ; autrement dit, x et y sont synonymes pour G_1 mais ne le sont pas pour $G2$. Un tel désaccord n'est pas compatible avec l'hypothèse d'une synonymie qui reflèterait la structure sémantique du lexique commune aux membres d'une même communauté linguistique.

Pour résoudre cette contradiction apparente, il faut regarder les réseaux lexicaux dans une perspective plus large. La figure 1 est un exemple artificiel de désaccord généralisé entre les arêtes de deux graphes malgré une similarité structurelle. Bien qu'ils n'aient aucune arête en commun ($E_1 \cap E_2 = \varnothing$), ces deux graphes se ressemblent parce que les deux zones denses dessinées par chacun des graphes contiennent les mêmes sommets : $\{1, 2, 3, 4, 11, 12, 13\}$ et $\{4, 5, 6, 7, 8, 9, 10\}$. Cette similarité structurelle est observable en considérant chacun des graphes comme un tout, et non en les comparant arête par arête. Les zones denses de cet exemple artificiel (fig. 1) sont caractéristiques des *graphes de terrain* [1] qui, pour la plupart, sont des réseaux petits mondes hiérarchiques (RPMH) partageant les mêmes propriétés (Newman, 2003; Gaume *et al.*, 2010; Steyvers & Tenenbaum, 2005). Ils présentent une **faible densité en arêtes** (peu d'arêtes par rapport au nombre maximal d'arêtes potentielles), des **chemins courts** (le nombre moyen d'arêtes L sur les plus courts chemins entre deux sommets est faible), un fort **taux d'agrégation** C (des sous-graphes localement denses en arêtes, ou agrégats, peuvent être identifiés alors que le graphe est globalement peu dense en arêtes (Watts & Strogatz, 1998)), et la distribution des degrés d'incidence de leurs sommets approche une **loi de puissance** (Albert & Barabasi, 2002). Nous montrons dans la section 2 que les réseaux lexicaux étudiés dans cet article possèdent les caractéristiques des RPMH. Ainsi, comme le suggère la figure 1, un désaccord apparent entre les réseaux lexicaux n'implique pas nécessairement une incompatibilité structurelle des données modélisées.

Dans cet article, nous étudions les réseaux lexicaux avec une méthode fondée sur des marches aléatoires. Au lieu de caractériser les paires de sommets selon leur seule connectivité binaire (existence ou absence d'une arête entre les som-

1. Les graphes de terrain sont des graphes qui modélisent les données réelles récoltées sur le terrain, par exemple en sociologie, linguistique ou biologie. Ils s'opposent en cela aux graphes artificiels (déterministes ou aléatoires).

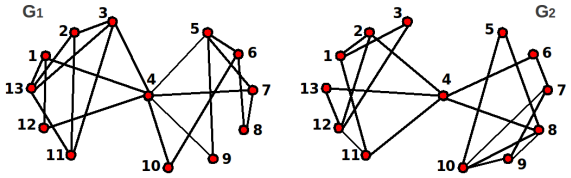


FIGURE 1 – Contradiction entre la variaribilité locale et la similarité globale.

mets), nous mesurons leur proximité structurelle par la probabilité relative d'atteindre un sommet depuis l'autre par une courte marche aléatoire [2]. Parce que cette proximité rapproche les sommets d'une même zone dense en arêtes, elle permet de mesurer la qualité de la divergence de surface entre deux réseaux lexicaux. Notons que ce travail vient à la suite de (Gaillard *et al.*, 2011; Navarro *et al.*, 2012) et de (Navarro, 2013, chap. 3).

Nous montrons dans la section 2 les limites des approches arête-par-arête pour l'analyse et la comparaison des réseaux lexicaux selon lesquelles les réseaux de synonymie d'une même langue seraient significativement différents. Dans la section 3, nous présentons une méthode de comparaison structurelle de graphes basée sur la *confluence*, mesure de proximité entre sommets qui repose sur les marches aléatoires et permet d'analyser structurellement les réseaux lexicaux. En section 4, nous appliquons cette méthode de comparaison de graphes à des ressources construites par des lexicographes et par les foules (crowdsourcing), ressources dont les méthodes d'élaboration diffèrent mais qui tentent de décrire la même relation lexicale : la synonymie. Nous concluons en section 5.

2 Compararer $\mathbf{G_1} = (\mathbf{V}, \mathbf{E_1})$ et $\mathbf{G_2} = (\mathbf{V}, \mathbf{E_2})$ en comparant les ensembles $\mathbf{E_1}$ et $\mathbf{E_2}$ comme des «sacs de liens» sans structures

Nous illustrons notre propos dans cette section sur la comparaison de deux ressources lexicales, toutes deux construites par des lexicographes approximativement à la même époque pour représenter la même relation de synonymie :

– **Rob** $= (\mathbf{V_{Rob}}, \mathbf{E_{Rob}})$: Le dictionnaire Le Robert (Robert & Rey, 1985) a été informatisé au cours d'un parteneriat IBM / ATILF [3]. Cette ressource électronique liste les synonymes des différentes acceptions des vocables du français. Les sommets du graphe lexical *Rob* qui a été construit à partir de cette ressource sont les vocables (les vocables homonymes ne sont pas distingués et sont représentés par un même sommet). La paire $\{x, y\}$ appartient à E_{Rob} si et seulement si une des acceptions de x a été considérée comme synonyme d'une des acceptions de y par l'équipe lexicographique du Robert. Par exemple, le verbe *causer* est à la fois synonyme de *parler* et de *engendrer*.
– **Lar** $= (\mathbf{V_{Lar}}, \mathbf{E_{Lar}})$: Le graphe lexical *Lar* a été construit à partir du dictionnaire Larousse (Guilbert *et al.*, 1971 1978) de la même manière que le graphe *Rob*.

Les caractéristiques (que nous appelons « pédigrés ») des graphes *Rob* et *Lar* sont fournis dans le tableau 1. Ces mesures sont en accord avec la plupart des études sur les réseaux lexicaux (Motter *et al.*, 2002; de Jesus Holanda *et al.*, 2004; Gaume, 2004) qui montrent que les réseaux lexicaux comme la majorité des réseaux de terrains sont des RPMH typiques.

TABLE 1 – Pédigrés des graphes lexicaux *Lar* et *Rob* : n et m sont les nombres de sommets et d'arêtes, $\langle k \rangle$ est la moyenne des degrés d'incidence des sommets, C est le coefficience d'agrégation du graphe, L_{lcc} est la moyenne des plus courts chemins entre tous les nœuds de la plus grande partie connexe (sous-graphe dans lequel il existe au moins un chemin entre deux noeuds quelconques de ce sous-graphe), r^2 est le coefficient de corrélation entre la distribution des degrés d'incidence et la loi de puissance la plus fortement corrélée à cette distribution, λ est la puissance de cette loi.

Réseaux Lexicaux	**n**	**m**	$\langle \mathbf{k} \rangle$	**C**	$\mathbf{L_{lcc}}$	λ ($\mathbf{r^2}$)
Lar	22066	73091	6,62	0,19	6,36	-2,43 (0,90)
Rob	38147	99998	5,24	0,12	6,37	-2,43 (0,94)

2. Notons que cette méthode de mesure de proximité entre sommets dans un graphe peut-être utilisée avantageusement pour la modélisation des graphes de terrain (Gaume *et al.*, 2010), sur des tâches de substitution lexicale ou de résolution de métaphore (Desalle *et al.*, 2014b, 2009; Desalle, 2012), pour l'enrichissement de ressources lexicales (Sajous *et al.*, 2011), la navigation dans les réseaux de terrain (Gaume, 2008), la recherche d'informations (Navarro *et al.*, 2011) ou encore pour la détection de pathologies (Desalle *et al.*, 2014a).

