Collège Doctoral du Languedoc-Roussillon

THÈSE

Pour obtenir le grade de
Docteur

Délivré par l'**Université de Montpellier**

Préparée au sein de l'école doctorale
I2S - Information, Structures, Systèmes

Et de l'unité de recherche
LGI2P de l'école des mines d'Alès

Spécialité: **Informatique**

Présentée par **Nicolas Fiorini**

Semantic similarities at the core of generic indexing and clustering approaches

Les similarités sémantiques au cœur d'approches génériques d'indexation et de catégorisation

Soutenue le 04/11/2015 devant le jury composé de

M. Pierre Zweigenbaum, Directeur de recherche LIMSI, Université Paris-Sud, Paris	Rapporteur
M. Éric Gaussier, Professeur LIG, Université Joseph Fourier, Grenoble	Rapporteur
M. Patrice Bellot, Professeur LSIS, Aix-Marseille Université Marseille	Examinateur
Mme. Marianne Huchard, Professeur LIRMM, Université de Montpellier, Montpellier	Examinateur Présidente du jury
M. Zhiyong Lu, Directeur de recherche NCBI/NLM/NIH, Bethesda, MD, États-Unis	Examinateur
Mme. Sylvie Ranwez, Docteur, HDR LGI2P, École des mines d'Alès, Nîmes	Examinateur
M. Jacky Montmain, Professeur LGI2P, École des mines d'Alès, Nîmes	Examinateur
M. Vincent Ranwez, Professeur Montpellier SupAgro, Montpellier	Examinateur

Université Montpellier

❖

Synopsis de thèse

Les similarités sémantiques au cœur d'approches génériques d'indexation et de catégorisation

Nicolas Fiorini

Directeur de thèse	Jacky Montmain
Co-directeur de thèse	Vincent Ranwez
Maître de thèse	Sylvie Ranwez
Institut	École Nationale Supérieure des Mines d'Alès

Table des matières

Cette synthèse introduit le manuscrit rédigé en anglais qui s'intitule "*Semantic similarities at the core of generic indexing and clustering approaches*". Elle en reprend les idées principales en détaillant tout d'abord les travaux existants et le positionnement de la thèse par rapport à ceux-ci, fixe nos objectifs de recherche, puis présente nos contributions en indexation et catégorisation. Enfin, cet avant-propos s'achève sur les conclusions et enseignements tirés de ces recherches, ainsi que nos perspectives de travaux futurs.

I. Contexte et positionnement

L'Intelligence Artificielle (IA) se divise selon Russell and Norvig (1995) en quatre catégories : penser humainement, agir humainement, penser rationnellement et agir rationnellement. Cette définition ouvre un large champ de recherche qui englobe de nombreuses disciplines. Prenons pour exemple un système de recommandation tel que celui d'Amazon proposant des produits pouvant intéresser un internaute, et un programme de jeu d'échecs. Clairement, ces deux applications semblent avoir des fonctionnements sans lien commun et pourtant, ils reposent tous deux sur de l'apprentissage automatique. Ce paradigme, grandement exploité en IA, consiste à créer des modèles sur la base de diverses sources afin de faire des prédictions ou de prendre des décisions. Il diffère notamment des autres approches par le fait qu'il requiert de définir non pas un modèle à suivre, mais un ensemble de critères que le système optimise grâce à un ensemble d'exemples (jeu d'entrainement) pour créer un modèle optimisé sur les données.

L'apprentissage automatique est notamment utilisé de façon intensive dans

le cadre de la Recherche d'Information (RI). Celle-ci est au cœur de bien des systèmes logiciels impliquant des techniques appartenant à l'IA : recommandation, aide à la décision, veille technologique, etc. Ce domaine consiste à étudier les moyens de représenter, stocker, organiser et accéder à des éléments d'information (Baeza-Yates and Ribeiro-Neto, 1999), que nous nommons documents dans ce manuscrit. Le processus de RI est généralement familier, du moins du côté utilisateur, du fait de notre utilisation intensive des moteurs de recherche. La Figure A propose une représentation d'un tel processus. Tout d'abord, un utilisateur exprime un besoin en information sous la forme d'une requête. Le SRI (Système de Recherche d'Information) a en charge de trouver dans un corpus préalablement indexé (nous y reviendrons dans la section suivante), les documents qui font sens au regard de la requête (étape 1). Un score de pertinence est affecté à chaque résultat (étape 2), puis les résultats sont ordonnés en fonction de ce score et retournés à l'utilisateur. Il en découle que l'étape critique est l'association d'un score de pertinence à chaque document, qui repose principalement sur l'indexation des documents dans le corpus. Ces deux étapes sont détaillées dans la section suivante.

I.I. L'indexation, un processus clé de la recherche d'information

Le processus d'indexation d'un corpus de documents consiste à construire une représentation du contenu de chaque document la plus fidèle possible, mais favorisant une exploitation logicielle des plus rapides et efficaces. L'index est donc construit de façon à ce que la recherche d'information (étape 1) soit rapide. Avant le calcul de pertinence, il est important d'identifier les documents candidats au regard d'une requête (premier filtrage grossier). Puisque parcourir l'ensemble des documents serait bien trop coûteux,

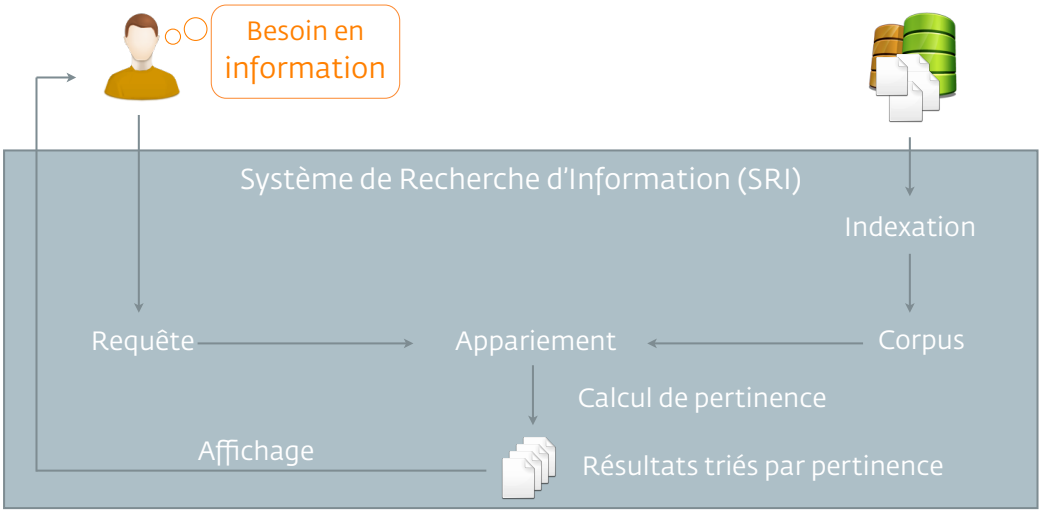


FIGURE A. : Processus classique de RI.

une solution consiste à construire un fichier inverse. L'idée est de stocker pour chaque élément de l'indexation (terme, concept,...), la liste des documents qui le contiennent dans leur indexation. De cette façon, pour une requête contenant un terme, le SRI peut instantanément retourner l'ensemble des documents potentiellement intéressants car contenant ce terme.

Etudions désormais l'étape suivante évoquée ci-dessus : le calcul de la pertinence. Dans la littérature, de nombreux modèles de pertinence ont été proposés. Les plus basiques sont les modèles booléens, qui demandent d'exprimer la requête sous une forme logique (Lancaster and Gallup, 1973). Le principal problème de cette approche est qu'une fois que les documents potentiels ont été identifiés, leur score de pertinence est binaire : 1 si le document satisfait la représentation logique de la requête, zéro sinon. Les modèles vectoriels (Salton et al., 1975) s'appuient quant à eux sur l'algèbre. Ils proposent de représenter la requête et les documents sous la forme de vecteurs de poids associés aux termes qu'ils contiennent. Ensuite, la si-

milarité entre deux vecteurs peut être estimée grâce aux propriétés algébriques, par exemple *via* une mesure de cosinus. Le dernier paradigme exploité est probabiliste (Maron and Kuhns, 1960 ; Robertson et al., 1995) et modélise la probabilité d'un document d'être pertinent pour une requête. Les probabilités élémentaires sont apprises à partir d'un ensemble d'exemples. Tous ces modèles sont malgré tout extrêmement dépendants de la qualité de l'indexation (ou annotation) des documents, qui reste au cœur de la recherche d'information. C'est pourquoi elle fait l'objet principal de cette thèse avec deux questions en filigrane : est-ce que l'utilisation d'ontologies de domaine et de mesures sémantiques conduisent à de meilleures prédictions/décisions ? Et ces approches permettent-elle le développement de solutions génériques ?

I.II. De l'utilité des données catégorisées

La RI exploite également d'autres processus de l'IA, comme par exemple la catégorisation ou classification. Catégoriser est une tâche que l'Homme s'attache à faire depuis bien longtemps, et ce de façon très fréquente à des fins diverses d'apprentissage ou de transmission de connaissance, par exemple. La catégorisation consiste à rassembler des objets *similaires* et à les éloigner de groupes d'objets *différents*, ce qui nécessite dans un cadre informatique de classification automatique la définition d'une *distance* qui permettra de faire ces rassemblements ou séparations (Manning et al., 2008). Deux grandes classes d'approches coexistent : la catégorisation hiérarchique et le partitionnement. La première consiste à identifier la structure inhérente aux documents catégorisés sous la forme d'un arbre (par exemple, la phylogénie des espèces en bioinformatique) alors que le partitionnement se focalise sur la délimitation de groupes. Le choix de l'une ou de l'autre

dépendra des données d'entrée et du résultat souhaité.

En RI, la classification est utilisée notamment pour la diversification des résultats. Comme Clarke et al. (2011) l'expliquent, les résultats d'une recherche, de par leur "proximité" à une requête, ont de bonnes chances d'être similaires entre eux, et cette redondance peut ne pas satisfaire l'utilisateur. La classification permet de pallier ce problème comme c'est notamment explicité dans Gollapudi and Sharma (2009), où l'idée est de proposer des groupes de résultats concernant les différents aspects de la requête. Agrawal et al. (2009) ; Skoutas et al. (2010) suggèrent que cette approche peut même contrer une éventuelle ambiguité de la requête en proposant des résultats correspondant aux différents sens qu'elle peut avoir.

La catégorisation est souvent suivie d'une étape d'étiquetage qui permet de comprendre les groupes qui ont été réalisés (Role and Nadif, 2014). Cette tâche est elle aussi parfois automatisée, comme dans les travaux de Bernardini et al. (2009) : leur application propose, suite à une recherche d'information, plusieurs groupes de résultats étiquetés que l'utilisateur peut choisir pour spécifier sa requête.

I.III. Vers des approches basées sur la connaissance d'un domaine

Afin de rendre un système *intelligent*, il semble intuitif de le pourvoir d'une certaine connaissance d'un domaine. Pour ce faire, une solution est de représenter la connaissance de façon compréhensible pour l'outil informatique (Russell and Norvig, 1995). L'ontologie est certainement la représentation la plus connue, mais d'autres plus ou moins formelles ont été décrites (Harispe et al., 2015b ; Sy et al., 2012). Par conséquent, des systèmes faisant usage de ces connaissances ont émergé, notamment dans les domaines où

un haut niveau d'expertise est requis comme dans le domaine biomédical (Smith et al., 2007). Ces systèmes ont été mis à l'épreuve et comparés aux systèmes plus *classiques*, par exemple ceux qui sont basés sur des termes et utilisent des techniques de traitement de la langue naturelle (TALN). Une nette amélioration a été constatée en RI (Haav and Lubi, 2001), principalement parce que les approches classiques sont perturbées par la présence de synonymie et l'ambiguité inhérente à la langue (Giunchiglia et al., 2009 ; Bhagdev et al., 2008 ; Stokoe et al., 2003). Le bénéfice de telles approches ne réside pas uniquement en l'absence de synonymie (ainsi que de polysémie) grâce à l'utilisation de concepts—les unités de sens dans une représentation de connaissances—en place des termes. La structure proposée par la représentation des connaissances permet par exemple de *savoir* que CHIEN est un MAMMIFÈRE en considérant les relations entre les concepts, et ensuite d'exploiter cette connaissance dans le processus.

Les mesures de similarité sémantique visent justement à exploiter ces différentes relations afin d'estimer la similarité de deux concepts. De nombreuses mesures existent et Harispe (2014) en propose une description étendue. Dans nos travaux, nous nous appuyons principalement sur des mesures de similarité sémantique exploitant la théorie de l'information et tenant compte des relations taxonomiques (lien de spécialisation *is_a* ou de généralisation), comme dans l'exemple cité au-dessus. A la vue du gain pour ce qui est de la qualité des systèmes basés sur des représentations des connaissances, il apparaît logique que leur utilisation ait été étendue à l'indexation et à la catégorisation.

I.III.1. L'annotation sémantique de documents

Une nette amélioration des résultats de la RI ayant été constatée lors de

l'utilisation de systèmes basés sur la représentation de connaissance, l'indexation sémantique s'est rapidement imposée. Elle a été notamment appliquée aux articles scientifiques, dans le domaine biomédical (PubMed, du NCBI, propose des articles annotés par le MeSH par exemple). L'indexation sémantique consiste à indexer les documents d'un corpus avec des concepts d'une ontologie de domaine plutôt qu'avec des termes. Pour automatiser ce processus, plusieurs stratégies ont été envisagées. La première découle directement des approches textuelles et consiste à extraire les concepts du texte. En sus d'identifier les termes du document qui prédominent, le but est ici de rechercher les concepts auxquels il font référence. C'est précisément l'idée derrière MetaMap (Aronson, 2001). La principale difficulté de ces approches reste de lever toute l'ambiguité lors du passage du terme au concept, problème pour lequel MaxMatcher (Zhou et al., 2006a) apporte des éléments de réponse. Pour avoir une vision plus large de ces approches, il est possible de se referer à Neves and Leser (2014) qui proposent une étude rassemblant de nombreux outils.

L'alternative à l'extraction des concepts consiste à reposer sur des approches d'apprentissage automatique. Ces méthodes requièrent l'identification de critères qui, optimisés sur un jeu d'apprentissage, permettront de prédire les annotations du document. Plusieurs pistes sont suivies, comme l'utilisation des précédents articles des auteurs ou les articles cités (Delbecque and Zweigenbaum, 2010) ; le score associé à chaque concept retourné par une méthode d'extraction de concepts (Gay et al., 2005) ; ou la fréquence d'apparition du concept dans des documents du corpus proches du document à annoter (Trieschnigg et al., 2009). Le fait de s'appuyer sur des documents déjà annotés est une méthode très utilisée, appelée méthode des k plus proches voisins (Huang et al., 2011 ; Mao and Lu, 2013 ; Mao et al.,

2014). Elle consiste à identifier le *voisinage* d'un document grâce à un système similaire à la RI. Par exemple, en soumettant le titre du document cible en requête, on obtient une liste de documents proches. Ces documents étant déjà annotés, leurs concepts sont ensuite filtrés ou ordonnés par de nombreuses approches d'apprentissage automatique (Huang et al., 2011 ; Delbecque and Zweigenbaum, 2010 ; Liu et al., 2014).

I.III.2. La catégorisation sémantique

L'intégration de bases de connaissances a également été expérimentée pour plusieurs approches de classification automatique comme le notent Bharathi and Venkatesan (2012). Les premiers à développer l'idée et proposer de classifier des documents textuels sur la base des concepts extraits de leur contenu sont Hotho et al. (2001, 2002, 2003). Ces derniers émettent l'hypothèse que comparer les documents sur la base de leurs annotations textuelles n'est pas suffisant et proposent de les comparer selon les concepts qui en sont extraits. Afin de tirer parti de la structure de connaissance associée, ils intègrent dans le calcul les hypernymes directs des concepts. Ainsi, deux documents annotés par les termes "chien" et "chat" seront considérés comme proches puisqu'ils seront respectivement annotés par les concepts {CHIEN, CARNIVORE, MAMMIFÈRE} et {CHAT, CARNIVORE, MAMMIFÈRE}. Globalement, cette façon de procéder est au coeur des approches de catégorisation sémantique (Spanakis et al., 2011 ; Yoo and Hu, 2006). Certains travaux vont plus loin pour exploiter la représentation de connaissances en utilisant des mesures de similarité sémantique entre les concepts annotant les documents (eux aussi extraits du contenu).

A la suite de regroupements faisant intervenir diverses méthodes de calcul de similarité, il est nécessaire de les étiqueter pour rendre explicites les

raisons de ce regroupement. Les approches existantes sont principalement dédiées à l'étiquetage de groupes de gènes. Cela s'explique par le fait que les biologistes ont spécifiquement besoin de ces outils pour analyser les résultats d'une puce à ADN, par exemple. En effet, ce processus fournit de nombreuses valeurs d'expression pour un grand nombre de gènes testés dans différentes conditions. Ceux exprimés de façon similaire et dans des conditions similaires ont de grandes chances d'intervenir dans le même processus métabolique qu'il convient d'identifier. Cette identification passe par un *étiquetage* du groupe de gènes concernés sur la base des annotations sémantiques qui leur sont associées[1]. La principale limite de cette approche est qu'elle est souvent basée sur une étude statistique qui cherche à identifier les concepts surreprésentés dans le groupe (Beissbarth and Speed, 2004 ; Lee et al., 2005 ; Bauer et al., 2008), or cette stratégie est souvent appliquée à des partitions de gènes et non à des structures hiérarchiques. Dans cette thèse, nous nous confrontons au problème de donner une sémantique à une hiérarchie de catégories, où les plus abstraites devraient être annotées par des concepts généraux alors que les plus fines devraient l'être par des concepts plus spécifiques.

II. Contributions de la thèse

II.I. Une approche générique d'indexation sémantique

L'état de l'art fait état de nombreuses approches pour annoter sémantiquement des documents textuels, particulièrement des articles du domaine biomédical ; or, d'autres types de documents sont souvent annotés par des concepts issus d'ontologies, notamment des séquences génétiques ou des

[1] Il est pratique courante d'annoter les gènes avec des concepts de la Gene Ontology.

images. Ainsi, nous nous somme concentrés sur la **réalisation d'une approche rapide et générique d'indexation sémantique de documents**.

II.I.1. Détail de la méthode

L'idée d'une telle approche est de ne pas dépendre exclusivement du contenu du document pour l'annoter, mais de s'appuyer sur les annotations des documents qui lui sont proches. Nous avons repris une technique récurrente dans la litérature, celle basée sur les k plus proches voisins. Celle-ci permet de disposer d'un ensemble de concepts potentiellement pertinents pour annoter un document donné. L'innovation de notre approche réside essentiellement dans la manière d'identifier les concepts pertinents au sein de cet ensemble. Nous partons du principe que des documents proches au regard d'un système de recherche d'information (et donc proches en matière de contenu) doivent aussi être proches en matière d'annotations. Ainsi, la qualité d'annotation du document est calculée selon sa similarité avec les documents voisins. Le calcul de similarité est rendu possible par l'utilisation de mesures de similarités sémantiques, qui tiennent compte de la structure de connaissance. De plus, il est possible de calculer la similarité sémantique de deux groupes de concepts (donc de deux annotations) avec une telle approche. De fait, on compare non plus la pertinence d'un concept au regard du voisinage, mais une annotation complète. Ce type d'approche permet de favoriser la *synergie* de l'annotation en empêchant des redondances comme un concept parent et son fils dans l'annotation (par exemple {CHIEN, CANIDÉ}).