3. `http://www.atilf.fr`

2.1 Distance d'édition

Soit deux graphes $G_1 = (V_1, E_1)$ et $G_2 = (V_2, E_2)$, nous mesurons la similarité des couvertures lexicales de G_1 et G_2 par l'indice de *Jaccard* : $J(G_1, G_2) = \frac{|V_1 \cap V_2|}{|V_1 \cup V_2|}$. Nous avons alors $J(Rob, Lar) = 0,49$. Ces deux graphes ont donc une couverture lexicale commune suffisament large pour que la comparaison entre les jugements de synonymie qu'ils modélisent soit réalisée sur cette couverture lexicale commune : $V_1 \cap V_2$.

Pour mesurer l'accord entre les arêtes de G_1 et de G_2, nous commençons donc par réduire les deux graphes à leurs sommets communs : $G'_1 = \big(V' = (V_1 \cap V_2), E'_1 = E_1 \cap (V' \times V')\big)$ et $G'_2 = \big(V' = (V_1 \cap V_2), E'_2 = E_2 \cap (V' \times V')\big)$. Pour chaque paire de sommets $\{a,b\} \in (V' \times V')$, quatre configurations sont possibles :

- $\{a,b\} \in E'_1 \bigcap E'_2$: accord sur la paire $\{a,b\}$, a et b sont synonymes dans G'_1 et dans G'_2 ;
- $\{a,b\} \in \overline{E'_1} \bigcap \overline{E'_2}$: accord sur la paire $\{a,b\}$, a et b ne sont synonymes ni dans G'_1 ni dans G'_2 ;
- $\{a,b\} \in E'_1 \bigcap \overline{E'_2}$: désaccord sur la paire $\{a,b\}$, a et b sont synonymes dans G'_1 mais pas dans G'_2 ;
- $\{a,b\} \in \overline{E'_1} \bigcap E'_2$: désaccord sur la paire $\{a,b\}$, a et b sont synonymes dans G'_2 mais pas dans G'_1 ;

Une longue tradition dans la recherche sur la comparaison de graphes consiste à déterminer si deux graphes sont isomorphes. Deux graphes $G_1 = (V_1, E_1)$ et $G_2 = (V_2, E_2)$ sont isomorphes s'il existe une fonction bijective $f : V_1 \mapsto V_2$ telle que, pour toute paire de sommets $\{u,v\} \in \mathbf{P}_2^V$, $\{u,v\} \in E_1 \Leftrightarrow \big\{f(u), f(v)\big\} \in E_2$. La comparaison entre graphes consiste alors à rechercher de tels isomorphismes. Dans les graphes étudiés dans cet article, les nœuds sont étiquetés et ne peuvent correspondre que s'ils ont les mêmes étiquettes : la seule bijection possible est donc la fonction identité. Ainsi, pour savoir si deux graphes étiquetés $G_1 = (V, E_1)$ et $G_2 = (V, E_2)$ sont isomorphes, il suffit de vérifier que $E_1 = E_2$.

Une telle similarité est très basique : si aucune arête ne diffère alors les deux graphes sont similaires, sinon ils sont différents (ils ne sont pas isomorphes). Afin d'assouplir cette approche de l'isomorphisme pour fournir une mesure quantitative continue de la différence entre deux graphes, plusieurs alternatives ont été proposées (pour une revue de ces méthodes, voir par exemple (Gao *et al.*, 2010)). Ces méthodes s'inspirent de la distance d'édition entre deux chaînes de caractères (Levenshtein, 1966). La distance d'édition entre entre deux graphes $G_1 = (V_1, E_1)$ et $G_2 = (V_2, E_2)$ est définie par la série d'opérations la plus économique pour transformer G_1 en un isomorphisme de G_2. Habituellement, l'ensemble des opérations possibles ne contient que l'insertion, la suppression et la substitution de sommets et d'arêtes. Cet ensemble peut éventuellement être étendu selon les données que les graphes modélisent. Par exemple, dans le cas d'une segmentation d'image, (Ambauen *et al.*, 2003) introduisent les opérations de cission et de fusion de noeuds.

Dans le cadre de cet article, puisque après la réduction des deux graphes à leurs sommets communs, nous aurons $V_1 = V_2 = V$, les seules opérations possibles vont être la suppression et l'insertion d'arêtes. Si le coût d'édition d'une arête est 1, alors la distance d'édition entre G_1 et G_2 est :

$$ED = |E_1 \cap \overline{E_2}| + |E_2 \cap \overline{E_1}| \tag{1}$$

Remarquons que $ED \in [0, |E_1| + |E_2|]$. Cette mesure de dissimilarité ne prend pas en compte le nombre d'arêtes de G_1 et de G_2. Editer dix arêtes pour rendre deux graphes de quinze arêtes isomorphes n'est pas la même chose qu'éditer dix arêtes pour rendre deux graphes de quinze mille arêtes isomorphes. Cette distance d'édition doit donc être normalisée :

$$GED(G_1, G_2) = \frac{|E_1 \cap \overline{E_2}| + |E_2 \cap \overline{E_1}|}{|E_1| + |E_2|} \tag{2}$$

Maintenant, $GED(G_1, G_2) \in [0, 1]$. Appliquée à Lar'/Rob', $GED(Lar', Rob') = 0,47$. Ce résultat montre que Lar' et Rob' sont dissemblables : les dictionnaires Larousse et Le Robert n'ont qu'un faible accord sur les paires de lexèmes qu'ils jugent synonymes. Ceci peut s'expliquer par le fait que la projection de la notion graduelle de quasi-synonymie sur des jugements binaires de synonymie offre une large place à l'interprétation, même si les juges sont des lexicographes experts comme pour les dictionnaires Larousse et Robert. En fait, on observe souvent un faible accord entre des ressources qui décrivent la même réalité linguistique mais qui sont construites indépendamment, même lorsqu'elles reposent sur des jugements humains qui suivent un même protocole (Murray & Green, 2004).

3 Comparer $\mathbf{G_1 = (V, E_1)}$ et $\mathbf{G_2 = (V, E_2)}$ en comparant la structure engendrée par $\mathbf{E_1}$ sur V à la structure engendrée par $\mathbf{E_2}$ sur V

GED est une mesure quantitative de surface qui analyse les graphes comme des «sacs de liens» sans structure. En comparant les graphes arête par arête, elle ne tient pas compte de la structure globale profonde des graphes bien que celle-ci

soit très spécifique puisqu'il s'agit de RPMH. La présence ou l'absence d'une arête entre deux sommets est un jugement de synonymie qui peut être confirmé ou infirmé par la structure topologique du graphe autour de ces sommets. Dans cette section, nous décrivons une mesure quantitative de la similarité structurelle entre graphes. Cette mesure est basée sur les marches aléatoires, ce qui nous permet d'enrichir l'information sur les paires de sommets par une mesure de proximité structurelle entre sommets : *la confluence*.

3.1 Confluence

Soit $G = (V,E)$ un graphe réflexif[4] et non dirigé, définissons $d_G(u) = |\{v \in V / \{u,v\} \in E\}|$ le degré d'incidence d'un sommet u dans le graphe G et imaginons un marcheur se déplaçant sur le graphe G : au temps $t \in \mathbb{N}$, le marcheur est sur un sommet $u \in V$; au temps $t+1$, le marcheur peut atteindre n'importe quel voisin de u avec un probabilité uniforme.