Cependant, la simple similarité avec le voisinage ne suffit pas pour constituer une annotation valable. En effet, cette seule condition n'empêcherait pas d'annoter le document avec de nombreux concepts, or souvent, l'in-

dexation d'un document se résume à une dixaine de concepts. Au minimum, il faut être capable d'adapter la taille de l'annotation selon la taille usuellement choisie pour le corpus. Nous proposons une fonction objectif rassemblant ces éléments (similarité sémantique, similarité avec le voisinage et contrainte sur la taille de l'annotation) qui **permet d'annoter un document uniquement sur la base de son voisinage**. La description de la fonction objectif ainsi que d'un algorithme l'implémentant a fait l'objet de la publication scientifique suivante :

Indexation conceptuelle par propagation. Application à un corpus d'articles scientifiques liés au cancer.
Nicolas Fiorini, Sylvie Ranwez, Vincent Ranwez, et Jacky Montmain. Actes de CORIA 2014, COnférence en Recherche d'Information et Applications, Nancy, France, 19-21 mars 2014.

II.I.2. Optimisation algorithmique

La principale limite de l'utilisation des similarités sémantiques est la complexité algorithmique qu'elles induisent. Par conséquent, nous nous sommes efforcés de créer une approche de complexité polynomiale raisonnable. L'algorithme permettant d'indexer un document nommé USI (User-oriented Semantic Indexer) est optimisé, ce qui nous permet de réduire significativement sa complexité. Afin de tester la pertinence de la complexité finale, nous comparons notre approche aux approches de l'état de l'art pour ce qui est du temps d'exécution. Les résultats de cette comparaison sont sans appel, puisqu'**USI s'exécute 50 fois plus rapidement qu'une approche basée sur l'apprentissage automatique**, déjà rapide en soi. Cette optimisation ainsi que la comparaison des résultats d'USI avec l'existant sont proposées dans la publication suivante :

USI : a fast and accurate approach for conceptual document annotation.

Nicolas Fiorini, Sylvie Ranwez, Jacky Montmain, Vincent Ranwez. BMC Bioinformatics, Volume 16, Issue 83, 14 March 2015.

II.I.3. Évaluation sur un challenge international

Afin de tester la qualité et la pertinence de notre approche, nous avons participé à un challenge d'annotation d'articles biomédicaux à grande échelle, BioASQ 2015. Nous avons développé plusieurs variantes d'USI afin non seulement de tester sa flexibilité, mais aussi de proposer un système optimisé pour ce challenge. En effet, bien qu'USI soit générique, son but est aussi d'être facilement adaptable, c'est-à-dire facilement optimisable sur un ensemble spécifique de documents.

Nous avons testé différentes tailles de voisinage, différentes mesures de similarité sémantique, l'intégration de *baselines*[2]. Chaque jeu de test[3] devait être annoté en moins d'une journée et pouvait contenir jusqu'à plus de 21 000 documents. Le meilleur résultat d'USI est obtenu pour le troisième jeu de test où il se positionne second. Sur la totalité des jeux de test, il est troisième parmi une quinzaine de participants internationaux. Ce résultat est extrêmement encourageant, tenant compte du fait qu'USI est une méthode générique en comparaison aux approches bien plus lourdes (comportant des phases d'apprentissage longues) et capturant les moindres spécificités du contexte. **Il valide par là même l'utilisation de similarités sémantiques et suggère l'utilisation d'une approche générique cou-**

[2] BioASQ fournit, pour chaque jeu de test, les résultats d'approches considérées comme références.

[3] Au total, 15 jeux de tests ont été proposés.

plée à des approches exploitant plus précisément les spécificités du contexte applicatif. Les résultats à ce challenge, ainsi que la description des différentes variantes d'USI soumises au challenge ont fait l'objet de la publication suivante :

USI at BioASQ 2015 : a Semantic Similarity-Based Approach for Semantic Indexing

Nicolas Fiorini, Sylvie Ranwez, Sebastien Harispe, Jacky Montmain and Vincent Ranwez. In Working Notes for the Conference and Labs of the Evaluation Forum, CLEF 2015, Toulouse, France, September 8-11 2015.

II.I.4. "U" pour *User-oriented*

Au-delà de l'annotation entièrement *automatique* de documents, nous avons voulu étudier l'impact occasionné par l'intervention d'un expert. En effet, notre objectif n'est en rien de remplacer l'expert, mais au contraire de lui fournir un environnement qui l'assiste dans ses activités quotidiennes et allège certaines phases du processus d'indexation, ceci en s'adaptant au maximum à son contexte. Puisque la sélection du voisinage est une étape critique pour USI, nous souhaitons permettre à l'utilisateur de participer à sa sélection. Nous proposons de suivre l'intuition de Delbecque and Zweigenbaum (2010) sur un corpus d'articles scientifiques. Pour un document à annoter, nous recherchons les articles déjà publiés des co-auteurs ainsi que ceux cités en références. Comme chaque document est annoté sémantiquement, nous calculons la similarité sémantique de toutes les paires de documents que nous projetons sur un espace à deux dimensions grâce à la technique MDS (Multi-Dimensional Scaling). Ensuite, nous demandons à l'utilisateur de pointer l'endroit où, selon lui, ce document devrait se situer sur la carte sémantique. Un clic sur cette carte identifie implicitement un

voisinage qui permet ensuite l'annotation du document.

Afin de comparer une telle approche à une autre entièrement automatique, nous simulons des clics experts aux endroits où le document *devrait* être positionné. C'est-à-dire, lorsque nous calculons les similarités des paires de documents, nous calculons aussi celles correspondant au document à annoter, puisque nous disposons de son annotation dans le cadre de l'évaluation. La carte est ainsi créée et le document en question en est retiré, puis un clic est simulé à l'endroit où il était positionné. USI est lancé sur la base du voisinage défini implicitement par le clic et une annotation est générée. Les résultats montrent qu'une telle approche **permet d'obtenir de meilleures annotations grâce à un voisinage mieux défini** qu'avec une approche totalement automatique. Deux démonstrateurs sont disponibles afin d'essayer le principe de la carte sémantique pour annoter un document :

Démonstration sur des articles biomédicaux
http://bio.usi.nicolasfiorini.info

Démonstration sur des films
http://movies.usi.nicolasfiorini.info

II.I.5. Impact de l'imprécision

L'action de l'expert peut toutefois être imprécise. Nous avons donc étudié l'impact d'un écart du clic de la souris et la sensibilité d'une telle approche.

Nous avons ainsi amélioré l'outil visuel **en l'agrémentant d'un indice de sensibilité**. Lorsque la carte est créée, elle est divisée en n sous-cartes (ou tuiles). Un clic est simulé au centre de chacune d'entre-elles, leur donnant

ainsi une annotation sémantique. Les similarités sémantiques de chaque tuile avec les tuiles adjacentes sont calculées. Lorsque l'utilisateur survole la carte avec sa souris, une zone est colorée sur la carte, correspondant aux tuiles très similaires à la tuile survolée. Par conséquent, si l'utilisateur survole une tuile dont la zone colorée est large, cela veut dire que l'imprécision importe peu puisque l'annotation en résultant sera à peu près similaire. Par contre, si cette zone est petite, alors l'utilisateur devra prêter plus d'attention à l'endroit du clic. Cette étude de l'impact de l'imprécision lors de l'interaction avec l'utilisateur a fait l'objet de la publication suivante :

Coping with imprecision during a semi-automatic conceptual indexing process.
Nicolas Fiorini, Sylvie Ranwez, Jacky Montmain, and Vincent Ranwez. In Information Processing and Management of Uncertainty (part III), proceedings of IPMU 2014, 15th International Conference on Information Processing and Management of Uncertainty in Knowledge-Based Systems, Series : Communications in Computer and Information Science, Vol. 444, Springer, Laurent, A., Strauss, O., Bouchon-Meunier, B., Yager, R.R. (Eds.), ISBN : 978-3-319-08851-8, pp. 11-20, Montpellier, France, July 15-19 2014.

II.II. Catégorisation et étiquetage sémantique

Notre approche générique de l'indexation sémantique a été étendue pour répondre au besoin de catégorisation et d'étiquetage de catégories. Plus particulièrement, nous y explorons l'impact de l'utilisation de similarités sémantiques dans le cadre de la catégorisation de documents annotés sémantiquement. Là encore, l'idée est de **réaliser une approche de classifica-**

tion et de labellisation indépendante du type de documents.

II.II.1. Détails de l'approche

La catégorisation hiérarchique proposant une structure de catégories et non une simple partition, nous nous sommes concentrés sur cette technique plus informative. L'algorithme classique pour une telle approche est itératif. Dans un premier temps, chaque document est placé seul dans une catégorie. Ensuite, les paires de catégories sont itérativement regroupées jusqu'à ce qu'il n'en reste qu'une. Cette technique requiert donc d'être capable de calculer plusieurs similarités. La première est celle entre deux documents, la seconde est celle entre deux catégories, i.e. entre groupes de documents. La similarité entre deux documents est fortement dépendante du type de documents à classifier. Quant à la similarité entre groupes de documents, elle est traditionnellement une agglomération des similarités des paires de documents dans les deux catégories comparées (minimum, maximum, moyenne...).

Nous proposons de remplacer ces métriques par une similarité sémantique de groupe. La similarité entre deux documents peut en effet être calculée selon la similarité sémantique de leurs annotations. Pour comparer deux groupes de documents, au lieu d'utiliser une agglomération des similarités par paire des documents qu'elles contiennent, nous proposons (i) d'étiqueter sémantiquement la nouvelle catégorie et (ii) d'utiliser cet étiquetage sémantique pour la comparer avec les autres. L'étiquetage d'une catégorie reprend notre algorithme d'indexation à la différence—importante—près que celui-ci inspecte les concepts plus généraux dans l'ontologie et utilise un critère permettant de sélectionner ces concepts plus généraux au fur et à mesure que les catégories deviennent abstraites. Ainsi, la classification re-

pose entièrement sur l'utilisation de similarités sémantiques et le contenu des documents n'est jamais pris en compte. Enfin, les arbres générés étant binaires (chaque nœud contient deux enfants), nous les rendons plus exploitables par l'Homme en diminuant leur profondeur sur la base des similarités sémantiques calculées entre les nœuds.

Il en découle que la création des catégories est fortement dépendante de leur étiquetage, et vice-versa. Par conséquent, nous proposons une approche **de catégorisation et d'étiquetage sémantique entièrement cohérente au regard d'une mesure de similarité sémantique** (i.e. les étiquettes coincident avec la classification).

II.II.2. Constitution de jeux de référence

Le principal problème rencontré dans notre approche était qu'il n'existait pas de jeu d'évaluation permettant de tester sa fiabilité. En effet, la catégorisation hiérarchique dépend énormément des données catégorisées (d'où notre motivation à créer une méthode générique, encore une fois). De plus, il n'existe pas non plus, à notre connaissance, de jeu de test pour catégoriser des données annotées sémantiquement.

Nous avons souhaité pallier ce manque en proposant un jeu d'évaluation. Sur la base de marque-pages annotés extraits du site `del.icio.us` (Wetzker et al., 2008), Andrews et al. (2011) ont proposé un ensemble de marque-pages annotés sémantiquement avec WordNet. Nous avons développé une **interface en ligne permettant de classifier manuellement des documents annotés sémantiquement** et proposé l'outil adapté aux marque-pages à des étudiants et chercheurs de l'école des mines d'Alès. Cet outil est une contribution à part entière car il a été conçu dans l'unique but d'aider

l'utilisateur à classifier les documents, en lui fournissant de nombreux outils visuels et en utilisant des moyens d'interaction intuitifs (par exemple, par glisser-déposer). Cette interface est disponible à l'adresse suivante en tant que démonstrateur : http://clustering.nicolasfiorini.info.

Les retours des utilisateurs nous ont permis de constituer un jeu d'évaluation. L'ensemble des marque-pages a été divisé en 8 jeux de données : un pour optimiser la méthode, les autres pour l'évaluer. Les classifications fournies par les utilisateurs concernent donc les 7 jeux de données et consistent en une arborescence de marque-pages dont chaque nœud est étiqueté sémantiquement avec WordNet. Ces jeux de données répondent parfaitement à notre désir d'approche générique puisqu'ils sont composés uniquement d'éléments annotés, sans aucune information sur le contenu de ces éléments. Les arbres ainsi proposés peuvent servir de **jeux de référence pour évaluer la classification, mais aussi l'étiquetage des catégories**. Les jeux de références sont bien sûr à disposition sur Internet, à l'adresse suivante : http://benchmark.nicolasfiorini.info.

II.II.3. Evaluation de la méthode

L'évaluation porte sur deux points : la classification et l'étiquetage des catégories. Pour évaluer la classification, nous nous sommes appuyés sur des mesures de distance d'arbres en comparant les arbres obtenus avec les arbres réalisés par les opérateurs humains (que l'on appelle arbres experts). Tout d'abord, nous avons observé les divergences entres les arbres experts entre eux et déduit un *écart type*. Ensuite, nous avons comparé la distance moyenne entre l'arbre obtenu et les arbres experts pour chaque jeu de test avec cet écart type. Nous avons procédé de la même facon pour évaluer une approche classique de classification. Il apparaît clairement que **la classification sé-**

mantique est meilleure qu'une approche classique basée sur une agglomération des similarités de paires de documents. Sur certains jeux de données, on n'observe pas de différence plus grande entre les arbres créés automatiquement et les arbres experts, qu'entre les arbres experts entre eux. Le code source de l'approche, ainsi que les résultats obtenus sur les jeux de référence sont disponibles à cette adresse : http://sc.nicolasfiorini.info.

L'évaluation des étiquettes sémantiques associées aux catégories consiste à comparer, pour chaque arbre expert de chaque jeu de données, les étiquettes produites par notre méthode à celles données par l'utilisateur. La qualité d'une étiquette est représentée par sa similarité sémantique avec l'étiquette experte. Les similarités sont moyennées, ce qui constitue un score pour chaque jeu de test. Nous avons comparé ces résultats à plusieurs autres techniques d'étiquetage, à savoir (i) prendre l'ensemble des concepts annotant les documents de la catégorie et (ii) utiliser exactement l'algorithme d'indexation présenté ci-dessus. Notre approche fournit de meilleures étiquettes que ces deux alternatives, prouvant qu'elle permet mieux qu'une approche d'indexation de **résumer un ensemble de concepts en quelques concepts en utilisant les propriétés de généralisation/spécialisation de l'ontologie**.

III. Conclusions et ouverture

Nos travaux explorent la faisabilité et la pertinence de plusieurs objectifs dans les domaines de l'indexation et de la catégorisation sémantiques.

1. *Étudier l'impact de l'utilisation de similarités sémantiques.*
   Toutes nos applications sont basées sur des similarités sémantiques.

Estimant qu'elles sont un excellent moyen d'exploiter la structure inhérente à une représentation de connaissance, nous proposons des approches innovantes en indexation et catégorisation les utilisant. Les résultats obtenus dans ces deux domaines suggèrent que leur utilisation devrait être encouragée.

2. *Proposer des approches génériques pour certains traitements basés sur la sémantique.*
   Parce qu'elles sont basées sur une description sémantique à l'aide des concepts d'une ontologie, les approches que nous proposons pour la manipulation de documents sont génériques. En effet, qu'il s'agisse de RI, de classification ou d'étiquetage, les mêmes traitements peuvent être accordés à des images, des publications scientifiques, des gènes, etc. Il est possible, dans ces approches, de faire abstraction du contenu des documents et la perte d'information due à ce choix semble compensée par la qualité apportée par l'utilisation d'une base de connaissances. Bien sûr, cette généricité n'empêche en aucun cas de réaliser des méthodes hybrides en enrichissant une approche générique avec une autre plus spécifique.

3. *Analyser la place de l'opérateur humain dans le processus.*
   Nos solutions visent à assister l'opérateur humain confronté à une situation faisant appel à de hautes compétences cognitives. Ainsi la place de l'expert reste un point clé de notre approche. Sans désir de se subsister à lui, les différents traitements que nous proposons se doivent d'être très fiables et d'apporter rapidement une réponse pertinente à ses attentes. C'est pourquoi notre évaluation tient systématiquement compte des experts : soit pour vérifier que l'interaction avec le système est adaptée à son contexte (des tests plus poussés avec des

ergonomes devront être envisagés), que la pertinence des résultats est équivalente à celle produite par un humain, soit par l'étude du gain de temps occasionné par l'introduction de traitements automatiques dans un processus par exemple d'indexation.

4. *Produire des approches algorithmiquement efficaces.*
   Ce manuscrit s'attarde sur les aspects algorithmiques des approches réalisées pour valider leur pertinence dans un contexte où le volume de données est très important. Nous avons démontré que les approches sémantiques n'empêchent pas un passage à l'échelle.

Ces conclusions apportent de nouvelles interrogations, ouvertures et perspectives. Tout d'abord, même si USI a été conçu et développé comme une approche générique, nous n'avons pas pu tester sa pertinence dans tous les domaines. Un effort a été réalisé concernant l'indexation d'articles biomédicaux, motivé par le fait qu'une communauté active anime ce champ d'application. D'autres cas d'utilisations ont été étudiés dans ce manuscrit comme une indexation de films, mais aucune évaluation formelle n'a été faite à ce niveau. Il serait très intéressant de tester, par exemple, cette approche pour l'annotation sémantique de gènes, en se basant sur leurs séquences génétiques pour récupérer les gènes voisins.

De plus, l'utilisation des bases de connaissances comme élément central de nos approches limite leur application puisqu'elle suppose de disposer d'un modèle de connaissance et de documents annotés sémantiquement avec ce modèle. Cette limite est à modérer cependant puisque de plus en plus de tels modèles sont disponibles. Au même titre que nous proposons toujours le code source, les jeux d'évaluations et les résultats, ces efforts devraient être poursuivis. Nous avons trop de fois été confrontés au manque de don-

nées nous empêchant de nous comparer à l'existant, ou simplement de valider une approche, alors que l'idéologie même du Web Sémantique est de partager et réutiliser les données[4].

Enfin, puisque BioASQ fédère les dernières contributions dans le domaine de l'indexation automatique de papiers biomédicaux, il est possible de connaître les méthodes qui ont permis les meilleurs résultats au challenge. Bien que nous ayons adapté USI au contexte du challenge, nous n'avons pas utilisé toutes les données mises à disposition (titre, résumé de publication, auteurs...). Nous supposons dans le manuscrit qu'enrichir USI avec des approches d'apprentissage automatique, de traitement automatique des langues ou de classification permettrait d'améliorer la qualité des annotations. Cependant, cela reste encore à prouver et constitue une réelle perspective de recherche.

[4] *The Semantic Web provides a common framework that allows data to be shared and reused across application, enterprise, and community boundaries*, http://www.w3.org/2001/sw/

Short abstract

In order to improve the search and use of documents, Artificial Intelligence has dedicated a lot of effort to the creation and use of knowledge bases such as ontologies. They are graphs in which nodes represent a meaning unit—a concept—and edges are their relationships. For example, this allows to represent the concept "dog" as a subclass of the concept "mammal". Indexing documents is a useful process for further processing and consists of associating them with sets of terms that describe them. These terms can be concepts from an ontology, in which case the annotation is said to be **semantic**. Such annotations benefit from the inherent properties of ontologies: the absence of synonymy and polysemy. Most approaches designed to annotate documents have to read them and extract concepts from this reading. This means that the approach is dependent from the type of documents, as a text would not be processed the same way a picture or a gene would be. Approaches that solely rely on semantic annotations can ignore the document type, leading to **generic processes**. This has been proved in Information Retrieval where researchers experienced approaches called

semantic information retrieval that can fit any type of document.