Ce processus est une simple marche aléatoire (Bollobas, 2002; Kinouchi *et al.*, 2002; Baronchelli *et al.*, 2013). Il peut être défini par une chaîne de Markov sur V à l'aide d'une matrice de transition $[G]$:

$$[G] = (g_{u,v})_{u,v \in V} \quad \text{avec} \quad g_{u,v} = \begin{cases} \dfrac{1}{d_G(u)} \text{ si } \{u,v\} \in E, \\ 0 \text{ sinon.} \end{cases}$$

Puisque G est réflexif, chaque sommet a au moins un voisin (lui-même) ; G est donc bien définie. De plus, par construction, $[G]$ est une matrice stochastique : $\forall u \in V, \sum_{v \in V} g_{u,v} = 1$. La probabilité $P^t_G(u \rightsquigarrow v)$ qu'un marcheur démarrant sur le sommet u atteigne le sommet v après t pas est :

$$P^t_G(u \rightsquigarrow v) = ([G]^t)_{u,v} \tag{3}$$

On peut alors prouver (Gaume, 2004) à l'aide du théorème de Perron-Frobenius (Stewart, 1994) que si G est connexe, réflexif et non-dirigé, alors $\forall\, u,v \in V$:

$$\lim_{t \to \infty} P^t_G(u \rightsquigarrow v) = \lim_{t \to \infty} ([G]^t)_{u,v} = \frac{d_G(v)}{\sum_{x \in V} d_G(x)} = \pi_G(v) \tag{4}$$

Cela signifie que quand t tend vers l'infini, la probabilité d'être sur un sommet v au temps t ne dépend pas du sommet de départ mais seulement du degré d'incidence de v. Nous noterons cette limite $\pi_G(v)$ dans la suite.

Par contre, la dynamique de convergence vers cette limite (équation (4)) dépend fortement du sommet de départ. En effet, la trajectoire du marcheur est totalement régie par la topologie du graphe autour de ce sommet de départ : après t pas, tout sommet v situé à une distance de t arêtes (ou moins) peut être atteint. La probablité de cet événement dépend du nombre de chemins entre u et v et de la structure du graphe autour des sommets intermédiaires le long de ces chemins. Plus il y a de chemins courts entre les sommets u et v, plus la probibilité d'atteindre v à partir de u est grande. Par exemple, si l'on prend $G_1 = Rob$ et $G_2 = Lar$ et que l'on choisit les trois sommets $u = éplucher$, $r = dépecer$ et $s = sonner$ tels que :

- u et r sont jugés synonymes dans Rob : $\{u,r\} \in E_1$;
- u et r ne sont pas jugés synonymes dans Lar : $\{u,r\} \notin E_2$;
- r et s ont le même nombre de synonymes dans G_1 : $d_{G_1}(r) = d_{G_1}(s) = d_1$;
- r et s ont le même nombre de synonymes dans G_2 : $d_{G_2}(r) = d_{G_2}(s) = d_2$.

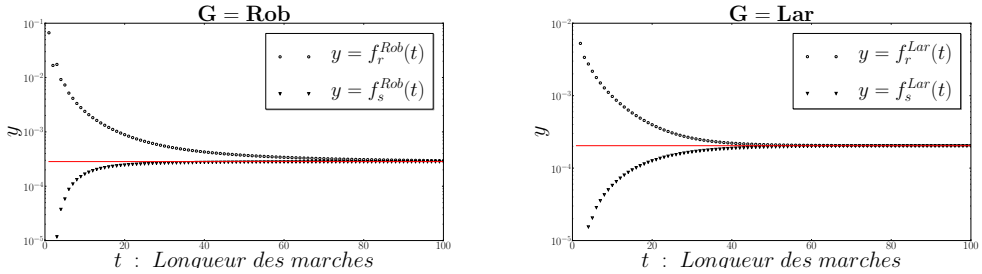
Alors, d'après l'équation (4), les deux séries $(P^t_{G_1}(u \rightsquigarrow r))_{1 \leq t}$ et $(P^t_{G_1}(u \rightsquigarrow s))_{1 \leq t}$ convergent vers la même limite : $\pi_{G_1}(r) = \pi_{G_1}(s) = \frac{d_1}{\sum_{x \in V_1} d_{G_1}(x)}$ tout comme les deux séries $(P^t_{G_2}(u \rightsquigarrow r))_{1 \leq t}$ et $(P^t_{G_2}(u \rightsquigarrow s))_{1 \leq t}$: $\pi_{G_2}(r) = \pi_{G_2}(s) = \frac{d_2}{\sum_{x \in V_2} d_{G_2}(x)}$. Cependant, les deux séries ne convergent pas selon la même dynamique. Au début de la marche, avec t petit, on peut s'attendre à ce que $P^t_{G_1}(u \rightsquigarrow r) > P^t_{G_1}(u \rightsquigarrow s)$ et $P^t_{G_2}(u \rightsquigarrow r) > P^t_{G_2}(u \rightsquigarrow s)$ puisque *éplucher* est sémantiquement plus proche de *dépecer* que de *sonner*. En effet, le nombre de chemins courts entre *éplucher* et *dépecer* est plus grand qu'entre *éplucher* et *sonner*.

La figure 2(a) présente les valeurs de $P^t_{Rob}(u \rightsquigarrow r)$ et de $P^t_{Rob}(u \rightsquigarrow s)$ en fonction de t et les compare à leur limite commune. La figure 2(b) présente ces mêmes valeurs calculées sur Lar. Ces figures confirment notre hypothèse : puisque *éplucher* et *dépecer* sont sémantiquement proches, $P^t_{Rob}(u \rightsquigarrow r)$ et $P^t_{Lar}(u \rightsquigarrow r)$ décroissent vers leurs limites même si r et s ne sont pas synonymes (comme c'est le cas dans Lar).

En fait, la limite $\pi_G(v)$ ne fournit pas d'information sur la proximité entre u et v dans le graphe ; au contraire, elle la masque par la seule prise en compte de v dans son calcul. Nous définissons donc la t-confluence $CONF^t_G(u,v)$ entre deux sommets u et v sur un graphe G comme suit :

$$CONF^t_G(u,v) = \frac{P^t_G(u \rightsquigarrow v)}{P^t_G(u \rightsquigarrow v) + \pi_G(v)} \tag{5}$$

4. C'est-à-dire que chaque sommet est connecté à lui-même. Si ce n'est pas le cas, on peut généralement en ajouter sans perte d'information.



(a) $u =$ **éplucher**, $r =$ *dépecer*, $s =$ *sonner*, $f_r^{Rob}(t) = P_{Rob}^t(u \rightsquigarrow r)$, $f_s^{Rob}(t) = P_{Rob}^t(u \rightsquigarrow s)$

(b) $u =$ **éplucher**, $r =$ *dépecer*, $s =$ *sonner*, $f_r^{Lar}(t) = P_{Lar}^t(u \rightsquigarrow r)$, $f_s^{Lar}(t) = P_{Lar}^t(u \rightsquigarrow s)$

FIGURE 2 – Les différentes dynamiques de convergence de la série $(P_G^t(u \rightsquigarrow v))_{1<t}$ vers sa limite pour trois types de relations entre u et v : **(1)** $\mathbf{f_r^{Rob}(t)}$: u et v sont synonymes comme *éplucher* et *dépecer* dans *Rob* ; **(2)** $\mathbf{f_s^{Rob}(t)}$ et $\mathbf{f_s^{Lar}(t)}$: u et v ne sont pas synonymes et sont sémantiquement éloignés comme *éplucher* et *sonner* dans *Rob* et dans *Lar* ; (3) $\mathbf{f_r^{Lar}(t)}$: u et v ne sont pas synonymes mais sont sémantiquement proches comme *dépecer* et *éplucher* dans *Lar*.

$CONF_G^t$ définit une famille de mesures symétriques de proximité entre sommets, une mesure pour chaque longueur de marche t. Par souci de clarté, nous choisissons un t unique pour la suite de l'article. Ce choix est fait en considérant que :
- **Si t est trop grand :** $\forall u_1, v_1, u_2, v_2 \in V,\ CONF_G^t(u_1, v_1) \approx CONF_G^t(u_2, v_2) \approx 0,5$. La mesure $CONF_G^t(u,v)$ n'indique donc pas si les sommets u et v appartiennent ou non à une même zone dense en arêtes de G ;
- **Si t est trop petit :** pour toute paire $\{u,v\}$ telle que la longueur du chemin le plus court entre u et v dans G est plus grande que t, $P_G^t(u \rightsquigarrow v) = 0$ donc $CONF_G^t(u,v) = 0$. Cette mesure n'indique donc pas non plus si les sommets u et v appartiennent ou non à une même zone dense en arêtes G.