This thesis capitalizes on genericity accessible through **semantic annotations** to build novel systems and compare them to state-of-the-art approaches. To this end, we rely on semantic annotations coupled with semantic similarity measures. Of course, such generic approach can then be enriched with type-specific ones, which would increase the quality of the results. This work explores the relevance of this paradigm for indexing documents. The idea is to rely on already annotated close documents to annotate a target document. We defined a heuristic algorithm for this purpose that uses the semantic annotations of these close documents and semantic similarities to provide a generic indexing method. This resulted in USI (User-oriented Semantic Indexer) that we showed to perform as well as best current systems while being faster. This idea has been extended to another task, **clustering**. Clustering is a very common process that is useful for finding documents or understanding a set of documents. We propose a hierarchical clustering algorithm that reuses the same components of classical methods to provide a novel one applicable to any kind of documents. Another benefit of this approach is that when documents are grouped together, the group is annotated by using our indexing algorithm. Therefore, the result is not only a hierarchy of clusters containing documents as clusters are actually described by concepts as well. This helps a lot to better understand the result of the clustering. A particular attention has been devoted in this work to **algorithmic optimization** and user-friendliness, with **interactive human-machine interfaces**, that take into account imprecision of human actions.

This thesis shows that apart from improving the results of classical ap-

proaches, building conceptual approaches allows us to abstract them and provide a generic framework. Yet, while bringing **easy to setup methods**—as long as documents are semantically annotated—, genericity does not prevent us from mixing these methods with type-specific ones, in other words creating hybrid methods.

Dedicated to Maylin.

❖

Acknowledgements

My two supervisors, Jacky Montmain and Vincent Ranwez, have been of great help during this thesis and I would like to thank them first. Jacky Montmain gave his fantastic point of view on many topics that I would never have had otherwise. Vincent Ranwez provided me with his great expertise regarding algorithm optimization and clues for my research. Sylvie Ranwez, my advisor, has been tremendously helpful throughout these three years by guiding me and making my life at the lab so easy and for that, I am truely grateful.

I can't thank the two reporters of this thesis enough, Pierre Zweigenbaum and Éric Gaussier, for having accepted to review it. In fact, all members of the committee of the PhD defense have asked many relevant questions that provided valuable perspectives for my work. These questions or remarks also highlighted a few research interests that I did not consider so far. I am therefore really happy for the input they all took time to give me and I deeply thank Pierre Zweigenbaum, Éric Gaussier, Patrice Bellot, Marianne

Huchard and Zhiyong Lu.

My coworkers played a role in my every day motivation and pleasure. Sébastien Harispe deserves special thanks for his help with the Semantic Measures Library that he created, which is a key element of my whole work. Abdelhak Imoussaten put me back in shape at some point (now the writing up wrecks everything), I am glad he proposed me so many tennis matches—besides, his eternal good mood is a great source of self inspiration in life. I wish Stéphane Billaud a lot of joy with his newborn son, his presence and the discussions we had were greatly appreciated. Pierre-Antoine Jean, despite his recent arrival in the office has been a incredible pal to share time with. I also thank the whole PhD student crew of the lab for all the good moments: Sami Dalhoumi, Blazo Nastov, Abderrahman Mokni, Darshan Venkatrayappa, Mirsad Buljubasić and Diadié Sow.

I also want to express my deepest gratitude to Moreno Mitrović, a.k.a ðü and recently a.k.a my best man. You convinced me to start the PhD adventure and I have no regret at all. I am looking forward to discussing with you about all these topics we fancy and most of all, to working with you again. Two other great friends, Benjamin Chadouteau and Matthieu Thomas have been here for most of the entertainment I had these years and we all know how important those moments are. Thank you guys.

Many thanks to all the people I had the opportunity to work with aside from the doctoral work. Renata Lemos for a fantastic data extraction project that has been challenging on many aspects but your attitude contributed a lot to my motivation and I learnt a lot thanks to it. Freddy Xuhui Hu, who is sometimes in need of French data, for highlighting very interesting rules in French that I did not even realize. Linguistics become even more

magical with you for a newbie like me! Olivier Biberstein for coming a few months in Nîmes and sharing his Swiss enthusiasm along his vast knowledge. The LIRMM squad, Vincent Berry, Anne-Muriel Chifolleau, Vincent Lefort, François Chevenet for their support before and during the PhD. I think you have been the reason of my ambition for the five last years.

My Master's internship in the UK definitely changed my life and impacted many of my choices afterwards. I was priviledged to meet fantastic friends, Elaine Schmidt, Tarun Kumar, Anthony Davidson, James Willmoth, who made my stay so wonderful. Their attitude has been a tremendous source of inspiration since then and I am really happy to have met them. My work at the EBI, supervised by Javier Herrero, helped in convincing me to pursue with a PhD. Thank you Javier for your motivation and ambition at any moment, you really played a role in my choices and I am grateful for that.

I end these acknowledgements with a big thank you to my family. Olivier and Nathalie have always been my examples in life, I would not have been able to do all this without them. Huge thanks also to every other family member. The list would be too long, but to cite a few, thank you Éliane, Sophie, Xavier, Angélique, Christian, Jean-Michel, Brigitte & René & their children. My days are illuminated by Mathilde's love, who supports me in every situation. None of this would have been possible without you. We now share our lives with Maylin to whom I am happy—and I admit a bit proud—to dedicate this work. Finally, I would like to have a thought of love in memory of Bruno, Yves and Simone.

❖

Contents

33 Chapter 2

Semantic indexing

Chapter 3

103 Semantic clustering and cluster labeling

List of Figures

List of Tables

1

Introduction

Contents

> *Artificial Intelligence is the building of computer programs which perform tasks which are, for the moment, performed in a more satisfactory way by humans because they require high level mental processes such as: perception learning, memory organization and critical reasoning.*
>
> — Marvin Lee Minsky

1.1. General context

The above definition of Artificial Intelligence (AI[1]) by Marvin Lee Minsky, one of its creators, shows the wideness of the field that can be perceived through the diversity and the abstraction of the human cognitive processes AI tries to perform. Russell and Norvig (1995) classify AI into four categories: thinking humanly, acting humanly, thinking rationally and acting rationally. These are the trends that have existed—and still exist—in AI with their respective objectives. A rational system seeks to provide the best solution while others seek to mimic the Human. They also differentiate the process of acting, e.g. communicating, from that of thinking, e.g. reasoning.

The processes involving AI can thus be discrete such as the recommender systems—think of Amazon—, or, on the contrary very obvious like a chess playing program. This shows again the diversity of AI in terms of use, goal and implementation: playing chess seems intuitively very different from

[1] A comprehensive list of abbreviations used in this thesis is available in Appendix A.1

recommending a product to a user. However, the foundations of those two are quite similar as they are built upon key pillars of AI. The most obvious certainly is Machine Learning (ML), which is a whole AI domain that aims at building models out of several sources in order to make predictions or decisions. Instead of relying on a model or rules created by a human, the system learns from examples and experiences with a set of features that are considered important by the human. The idea behind it is that the system will converge towards human-like predictions or decisions when the number of examples and experiments increases. Each method has its own specifications, the so-called SVM (Support Vector Machine) excels in dividing an input space into two regions and L2R (Learning to Rank) is best for ordering the data, to cite a few. While ML approaches are touted for offering scalable solutions and generally good results, they require a learning phase for which the behaviour of the predefined features can be observed to model the problem.

This PhD thesis seeks to study methods that are not based on ML. Indeed, in many fields like those we focus on (see the next sections), approaches differ from a few learned features and slightly different ML algorithms lead to slightly different results. However, some of these fields can benefit from the availability of Knowledge Representations (KR), which should, in theory, improve the quality of the results provided by the systems, e.g. by being able to infer better predictions. At least, such approaches would be novel and it would be worth studying how they behave compared to classical ones. The few goals we set are to find out how to best use KRs, how they can be useful and to compare them with state-of-the-art classical ML implementations.

Globally, this thesis project falls within the scope of Information Retrieval as we concentrate on two related domains: indexing and clustering. The next sections thus present these fields and their stakes before describing the benefits and uses of knowledge-based systems. Once this basis has been set, we explain the objectives of the thesis in more detail and its outline.

1.2. Information retrieval, indexing and clustering

Nowadays, many tasks we rely on are based on AI systems. In fact, everything is called *smart*. We have smartphones, we (will) live in smart homes that (will) belong to smart cities. The most famous AI process in our daily habits is certainly Information Retrieval (IR), usually represented by Google or Bing search engines. Accessing information is so frequent and easy that our way of thinking has changed (Sparrow et al., 2011). That is, instead of remembering the actual information, we remember how to access it: with which keywords, on which website, etc. The Internet became, in some way, an external storage of our memory. This section provides a short overview of IR and its relationship with indexing and clustering.

1.2.1. The Information Retrieval field

In their so-called book, Baeza-Yates and Ribeiro-Neto (1999) point out that Modern Information Retrieval "*deals with the representation, storage, organization of, and access to information items*". "Information item" is soon replaced by "document", which is an important recurring word in this thesis that needs to

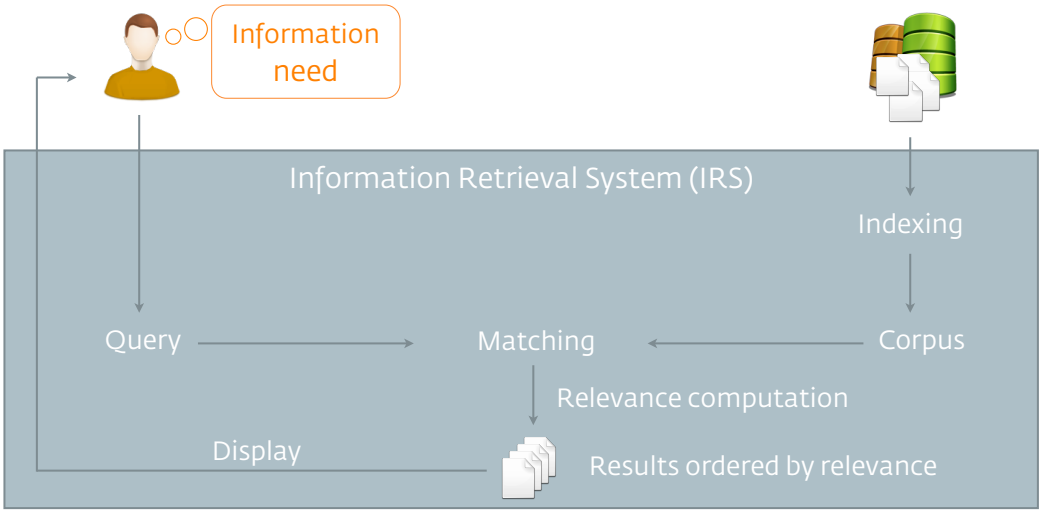


Figure 1.1.: Classical process of IR.

be defined. We refer to it the same way as the Oxford Dictionaries: "*A piece of written, printed, or electronic matter that provides information or evidence or that serves as an official record*"[2]. A document can thus be a scientific paper, a video or a gene sequence, to cite a few. All of them present the same properties of providing information while being stored electronically.