C'est pourquoi, dans la suite de cet article, t est fixé [5] à $t = 5$ et $CONF_G = CONF_G^5$.

$CONF_G$ est une mesure de proximité normalisée basée sur les marches aléatoires dans G :
- S'il existe, entre u et v, beaucoup plus de chemins courts qu'entre un sommet quelconque et v (u et v appartiennent à une même zone sur-dense en arêtes) : $P_G^5(u \rightsquigarrow v) > \pi_G(v)$ et donc $CONF_G(u,v) > 0,5$;
- S'il existe, entre u et v, autant de chemins courts qu'entre un sommet quelconque et v : $P_G^5(u \rightsquigarrow v) \approx \pi_G(v)$ et donc $CONF_G(u,v) \approx 0,5$;
- Si il existe, entre u et v, beaucoup moins de chemins courts qu'entre un sommet quelconque et v : $P_G^5(\rightsquigarrow v) < \pi_G(v)$ et donc $CONF_G(u,v) < 0,5$.

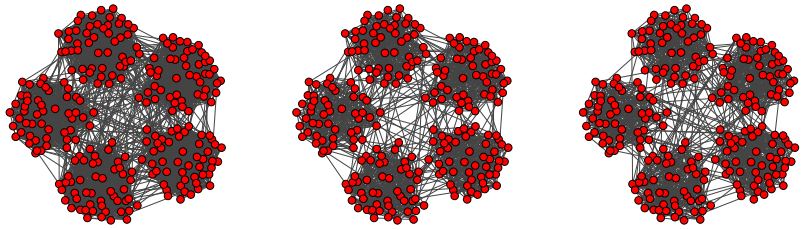
Nous considérons donc comme « proche » toute paire de sommets $\{u,v\}$ telle que la confluence $CONF_G(u,v)$ est plus grande que $0,5$. En d'autres termes, u et v sont proches si la probabilité d'atteindre v à partir de u après une marche aléatoire de cinq pas est plus grande que la probabilité d'être sur v après une marche infinie.

3.2 Une expérimentation contrôlée à l'aide de graphes artificiels

Nous avons construit artificiellement deux types de paire de graphes à comparer :
- **Deux graphes avec 5 zones denses :** nous avons d'abord construit un graphe $G_a = (V, E_a)$ tel que V est l'union de $k = 5$ ensembles $\Delta_1, \ldots \Delta_5$ de $n = 50$ sommets chacun [6] ; les arêtes de E_a ont été placées aléatoirement entre deux sommets u et v à partir de deux probabilités différentes : une probabilité $p_1 = 0,5$ entre deux sommets d'un même ensemble ($u, v \in \Delta_i$), et $p_2 = 0,01$ entre deux sommets appartenant à deux ensembles distincts ($u \in \Delta_i,\ v \in \Delta_j,\ i \neq j$). Nous avons ensuite construit un second graphe $G_b = (V, E_b)$ en choisissant aléatoirement la moitié des arêtes de G_a, et un troisième graphe $G_c = (V, E_c)$ tel que $E_c = E_a \setminus E_b$. Ces trois graphes sont représentés dans la figure 3. Bien que G_b et G_c n'aient aucune arêtes en commun, ($E_b \cap E_c = \varnothing$), G_b et G_c présentent tous deux cinq zones locales denses identiques : $\Delta_1, \ldots \Delta_5$.
- **Deux graphes aléatoires :** nous avons d'abord construit un graphe aléatoire $G_a^R = (V, E_a^R)$ tel que $|E_a^R| = |E_a|$. Nous avons ensuite construit un deuxième graphe $G_b^R = (V, E_b^R)$ en choisissant aléatoirement la moitié des arêtes de G_a^R, et un troisième graphe $G_c^R = (V, E_c^R)$ tel que $E_c^R = E_a^R \setminus E_b^R$. Ni G_b^R ni G_c^R n'ont de zones denses.
- Puisque $E_b \cap E_c = \varnothing$, $E_b \cap \overline{E_c} = E_b$ et $E_c \cap \overline{E_b} = E_c$, donc $GED(G_b, G_c) = \frac{|E_b \cap \overline{E_c}| + |E_c \cap \overline{E_b}|}{|E_b| + |E_c|} = \frac{|E_b| + |E_c|}{|E_b| + |E_c|} = 1$. Ce résultat

5. Avec $t = 5$ nous restons en général proche de la longueur moyenne des plus courts chemins dans les réseaux lexicaux (Motter *et al.*, 2002; Gaume, 2004; de Jesus Holanda *et al.*, 2004). Notons aussi qu'un t petit est favorable à complexité des algorithmes : avec $t = 5$, tous les calculs de confluence (calculs exactes en Python avec un processeur i7) nécessaires pour chacune des paires de graphes analysées dans ce papier ne nécessitent que quelques secondes. Cette approche peut être appliquée à de très grands graphes. Quand les graphes deviennent trop grands, on peut utiliser les méthodes de Monte Carlo qui sont efficaces sur les RPMH pourvu que le degré maximal des sommets soit borné.

6. Si $i \neq j$ alors $\Delta_i \cap \Delta_j = \varnothing$.



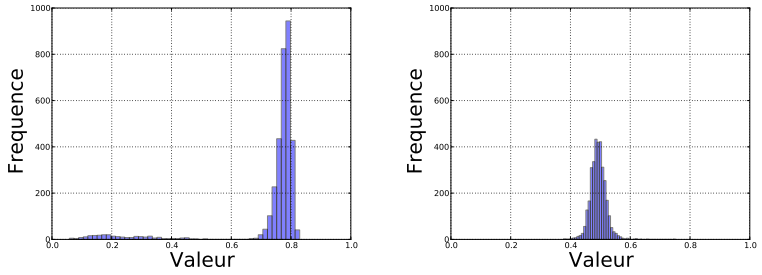
(a) $G_a = (V, E_a)$ (b) $G_b = (V, E_b)$ (c) $G_c = (V, E_c)$

FIGURE 3 – Graphe artificiel avec 5 zones locales denses identiques.

signifierait que ces graphes seraient complètement dissemblables, ce qui est vrai dans le sens où ils n'ont aucune arête en commun mais clairement faux du point de vue de l'« organisation » topologique qu'ils partagent. En effet si deux sommets appartiennent à la même zone relativement dense dans le premier graphe, ils appartiennent également à la même zone relativement dense dans le second.

– Puisque $E_b^R \cap E_c^R = \varnothing$, $E_b^R \cap \overline{E_c^R} = E_b^R$ et $E_c^R \cap \overline{E_b^R} = E_c^R$. Donc, $GED(G_b^R, G_c^R) = \frac{|E_b^R \cap \overline{E_c^R}| + |E_c^R \cap \overline{E_b^R}|}{|E_b^R| + |E_c^R|} = \frac{|E_b^R| + |E_c^R|}{|E_b^R| + |E_c^R|} = 1$.

Toutes les mesures quantitatives de surface comme *GED*, qui ne reposent que sur le décompte du nombre de désaccords, ont le désavantage de ne comparer les graphes que comme des « sacs de liens », étant ainsi insensibles aux contextes topologiques. Mais si nous comparons les distributions de la confluence des arêtes en désaccord dans G_b vs G_c d'un côté (fig. 4(a)), et dans G_b^R vs G_c^R de l'autre côté (fig. 4(b)), la différence est frappante.



(a) G_b/G_c (b) G_b^R/G_c^R

FIGURE 4 – Histogramme de l'ensemble $\{CONF_{G_c}(\{u,v\})$ tel que $\{u,v\} \in (E_b \cap \overline{E_c})\} \bigcup \{CONF_{G_b}(\{u,v\})$ tel que $\{u,v\} \in (E_c \cap \overline{E_b})\}$, en parallèle à l'histogramme de l'ensemble $\{CONF_{G_c^R}(\{u,v\})$ tel que $\{u,v\} \in (E_b^R \cap \overline{E_c^R})\} \bigcup \{CONF_{G_b^R}(\{u,v\})$ tel que $\{u,v\} \in (E_c^R \cap \overline{E_b^R})\}$.

Nous définissons donc $\mu(G_1, G_2)$, une mesure de la similarité structurelle entre les arêtes de G_1 et de G_2 :

$$\mu(G_1, G_2) = \frac{1}{|E_1 \cap \overline{E_2}| + |E_2 \cap \overline{E_1}|} \Big(\sum_{\{u,v\} \in (E_2 \cap \overline{E_1})} CONF_{G_1}(\{u,v\}) + \sum_{\{u,v\} \in (E_1 \cap \overline{E_2})} CONF_{G_2}(\{u,v\}) \Big) \quad (6)$$

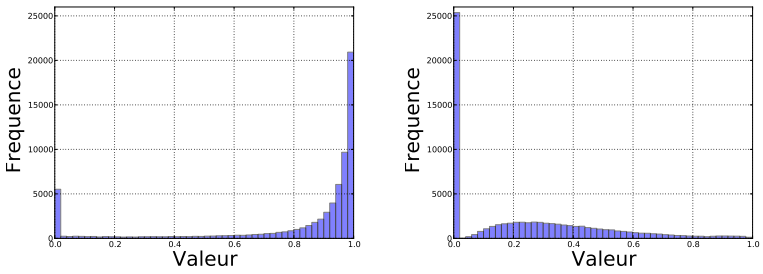
Grâce à la confluence, μ mesure le niveau de proximité structurelle dans G_1 entre les sommets des arêtes directement présentes dans G_2 et absentes de G_1, et le niveau de proximité structurelle dans G_2 entre les sommets des arêtes directement présentes dans G_1 et absentes de G_2.

Bien que $GED(G_b, G_c) = GED(G_b^R, G_c^R) = 1$, avec μ, nous pouvons maintenant voir la différence : sur cinquante réalisations $\mu(G_b, G_c) = 0,74$ (avec un écart type $std < 0,005$) alors que $\mu(G_b^R, G_c^R) = 0,49$ ($std < 0,005$). La différence entre G_b/G_c et G_b^R/G_c^R est identique quantitativement mais différente structurellement.

4 Applications sur les Réseaux lexicaux

Nous commençons par examiner la distribution de la confluence des arêtes contradictoires entre $Lar' = (V', E_{Lar'})$ vs $Rob' = (V', E_{Rob'})$. Nous la comparons à la distribution de la confluence des arêtes contradictoires entre les paires de *graphes aléatoires équivalents* $Lar'^R = (V', E_{Lar'}^R)$ et $Rob'^R = (V', E_{Rob'}^R)$ construits tels que :

$$|E_{Lar'}^R \cap E_{Rob'}^R| = |E_{Lar'} \cap E_{Rob'}|, \quad |E_{Lar'}^R \cap \overline{E_{Rob'}^R}| = |E_{Lar'} \cap \overline{E_{Rob'}}|, \quad |\overline{E_{Lar'}^R} \cap E_{Rob'}^R| = |\overline{E_{Lar'}} \cap E_{Rob'}|$$



(a) Lar'/Rob' (b) Lar'^R/Rob'^R

FIGURE 5 – Histogramme de l'ensemble $\{CONF_{Rob'}(\{u,v\})$ tel que $\{u,v\} \in (E_{Lar'} \cap \overline{E_{Rob'}})\} \bigcup \{CONF_{Lar'}(\{u,v\})$ tel que $\{u,v\} \in (E_{Rob'} \cap \overline{E_{Lar'}})\}$, en parallèle à l'histrogramme de l'ensemble $\{CONF_{Rob'^R}(\{u,v\})$ tel que $\{u,v\} \in (E_{Lar'^R} \cap \overline{E_{Rob'^R}})\} \bigcup \{CONF_{Lar'^R}(\{u,v\})$ tel que $\{u,v\} \in (E_{Rob'^R} \cap \overline{E_{Lar'^R}})\}$

Par construction nous avons $GED(Lar',Rob') = GED(Lar'^R,Rob'^R)$ par contre la différence est clairement visible en comparant la distribution des valeurs de confluence des arêtes contradictoires dans Lar' vs Rob' d'une part (fig. 5(a)) et dans Lar'^R vs Rob'^R de l'autre (fig. 5(b). Quantitativement, la différence entre Lar'/Rob' et Lar'^R/Rob'^R est identique : $GED(Lar',Rob') = GED(Lar'^R,Rob'^R) = 0,47$, mais elle diffère structurellement, $\mu(Lar',Rob') = 0,80$ alors que $\mu(Lar'^R,Rob'^R) = 0,24$ (sur 50 réalisations : $std < 0,005$). Il y'a le même nombre de désaccords, mais ces désaccords sont structurellement faibles entre Lar' et Rob', alors qu'ils sont structurellement forts entre Lar'^R et Rob'^R. C'est ce que nous permet de voir la figure 5 et c'est ce que mesure μ.

Nous comparons maintenant un ensemble de réseaux lexicaux d'origines diverses, ressources construites par des lexicographes et par les foules (crowdsourcing) :

- **Rob** $= (\mathbf{V_{Rob}}, \mathbf{E_{Rob}})$ et **Lar** $= (\mathbf{V_{Lar}}, \mathbf{E_{Lar}})$: voir section 2 ;
- **Wik** $= (\mathbf{V_{Wik}}, \mathbf{E_{Wik}})$: Le wiktionnaire français est construit par les foules sur la base du volontariat. Wiktionary [7] est le compagnon lexical de Wikipedia. Ce dictionnaire multilingue inclus des gloses, des exemples, des relations sémantiques et des liens de traduction que n'importe qui peut modifier. Des instructions sont données aux contributeurs sous la forme de recommandations, mais aucune définition de la relation de synonymie n'est fournie. La construction des graphes de synonymie à partir des « dumps » de Wiktionnary [8] est précisément documentée dans (Sajous *et al.*, 2011). Le graphe $Wik = (V_{Wik}, E_{Wik})$ extrait du wiktionnaire français en janvier 2014 est construit de la même façon que le graph Rob ;
- **Jdm** $= (\mathbf{V_{Jdm}}, \mathbf{E_{Jdm}})$: La ressource *Jeux De Mots* [9] est construite selon une autre forme de crowdsourcing, à partir d'un jeu décrit dans (Lafourcade, 2007). Les joueurs doivent trouver autant de mots que possible qu'ils associent à un terme présenté à l'écran, selon une règle fournie par le jeu. Le but est de trouver le maximum d'associations sémantiques parmi celles que les autres joueurs ont trouvées mais que le joueur concurrent n'a pas trouvées. Plusieurs règles peuvent être proposées, dont la demande d'une liste maximale de synonymes ou quasi-synonymes. A partir des résultats collectés jusqu'en janvier 2014, un graphe de mots liés par des relations sémantiques typées (en fonction des règles) a été construit. Nous travaillons ici sur le sous-graphe des relations de synonymie.

Chacune de ces ressources est découpée en parties du discours (Noms, Verbes, Adjectifs), donnant ainsi trois graphes (ex : $Rob \Rightarrow Rob_N, Rob_V, Rob_A$). Le tableau 2 fournit les pédigrés de ces graphes et montre qu'ils sont tous des RPMH typiques. Dans le tableau 3 nous comparons six paires de graphes par partie du discours.