The proposed IR definition shows how wide the field is, which explains why an entire book is dedicated to it. However, the process of IR can be easily summarized by the Figure 1.1. A user expresses an information need as a query and submits it to the IR system (IRS). The IRS relies on a corpus of documents that it uses to find the ones to return to the user. To this end, it matches the query with them and associates each document with a relevance score. Matching is different in essence from associating a score since, for reasons of efficiency, not all documents of the corpus are scanned and scored. Instead, the index, such as an inverted file (see the

[2] http://www.oxforddictionaries.com/definition/english/document?t=1, as of November 9, 2015

next section on indexing), provides a simple way to find matches first, and then computing relevance scores. Results are ordered according to this relevance score and displayed to the user. A crucial step for the matching is the indexing of documents as it allows the IRS to retrieve the potentially relevant documents quickly. Even if this Figure is very simple to understand, each step is the result of an extensive research. To name a few, the transformation of an information need into a query is helped by the query expansion field; computing relevance of the documents depends on many factors that resulted in many models; and the display of results can assume many forms (e.g., ordered list, synthetic semantic map), each achieving a different goal.

Indexing appears to be a fundamental element in IR pipelines. In fact, it is crucial to allow the matching with the queries in a reasonnable time as it provides a logical representation of the documents that compose the corpus.

1.2.2. A need for indexed documents

The assessment of the relevance of each document is the most important step as it directly impacts the results. Therefore, many models have been proposed in the literature to improve the quality of IRSs, for most of which the index is at the basis. Indeed, the representation of the documents in the index is usually a set of keywords which are used for matching the query with the documents, but also to compute the relevance faster than by using the whole document.

The index is what is called an inverted file that is defined as follows. Con-

sider a corpus D[3] of three documents $d_1, d_2, d_3 \in D$. Each document is associated with a set of terms T_{d_i} that represent it so for example $T_{d_1} = \{t_1, t_2\}, T_{d_2} = \{t_1, t_3\}, T_{d_3} = \{t_1, t_2, t_3\}$. Building an inverted file consists in mapping the documents to the terms, in this case:

$$t_1 = \{d_1, d_2, d_3\}$$
$$t_2 = \{d_1, d_3\}$$
$$t_3 = \{d_2, d_3\}$$

By using an inverted file, matching a query with the corpus is much faster as we do not have to browse all documents to find those that are annotated with the terms corresponding to the query. We just need to get the documents associated to the keys that correspond to the query terms.

> In order to avoid confusion, let us clear up some terms that are commonly used throughout this thesis. The fact of associating words—or, further on, concepts—with documents is usually called *annotating*. These annotations are then used to build up an index for an IRS, by following the process illustrated above through an example. The most important part is thus the actual association on which we focus and not its implementation in IRSs, so following many authors, we denote this association by indexing or annotating interchangeably.

As an example of the use of the index, let us take the most simple model for IR, the boolean model (Lancaster and Gallup, 1973). Classical boolean

[3] A list of important and common mathematical notations is provided in Appendix A.2

operators—AND ($\wedge$), OR ($\vee$) and NOT ($\neg$)—are used in queries to make a logical representation of the user need. For example, consider the following query $q = t_1 \wedge \neg t_3$. In this case, d_1, d_2, d_3 are potential matches but only the document d_1 perfectly satisfies the logical representation. The relevance score is binary, which means it is 1 if the document annotation satisfies the boolean expression, 0 otherwise. Such an approach cannot propose an ordered list of results (apart from a list containing documents with a relevance of 1 first) and is limited in most contexts where we do not need a query as strict as a logical representation.

While the boolean model relies on the set theory, two other paradigms have been at the origin of many models: algebra and probability theories. The vector models (Salton et al., 1975) rely on algebra and need weighted term vectors for the query and the documents. They consist of weights that correspond to the terms annotating the documents or in the query. The similarity of two vectors can then be calculated by using algebraic function such as the cosine similarity. The probabilistic models (Maron and Kuhns, 1960; Robertson et al., 1995) model the probability of a given document to be relevant w.r.t the query. The elementary probabilities that are used in those models are estimated by learning from a set of examples. Boolean, vector and probabilistic models all highly rely on the index, that is the correspondance between each document and a set of terms. Sometimes a weight or frequency is also associated with the terms to refine the relevance scores, but the actual association of document-terms is therefore crucial in IR.

1.2.3. The importance and use of clustering

Clustering is useful for organizing documents and often allows for a better understanding of the underlying nature or meaning of data. Let us consider the whole set of articles of a newspaper in the last 10 years. An interesting study is to understand the trends of this newspaper: does it emphasize politics, sport, local news? Clustering the articles gives a good idea of these orientations. Biologists also frequently cluster genes to identify biological processes involved in molecular response to a change of environment. By storing the expression data of hundreds of genes in different environments—or after different stimuli—, they cluster these genes according to their expression patterns. The function of some unknown genes can appear to be part of the same metabolic pathway as the other known ones, for instance.

The whole point of clustering is to find groups of *similar* items that are *different* from other groups. As a result, the definition of a distance metric is the most important choice in clustering (Manning et al., 2008). Clustering methods are numerous and depend on the type of data: numeric values and textual ones are compared differently. Different algorithms may lead to different clusters as they come with different requirements and objectives. If you need a hierarchical representation of classes, you will use *hierarchical clustering* (e.g., in systematic biology where you need a phylogenetic tree); if you have a predefined number of clusters you will opt for k-means (e.g., you want to find the 5 main topics of a newspaper), and so on. The distance measures implicitly determine the feature(s) you will use for clustering the documents. However, note that several distance measures can rely on the same features, while providing a great diversity of output

clusters. Manning et al. (2008) also point out that there are two important paradigms called *soft* and *hard* clustering. The former aims at providing for each document a probability distribution over the classes, while the latter assigns each document to one class only.

The Figure 1.2 illustrates the usual k-means method that follows Lloyd's algorithm (Lloyd, 1982) to better understand the stakes of clustering in general. This approach proposes to identify k clusters among a set of items. It all starts by randomly defining k points as cluster means (1). Then, all items are assigned to the closest mean (2). The new means are computed regarding the items identified in each cluster (3). Assignment of items (2) and recomputation of the means (3) is done repeatedly until the means stop changing (4), k clusters are identified.

As we are facing an increasing volume of data, clustering appears to be a very advantageous course of action in many fields. Recently, in IR, some effort has been devoted to result diversification, motivated by a few objectives. We have all faced this problem one day, where all results of a search are literally the same ones. In fact, Clarke et al. (2011) even make the assumption that the first results of a search have good odds to be very similar to other searches. In this case, if the solution to the need is not in the first result, we have to crawl among the result pages to find something of interest with difficulty. Clustering is one solution among others for diversifying the results (Gollapudi and Sharma, 2009). The idea is to cluster the results so that groups of results emerge. These groups can represent different aspects or meaning of the topic. For example for the query "nuclear power plant", results could be grouped in ecological, political and energetic aspects. In order to diversify the results, once the clusters have been iden-

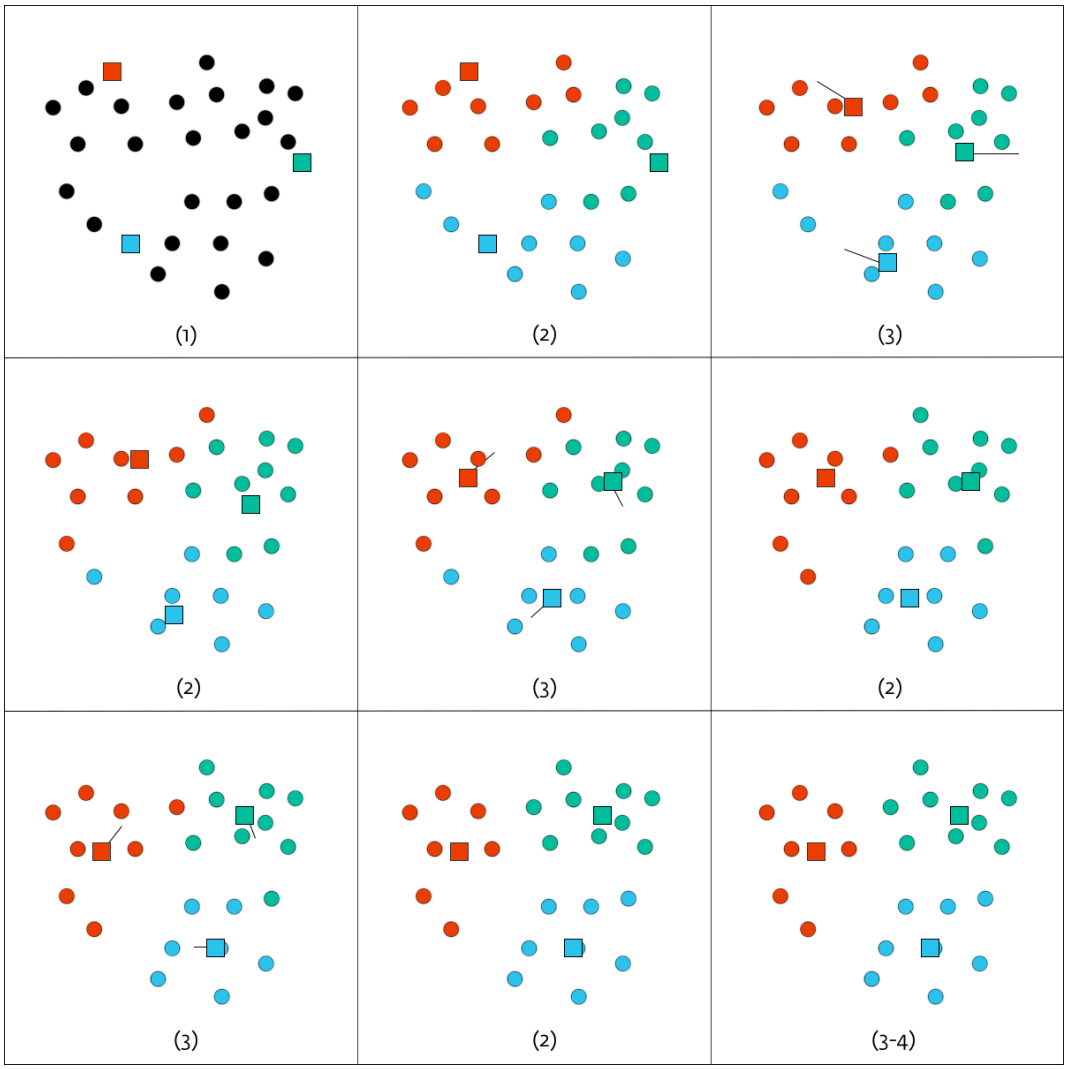


Figure 1.2.: Visualization of k-means method with $k = 3$. Step (1) corresponds to identifying cluster means at random. Step (2) associates each item to the closest mean. Step (3) recalculates the cluster means according to their associated items. Step (2) and (3) are repeated until the means stop changing (4).

tified, the system can for example display the most relevant document(s) of each cluster to increase the user's satisfaction. Some authors argue that it also allows to overcome ambiguity (Agrawal et al., 2009; Skoutas et al., 2010) by automatically identifying the meaningful groups of documents associated to a query, since ambiguous words such as "Jaguar", "Java" or "Flash" may hamper IRSs a lot.

Finally, Manning et al. (2008) and Role and Nadif (2014) both highlight that the labeling of clusters is an important step when the categories are not predefined. The labeling consists in associating a description to the clusters that are created, which seems related, if not similar, to document annotation. The only difference lies in the fact that we need to annotate a group of documents instead of only one. That is, identifying the reason why documents have been gathered in a same cluster. The application of this task in IR as an example is for subtopic retrieval. Bernardini et al. (2009) provide a screenshot (see Figure 1.3) of their system in which the query is "artificial intelligence". Apart from the results for this query, labeled clusters of documents are presented, containing, for example, "computer science", "artificial intelligence research", "john mccarthy", "science and technology", etc. Automatically labeling these clusters is tremendously useful for the user who can refine his/her query.

> Although we decided to use the terms "indexing" and "annotating" interchangeably, we think that "labeling" has a different meaning and it is used accordingly in this thesis. When clusters are labeled, the labels are supposed to be quite abstract (they represent the content of the whole cluster) but most of all they should be composed of

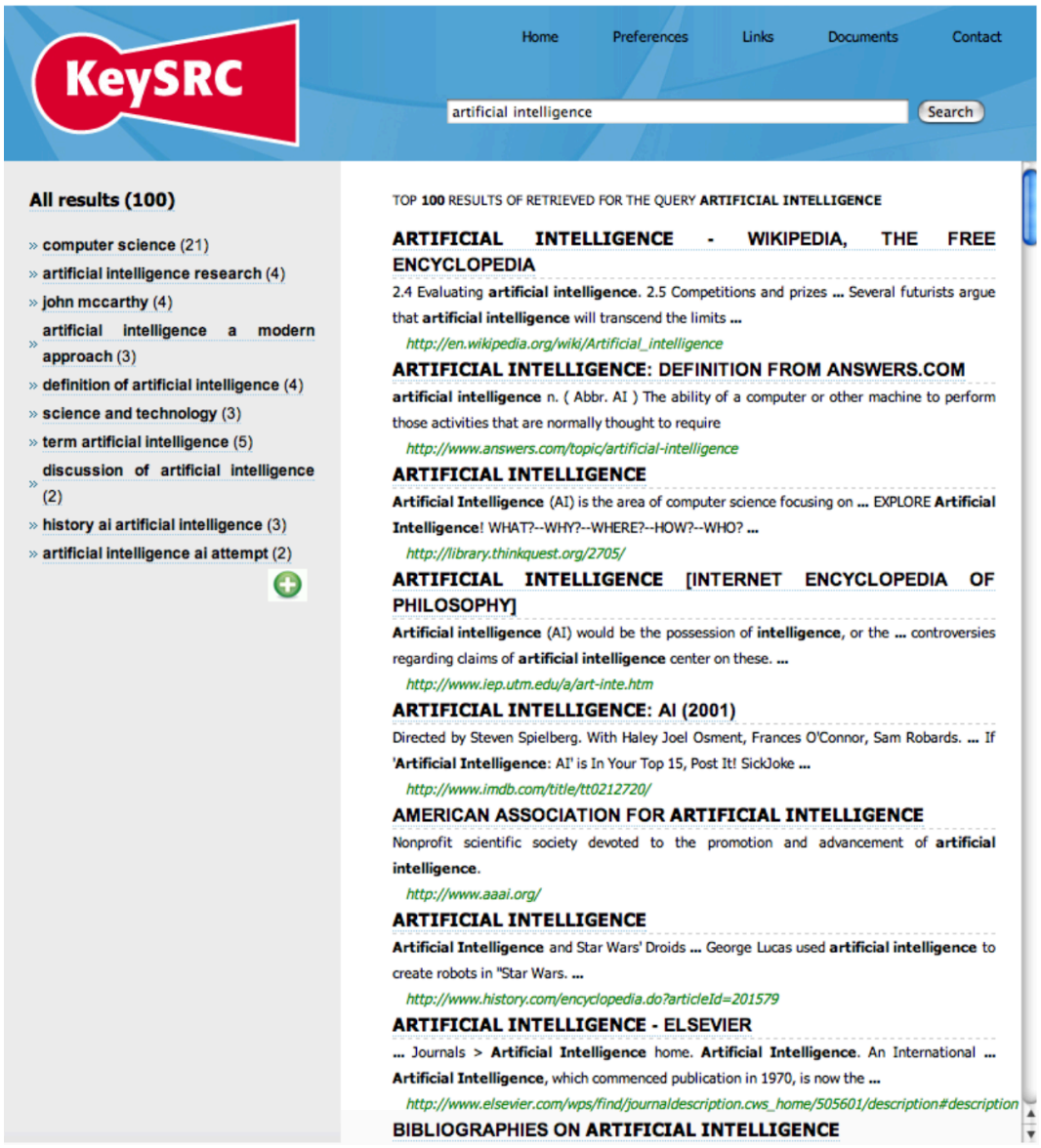


Figure 1.3.: The system of Bernardini et al. (2009) provides labeled clusters on the left to refine the query.

a few words to be instantly understood by the user. There is thus a divergence of objectives: while indexing aims at associating a sharp set or terms to a document, labeling seeks to give the big picture of a cluster. "Concision" is another term used in two different contexts in this thesis. When referring to "annotating", concision impacts further automated processes whereas when associated with "labeling" it seeks to make the label easily understandable.

1.3. The trend of knowledge-based systems

Intuitively, it seems that knowledge is at the basis of any intelligence and thereby of AI. As explained in the general context section, this thesis deals with the use of knowledge in indexing and clustering tasks in order to Figure out the pros and cons of such approaches compared to more classical ones. This section thus aims at laying the foundations of knowledge as it is used in knowledge-based systems.

Russell and Norvig (1995) explain that the so called Turing test requires, among other things, a Knowledge Representation (KR). Indeed, we have to *know things* if we want to *infer conclusions*. The KR is thus the storage of information that a machine would use to pass the Turing test. They also declare that "*all the skills needed for the Turing Test also allow an agent to act rationally, Knowledge Representation and reasoning enable agents to reach good decisions*". Many definitions of KR have been proposed in the literature, one of which the ontology is a famous formal representative (Harispe et al., 2015b; Sy et al., 2012).

A knowledge-based system is therefore a system that makes use of a KR. In several fields, for instance in biomedicine, the presence of knowledge is crucial for the systems to analyze, process, and organize the documents (Ben Abacha and Zweigenbaum, 2015). Recently, a lot of effort has been invested into the creation of such KRs, particularly in the biomedical field which requires a high level of expertise and accuracy (Smith et al., 2007). It seems obvious that the stakes of systems in this field are high as medical decisions or diagnoses may depend on them. As a result, the researchers mainly decided to rely on KR in a quest to improve the quality of their systems and therefore focused on semantics.

1.3.1. Inferring from knowledge bases

The idea of a Semantic Web emerged from Tim Berners Lee who defined it as a Web understandable by machines as well as humans. The idea is to enrich the current Web with knowledge that computers can process, therefore enhancing or facilitating the use of the Internet (Berners-Lee et al., 2001). This paradigm is criticized on the grounds of feasibility matters and, so far, the Semantic Web *per se* is nonexistent. However, there exist many proofs of the use of relying on a knowledge-enriched system in specific domains.