Entre les graphes *Lar*, *Rob*, et *Jdm* la mesure quantitative de surface *GED* est toujours comprise entre $0,45$ et $0,51$, ce qui indique un accord faible au niveau des liens locaux comparés indépendamment de leurs contextes structurels. Cependant la mesure structurelle μ est toujours supérieure ou égale à $0,70$ ce qui veut dire que malgré la proportion importante de désaccords locaux, ces trois graphes ont une structure globale semblable.

La mesure μ entre *wik* et les autres graphes est toujours inférieure à $0,50$ ce qui veut dire que *wik* diffère au niveau de ses zones denses par rapport à chacun des trois graphes *Lar*, *Rob*, et *Jdm*. Par exemple, la figure 6 montre les sous-graphes sur les voisins de *causer* extraits de Lar_V, Rob_V, Jdm_V et wik_V. On peut y voir un accord entre les trois graphes Lar_V, Rob_V, et Jdm_V au niveau de la *bisémie* du verbe *causer* (PARLER/PROVOQUER) ; le graphe *wik*, quant à lui ne distingue qu'un seul sens : PROVOQUER.

7. http://www.wiktionary.org/
8. Les dumps parsés sont disponibles au format XML à http://redac.univ-tlse2.fr/index_en.html
9. http://www.lirmm.fr/jeuxdemots/jdm-accueil.php

TABLE 2 – Pédigrés des graphes lexicaux (nous renvoyons à la légende de la figure 1 pour la description des colonnes).

Graphes lexicaux		n	m	⟨k⟩	C	L_{lcc}	λ (r^2)
Lar	**Adjectifs**	5510	21147	7,68	0,21	4,92	-2,06 (0,88)
	Noms	12159	31601	5,20	0,20	6,10	-2,39 (0,88)
	Verbes	5377	22042	8,20	0,17	4,61	-1,94 (0,88)
Rob	**Adjectifs**	7693	20011	5,20	0,14	5,26	-2,05 (0,94)
	Noms	24570	55418	4,51	0,11	6,08	-2,34 (0,94)
	Verbes	7357	26567	7,22	0,12	4,59	-2,01 (0,93)
Jdm	**Adjectifs**	9859	30087	6,10	0,16	5,44	-2,24 (0,90)
	Noms	29213	56381	3,86	0,14	6,48	-2,66 (0,93)
	Verbes	7658	22260	5,81	0,14	5,06	-2,08 (0,89)
Wik	**Adjectifs**	6960	6594	1,89	0,15	8,48	-2,46 (0,95)
	Noms	43206	37661	1,74	0,13	10,56	-2,51 (0,89)
	Verbes	7203	7497	2,08	0,25	9,22	-2,28 (0,92)

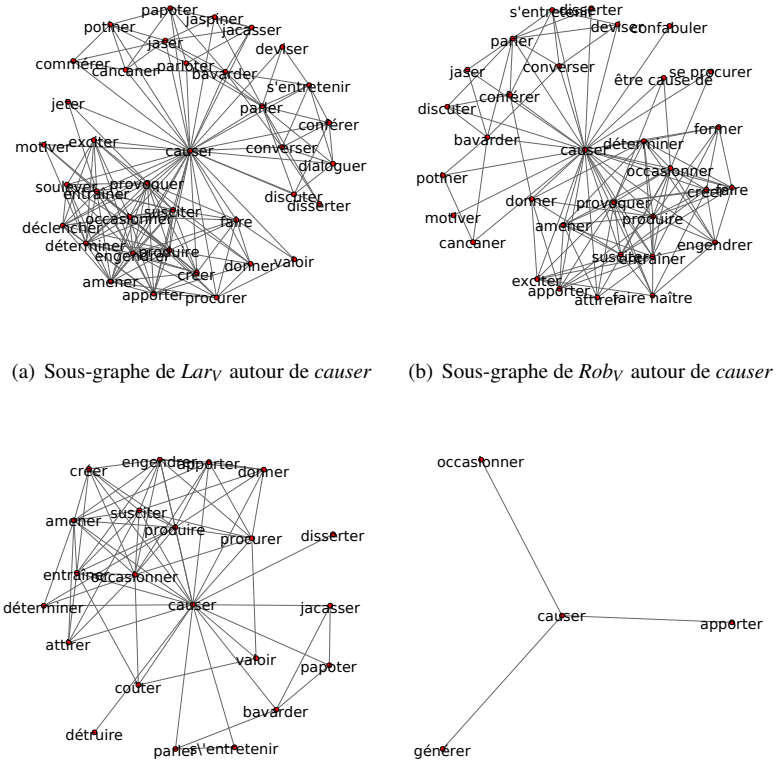
TABLE 3 – Pour comparer deux graphes lexicaux G_1/G_2, on réduit d'abord les deux graphes à leurs sommets communs : $G'_1 = \big(V' = (V_1 \cap V_2), E'_1 = E_1 \cap (V' \times V')\big)$ et $G'_2 = \big(V' = (V_1 \cap V_2), E'_2 = E_2 \cap (V' \times V')\big)$. Ensuite, nous construisons les graphes aléatoires équivalents G'^R_1 et G'^R_2 et calculons : $GED = GED(G'_1, G'2)$, $(\mu) = \mu(G'_1, G'2)$ et $(\mu^R) = \mu(G'^R_1, G'^R_2)$. Chaque valeur (μ^R) sur chacun des graphes aléatoires équivalents, est la moyenne sur 30 réalisations de $\mu(G'^R_1, G'^R_2)$ (tous les écarts type $std < 0,005$).

GED (μ) (μ^R) **sur les paires de graphes**			
G_1/G_2	Rob_A	Jdm_A	Wik_A
Lar_A	**0,45** (**0,76**) (0,34)	**0,47** (**0,71**) (0,38)	**0,75** (**0,41**) (0,06)
Rob_A		**0,51** (**0,70**) (0,29)	**0,71** (**0,42**) (0,05)
Jdm_A			**0,54** (**0,43**) (0,03)
G_1/G_2	Rob_N	Jdm_N	Wik_N
Lar_N	**0,48** (**0,70**) (0,20)	**0,48** (**0,70**) (0,19)	**0,72** (**0,31**) (0,03)
Rob_N		**0,47** (**0,70**) (0,13)	**0,71** (**0,29**) (0,02)
Jdm_N			**0,46** (**0,32**) (0,01)
G_1/G_2	Rob_V	Jdm_V	Wik_V
Lar_V	**0,48** (**0,73**) (0,40)	**0,46** (**0,70**) (0,39)	**0,78** (**0,25**) (0,05)
Rob_V		**0,47** (**0,70**) (0,37)	**0,78** (**0,25**) (0,06)
Jdm_V			**0,55** (**0,31**) (0,04)

5 Conclusion

« Dans un état de langue tout repose sur des rapports » disait Saussure (1972). Cependant, se limiter à l'analyse de ces rapports au seul niveau local, indépendamment de leurs contextes, n'est pas suffisant. En effet, nous avons montré que si $G_1 = (V, E_1)$ et $G_2 = (V, E_2)$ sont deux graphes standards de synonymie d'une même langue, une grande proportion de paires $\{x, y\} \in \mathbf{P}_2^V$ sont synonymes dans G_1 mais pas dans G_2. Une telle quantité de désaccords n'est pas compatible avec l'hypothèse selon laquelle la synonymie reflèterait une structure sémantique du lexique commune aux membres d'une même communauté linguistique. L'analyse d'une relation lexicale doit être faite au niveau de la structure globale dessinée par la relation. C'est ce que la mesure μ peut faire : avec un niveau de représentation adéquat, elle réconcilie les jugements portés par deux juges différents sur une même relation lexicale.

Ce n'est pas la somme du sens de ses arêtes qui donne le sens d'une relation lexicale, mais le sens de la relation lexicale dans la globalité de sa structure qui donne du sens à ses arêtes : *dans un état de langue tout repose sur la structure des rapports*.



(a) Sous-graphe de Lar_V autour de *causer*

(b) Sous-graphe de Rob_V autour de *causer*

(c) Sous-graphe de Jdm_V autour de *causer*

(d) Sous-graphe de Wik_V autour de *causer*

FIGURE 6 – Accord entre Lar_V, rob_V et Jdm_V sur la polysémie de *causer* (PARLER/PROVOQUER) mais désaccord avec Wik_V (PROVOQUER)

6 Remerciements

Nous remercions les organisateurs de RLTLN2014 pour avoir proposé et organisé ce workshop et les relecteurs qui, par leurs questions et leurs conseils toujours pertinents, nous ont permis d'améliorer cet article. Nous remercions aussi Franck Sajous, Yannick Chudy et Pierre Magistry pour les nombreuses discussions toujours enrichissantes que nous avons eues ensemble.

Références

ALBERT R. & BARABASI A.-L. (2002). Statistical Mechanics of Complex Networks. *Reviews of Modern Physics*, **74**, 74–47.

AMBAUEN R., FISCHER S. & BUNKE H. (2003). Graph edit distance with node splitting and merging, and its application to diatom idenfication. In *Graph Based Representations in Pattern Recognition, 4th IAPR International Workshop*, p. 95–106, York, UK.

BARONCHELLI A., I CANCHO R. F., PASTOR-SATORRAS R., CHATER N. & CHRISTIANSEN M. H. (2013). Networks in cognitive science. *CoRR*, **abs/1304.6736**.

BOLLOBAS B. (2002). *Modern Graph Theory*. Springer-Verlag New York Inc.

DE JESUS HOLANDA A., PISA I. T., KINOUCHI O., MARTINEZ A. S. & RUIZ E. E. S. (2004). Thesaurus as a complex network. *Physica A : Statistical Mechanics and its Applications*, **344**(3-4), 530–536.

DESALLE Y. (2012). *Réseaux lexicaux, métaphore, acquisition : une approche interdisciplinaire et inter-linguistique du lexique verbal.* PhD thesis, Université de Toulouse.

DESALLE Y., GAUME B. & DUVIGNAU K. (2009). SLAM : Solutions lexicales automatique pour métaphores. *Traitement Automatique des Langues*, **50**(1), 145–175.

DESALLE Y., GAUME B., DUVIGNAU K., CHEUNG H., HSIEH S.-K., MAGISTRY P. & NESPOULOUS J.-L. (2014a). Skillex, an action labelling efficiency score : the case for french and mandarin. In *Proc. of Cogsci'14, The 36th Annual meeting of the COGnitive SCIence society*, Quebec, Canada. À paraître.

DESALLE Y., NAVARRO E., CHUDY Y., MAGISTRY P. & GAUME B. (2014b). Bacanal : Balades aléatoires courtes pour analyses lexicales, application à la substitution lexicale. In *TALN'14, actes de l'atelier SemDis*, Marseille, France. À paraître.

GAILLARD B., GAUME B. & NAVARRO E. (2011). Invariant and variability of synonymy networks : Self mediated agreement by confluence. In *Proceedings of the The 49th Annual Meeting of the Association for Computational Linguistics : Human Language Technologies, 6th TextGraphs workshop : Graph-based Methods for Natural Language Processing*, Portland, Oregon.

GAO X., XIAO B., TAO D. & LI X. (2010). A survey of graph edit distance. *Pattern Anal. Appl.*, **13**(1), 113–129.

GAUME B. (2004). Balades Aléatoires dans les Petits Mondes Lexicaux. *I3 : Information Interaction Intelligence*, **4**(2).

GAUME B. (2008). Maping the form of meaning in small worlds. *Journal of Intelligent Systems*, **23**(7), 848–862.

GAUME B., MATHIEU F. & NAVARRO E. (2010). Building Real-World Complex Networks by Wandering on Random Graphs. *I3 : Information Interaction Intelligence*, **10**(1).

L. GUILBERT, R. LAGANE & G. NIOBEY, Eds. (1971-1978). *Le Grand Larousse de la langue française (7 vol.) 1971-1978*. Larousse.

KINOUCHI O., MARTINEZ A. S., LIMA G. F., LOURENÇO G. M. & RISAU-GUSMAN S. (2002). Deterministic walks in random networks : An application to thesaurus graphs. *Physica A*, **315**, 665–676. cond-mat/0110217.

LAFOURCADE M. (2007). Making People Play for Lexical Acquisition with the JeuxDeMots prototype. In *SNLP'07 : 7th Int. Symposium on NLP*, Pattaya, Thailand.

LEVENSHTEIN V. (1966). Binary codes capable of correcting deletions, insertions and reversals. *Soviet Physics Doklady*, **10**(8), 707–710.

MOTTER A. E., MOURA A. P. S., LAI Y. C. & DASGUPTA P. (2002). Topology of the conceptual network of language. *Physical Review E*, **65**, 065102.

MURRAY G. C. & GREEN R. (2004). Lexical Knowledge and Human Disagreement on a WSD Task. *Computer Speech & Language*, **18**(3), 209–222.

NAVARRO E. (2013). *Métrologie des graphes de terrain, application à la construction de ressources lexicales et à la recherche d'information.* PhD thesis, Université de Toulouse.

NAVARRO E., CHUDY Y., GAUME B., CABANAC G. & PINEL-SAUVAGNAT K. (2011). Kodex ou comment organiser les résultats d'une recherche d'information par détection de communautés sur un graphe biparti ? In *Proceedings of the Coria 2011 : Conférence en Recherche d'Information et Applications*.

NAVARRO E., GAUME B. & PRADE H. (2012). Comparing and fusing terrain network information. In E. HÜLLERMEIER, S. LINK, T. FOBER & B. SEEGER, Eds., *Scalable Uncertainty Management - 6th International Conference, SUM 2012, Marburg, Germany*, volume 7520 of *LNCS*, p. 459–472 : Springer.

NEWMAN M. E. J. (2003). The Structure and Function of Complex Networks. *SIAM Review*, **45**, 167–256.

P. ROBERT & A. REY, Eds. (1985). *Dictionnaire alphabétique et analogique de la langue française 2e éd. (9vol.)*. Le Robert.

SAJOUS F., NAVARRO E., GAUME B., PRÉVOT L. & CHUDY Y. (2011). Wisigoth semi-automatic enrichment of crowdsourced synonymy networks : an application to wiktionary. *LRE Language Resources and Evaluation : Special Issue on Collaboratively Constructed Language Resources*.

SAUSSURE (1972). *Cours de linguistique générale, édition critique préparée par Tullio De Mauro*.

STEWART G. W. (1994). *Perron-Frobenius theory : a new proof of the basics*. Rapport interne, College Park, MD, USA.

STEYVERS M. & TENENBAUM J. B. (2005). The large-scale structure of semantic networks : Statistical analyses and a model of semantic growth. *Cognitive Science*, **29**(1), 41–78.

WATTS D. J. & STROGATZ S. H. (1998). Collective Dynamics of Small-World Networks. *Nature*, **393**, 440–442.

Un reconnaisseur d'entités nommées du Français

Yoann DUPONT[1] Isabelle TELLIER[1]
(1) Université Paris 3 Sorbonne Nouvelle, 13, rue de Santeuil - 75231 Paris Cedex 05
yoann.dupont@etud.sorbonne-nouvelle.fr, isabelle.tellier@univ-paris3.fr

Résumé. Nous proposons une démonstration d'un reconnaisseur d'entités nommées du Français appris automatiquement sur le French TreeBank annoté en entités nommées.

Abstract. We propose to demonstrate a french named entity recognizer trained on the French TreeBank enriched with named entity annotations.

Mots-clés : REN, POS, apprentissage automatique, French Treebank, extraction d'information, CRF.

Keywords: NER, POS, machine learning, French Treebank, information extraction, CRF.

1 Introduction

La reconnaissance d'entités nommées (REN) est une tâche importante du TAL, pour laquelle il existe de nombreuses tâches telles que le MUC-7 (Appelt *et al.*, 1995), CoNLL (Tjong Kim Sang & Erik F. & De Meulder, 2003) ou encore NLPBA (KIM *et al.*, 2004) pour les entités biomédicales. Cependant, peu de corpus en Français sont disponibles pour cette tâche, rendant difficile la création d'outils appris automatiquement tels que (Favre & Béchet, 2005). Un autre reconnaisseur d'entités nommées du Français étant CasEN (Maurel *et al.*, 2011), qui a recours à des cascades de transducteurs. Parmi les corpus français les plus connus sont le corpus de la campagne d'évaluation ESTER (Gravier *et al.*, 2004), celui de la campagne d'évaluation ETAPE (Gravier *et al.*, 2012) ainsi que le French Treebank (Abeillé *et al.*, 2003) annoté en entités nommées (Sagot *et al.*, 2012). Ce dernier nous a permis d'entraîner un CRF (Lafferty *et al.*, 2001; Tellier & Tommasi, 2011) pour obtenir un reconnaisseur d'entités nommées du Français qui est l'objet de cette démonstration.

SEM (pour segmenteur étiqueteur markovien) est gratuit sous licence GNU 3 et librement disponible [1], il ne fonctionne que sur les systèmes Linux et Mac, il ne fonctionne pas sous Windows à l'heure actuelle. Pour fonctionner il nécessite :

- Un interpréteur python 2.5 ou supérieur. (`http://www.python.org/download/`)
- Wapiti version 1.5.0, une implémentation des CRF linéaires (`http://wapiti.limsi.fr/`)
- Pour bénéficier du versionnement du programme, le gestionneur de versions Bazaar est requis (`http://wiki.bazaar.canonical.com/`).

2 Fonctionnement du programme

Pour télécharger la branche du programme, il faut entrer la commande suivante dans un terminal [2] :

```
bzr branch lp:~yoann-dupont/crftagger/stand-alone-tagger
```

Le programme peut enchaîner divers traitements les uns à la suite des autres, comme effectuer un étiquetage POS, intégrer des dictionnaires, puis effectuer une passe de reconnaissance d'entités nommées. Une part importante dans l'apprentissage automatique supervisé étant l'ajout d'informations, en particulier pour la REN où les traits morphologiques et contextuels

1. la page web du programme : `http://www.lattice.cnrs.fr/sites/itellier/SEM.html`
2. Il est possible de télécharger une révision sans versionnement depuis : `https://code.launchpad.net/~yoann-dupont/crftagger/stand-alone-tagger`

sont parmi les plus pertinents, un langage (syntaxe XML) permet de définir des informations à extraire afin d'enrichir le corpus telles que :
– des observations locales (ex : le mot courant commence-t-il par une majuscule ? Est-il en début de phrase ?).
– des conjonctions/disjonctions d'observations locales (ex : le mot courant commence-t-il par une majuscule sans être en début de phrase ?).
– ajouter des dictionnaires de mots et de séquences de mots.
L'étiqueteur POS a été appris sur le French Treebank et validé selon un processus de validation croisée. Il reconnaît les étiquettes définies dans (Crabbé & Candito, 2008) avec une F-mesure de 97,3% en supposant les unités multi-mots déjà segmentées (c'est-à-dire regroupées en un seul token), et une de 95,2% lorsqu'elles ne sont pas déjà segmentées.

Le programme intègre plusieurs ressources externes, dont le LeFFF (Sagot, 2010) et divers dictionnaires constitués à partir de wikipedia français.

Le FTB annoté en entité nommées dispose de 7 types généraux que nous cherchons à reconnaître : Company (entreprise), FictionCharacter (personnage de fiction), Location (lieu), Organization (association ou organisation à but non lucratif par exemple), POI (Point Of Interest), Person (Personne) et Product (Produit).

L'évaluation du reconnaisseur d'entités nommées s'est faite selon un processus de validation croisée sur cinq plis en partant d'une annotation POS parfaite et en considérant l'égalité stricte sur les séquences des entités nommées (égalité du type et des frontières). En micro-average, la précision est de 86.38, le rappel de 80.30 pour une f-mesure de 83.23. En macro-average, les précision, rappel et f-mesure sont respectivement de 77.38, 53.01 et 62.92. La différence de qualité entre la micro- et la macro-average est due aux classes FictionCharacter et POI, dont la faible représentation dans le corpus rend leur identification particulièrement difficile par des outils statistiques.

Références

ABEILLÉ A., CLÉMENT L. & TOUSSENEL F. (2003). Building a treebank for french. In A. ABEILLÉ, Ed., *Treebanks*. Dordrecht : Kluwer.

APPELT D. E., HOBBS J. R., BEAR J., ISRAEL D., KAMEYAMA M., MARTIN D., MYERS K. & TYSON M. (1995). Sri international fastus system : Muc-6 test results and analysis. In *Proceedings of the 6th Conference on Message Understanding*, MUC6 '95, p. 237–248, Stroudsburg, PA, USA : Association for Computational Linguistics.

CRABBÉ B. & CANDITO M. H. (2008). Expériences d'analyse syntaxique statistique du français. In *Actes de TALN'08*.

FAVRE B. & BÉCHET F. (2005). Robust named entity extraction from large spoken archives. In *Proc. of the Emperical Methods in Natural Language Processing*.

GRAVIER G., ADDA G., PAULSSON N., CARR'E M., GIRAUDEL A. & GALIBERT O. (2012). The etape corpus for the evaluation of speech-based tv content processing in the french language. In *LREC*.

GRAVIER G., BONASTRE J., GALLIANO S., GEOFFROIS E., TAIT K. M. & CHOUKRI K. (2004). Ester, une campagne d'évaluation des systèmes d'indexation d'émissions radiophoniques. In *Journées d'Etude sur la Parole*.

KIM J.-D., OHTA T., TSURUOKA Y., TATEISI Y. & COLLIER N. (2004). An itroduction to the bio-entity recognition task at jnlpba. In *Proceedings of Natural Language Processing in Biomedical Applications (NLPBA 2004)*.

LAFFERTY J. D., MCCALLUM A. & PEREIRA F. C. N. (2001). Conditional random fields : Probabilistic models for segmenting and labeling sequence data. In *Proceedings of the Eighteenth International Conference on Machine Learning (ICML)*, p. 282–289.

MAUREL D., FRIBURGER N., ANTOINE J.-Y., ESHKOL I. & NOUVEL D. (2011). Cascades de transducteurs autour de la reconnaissance des entités nommées. In *Actes de TALN'11*.

SAGOT B. (2010). The lefff, a freely available, accurate and large-coverage lexicon for french. In *Proceedings of the 7th International Conference on Language Resources and Evaluation (LREC'10)*.

SAGOT B., RICHARD M. & STERN R. (2012). Annotation référentielle du corpus arboré de paris 7 en entités nommées. In *Actes de TALN'12, papier court (poster)*.

TELLIER I. & TOMMASI M. (2011). *Eric GAUSSIER et François YVON, Champs Markoviens Conditionnels pour l'extraction d'information*, chapter Modèles probabilistes pour l'accès à l'information textuelle. Hermès.

TJONG KIM SANG & ERIK F. & DE MEULDER F. (2003). Introduction to the conll-2003 shared task : Language-independent named entity recognition. In *Proceedings of the Seventh Conference on Natural Language Learning at HLT-NAACL 2003 - Volume 4*, CONLL'03, p. 142–147, Stroudsburg, PA, USA : Association for Computational Linguistics.