IR has been experienced with the use of KRs and results show a clear improvement compared to term-based approaches (Haav and Lubi, 2001). The main problem exposed regarding classical approaches is that they suffer from synonymy. As an example, a query containing "tumor" may lead to different results from that of "carcinoma", although the terms are synonyms (Giunchiglia et al., 2009; Bhagdev et al., 2008). Stokoe et al. (2003)

explain that ambiguity can also hamper these approaches because even when the query and the results perfectly match, the meaning in each may be different. Besides the terms ambiguity, another downside is the lack of connection among the terms in general. When we think about the terms "dog" and "cat", we imagine a sort of connection between them. Both are domestic animals, mammals, and the cartoons we used to watch in our childhood stressed enough that dogs do not like cats. Those connections we know about cats and dogs are not grammatical (i.e. the words cat and dog are not grammatically related) and refer to something that is not language but knowledge. Then, when a system has the ability to consider some knowledge, such connections, it can outperform classical methods in some cases by retrieving documents that better suit users' needs. Imagine, for example, that there is no perfect match for a given query: documents that are indexed by close terms—i.e. connected to the ones of the query by knowledge—might consitute a good result to be proposed instead. In order to better understand how knowledge is used in those systems, let us define the terminology and the mathematical objects that will be used throughout this thesis.

1.3.2. Terminology and formal definitions

As pointed out by Sy et al. (2012) (in French), the literature is sometimes confused with the use of knowledge-related terms. Although we do not pretend to exhaustively describe each term and its limits, the following gives an overview of the scope of our work regarding KR.

1.3.2.1. Vocabulary, taxonomy, thesaurus and ontology

Vocabulary, taxonomy, thesaurus and ontology all are KRs. What differs among those KRs is mainly the degree of formality of their definitions and their richness. The vocabulary, also referred to as controlled vocabulary, is not formal and simply proposes to explicitly enumerate the terms in an unambiguous and non-redundant way called concepts. This suggests that each concept is unique and has only one meaning within this model. The taxonomy is built upon the vocabulary by adding a hierarchy, that is, parent-child relationships representing the idea of generalization and specialization (the concept MAMMAL is more abstract than the concept CAT). The thesaurus enriches the taxonomy representation by adding other kinds of relationships, such as *related_to*. The ontology differs by its richer formalism and expresses axioms and restrictions (Staab and Studer, 2013). Even though these differences have a great impact on the creation of KRs, it is important to note that they are usually less considered when these KRs are used in IR. In other words, the degree of formality mostly matters for the creators who have to respect it, not directly for the users except through the efficiency and relevance of the results of their systems or the operations allowed with the chosen KRs.

Formally, Maedche and Staab (2001) proposed a definition of an ontology that can be derived as follows (Sy et al., 2012).

Definition 1 ONTOLOGY: *An ontology* θ *is defined as* $\theta = \{\mathcal{C}, \mathcal{R}, \mathcal{H}_C, \text{Rel}, \mathcal{A}x\}$ *such that:*

- $\mathcal{C}$ *is a set of concepts;*

- $\mathcal{H}_C$ *is a taxonomy organizing the concepts* $\mathcal{C}$ *with multiple inheritance;*
- $\mathcal{R}$ *is a set of nontaxonomic relations described by their domain and range notation;*
- $\mathrm{Rel} : \mathcal{R} \times \mathcal{C} \times \mathcal{C} \to \{0;1\}$ *associates each non-taxonomic relation with the pairs of concepts satisfying this relationship. Say* $r \in \mathcal{R}$ *is a non-taxonomic relationship and* $(c_x, c_y) \in C^2$ *are two concepts, then* $\mathrm{Rel}(r, c_x, c_y) = 1$ *if there exists a non-taxonomic relationship* r *between them,* $\mathrm{Rel}(r, c_x, c_y) = 0$ *otherwise;*
- $\mathcal{A}x$ *is a set of axioms that describe additional constraints on the ontology to infer implicit facts.*

As Harispe (2014) notes, the hierarchy $\mathcal{H}_C$ formally expresses the relationship of concepts as it is a non-strict partial order of $\mathcal{C}$. It defines the binary relation $\preceq$ over $\mathcal{C}$, which is

- reflexive: $\forall c \in \mathcal{C}, c \preceq c$,
- antisymmetric: $\forall u, v \in \mathcal{C}, (u \preceq v \wedge v \preceq u) \implies u = v$,
- transitive: $\forall u, v, w \in \mathcal{C}, (u \preceq v \wedge v \preceq w) \implies u \preceq w$.

All the notations relative to the definition on the ontology are kept throughout the thesis. Now that a specific KR has been formally defined, let us explain how it is intensively used in our projects.

1.3.2.2. An overview of similarity measures

The human mind has the innate ability of comparing concepts. In fact, comparing is at the basis of many operations such as learning. When one encounters a situation similar to a previously encountered one, the user

can detect the similarity with the previous experience and use the knowledge acquired the previous time to solve the new problem (Holyoak and Koh, 1987). Therefore AI extensively exploits algorithms that are able to assess *similarities* between "entities" in decision processes, recommender systems or information retrieval systems, to cite a few.

The field of similarity measures (SM) is wide and has been subject to many contributions (Harispe et al., 2014b), pursuing the idea that computing similarities of pairs of concepts is crucial in order to mimic the human thinking. This thesis focuses on graph-based SMs, i.e. SMs that assume that a graph-based KR is available for a given domain—vocabularies and corpus-based measures are thus not concerned. This section aims to provide an insight into what we call similarity measures and some of their definitions. A more comprehensive work on this topic is available in Harispe et al. (2015b).

There are two ways of comparing concepts through a semantic measure in the light of relationships that have been defined or that can be assessed from graph-based KRs. The first one is semantic similarity, the second one is called semantic proximity or relatedness. Semantic Similarity (SS) measures are often associated with the substituability property (the more a concept can be substituted by another, the more they are similar). They only exploit taxonomic relationships $\mathcal{H}_C$ of the KRs. Semantic proximity or relatedness measure, on the other hand, are associated with conceptual evocation: given a concept, what other concepts come to mind (Pedersen et al., 2007). For instance, NAIL evokes HAMMER, TOOTHBRUSH evokes TOOTH or TOOTHPASTE. To this end, the semantic proximity or relatedness uses the whole set of relationships $\{\mathcal{H}_C, \mathcal{R}\}$ provided by an ontology. Harispe et al.

(2015b), following the work of Resnik (1999) and Pedersen et al. (2007) thus define semantic proximity and similarity as follows.

Definition 2 Semantic proximity or relatedness*: the strength of the semantic interactions between two elements with no restrictions on the types of the semantic links considered.*

Definition 3 Semantic similarity*: subset of the notion of semantic relatedness only considering taxonomic relationships in the evaluation of the semantic interaction between two elements.*

In order to compute the SS of two nodes (i.e., concepts) belonging to a graph such as an ontology, several directions have been tried out. Graph traversal is one of them and comes from the graph theory field. The shortest-path is a famous problem that has been exploited and adapted to SSs by Rada et al. (1989), while Fouss et al. (2007) chose to rely on a random walk strategy. Graph-based measures that use the whole set of relationships provided by the graph are called proximity measures. Semantic similarity measures (SSM) are restricted to the *is_a* relationship and need an acyclic graph instead that is provided by the restriction of the Definition 4 below. Let us thus define the restriction of the ontology from Definition 1 applied to obtain θ_{Tax}, the directed acyclic graph that allows us to use most of the SMs.

Definition 4 *The taxonomy (Tax) restriction* θ_{Tax} *of an ontology* θ *is defined as* $\theta_{\text{Tax}} = \{\mathcal{C}, \mathcal{H}_{\text{C}}\}$.

Since the thesaurus differs from an ontology by the absence of $\mathcal{A}x$, the same restriction may be applied when a thesaurus is used. That is, say ω is a thesaurus such as $\omega = \{\mathcal{C}, \mathcal{R}, \mathcal{H}_C, \mathbf{Rel}\}$ and $\theta = \{\mathcal{C}, \mathcal{R}, \mathcal{H}_C, \mathbf{Rel}, \mathcal{A}x\}$ is an ontology, then $\omega_{\text{Tax}} = \theta_{\text{Tax}} = \{\mathcal{C}, \mathcal{H}_C\}$.

Although it has been explained that the SSMs we use rely on taxonomic relationships, we still need to describe how this taxonomy is used to assess the similarity of a pair of concepts.

1.3.2.3. Information Content

The SSMs our work uses are based on Information Theory, which implies the definition of an Information Content (IC) function that aims at expressing the amount of information conveyed by a given concept. Those ICs follow the guidelines of Shannon's information theory (Shannon, 1948), i.e. the more likely a concept is to be used, the less informative it is. From this paradigm, two main classes of ICs emerged: intrinsic and extrinsic ICs. Intrisic ICs solely rely on the graph structure (i.e. the KR model) while extrinsic ones also consider a corpus of texts associated with the KR. In 1995, Resnik (Resnik, 1995) proposed an extrinsic IC, here denoted by $\text{IC}_{\text{Resnik}}(c)$, based on the probability $p(c)$ of encountering an instance of concept c defined as follows.

$$p(c) = \frac{\mathcal{I}(c)}{|I|} \tag{1.1}$$

$\mathcal{I}(c)$ is the set of instances of c, that is to say the number of occurrences of c in a corpus or in a KR—represented by c itself and its descendants. $|I|$ is the total number of instances of all concepts in the corpus. Then,

$$IC_{Resnik}(c) = -\log(p(c)) = \log(|I|) - \log(\mathcal{I}(c)). \tag{1.2}$$

Therefore with this definition, say root is the concept at the root of the KR, $p(root) = 1$ so $IC_{Resnik}(root) = 0$ and the value increases when the concepts are less represented in the corpus or in the KR, i.e. when they become more specific.

Although such an IC captures the specificity of concepts from several sources (KR, corpus), it may suffer from inconsistencies between the corpus and KR data. In other words, the corpus should have a wide coverage of the domain and not be composed of documents that focus on one part of the KR for instance. Otherwise, the representation of a given concept in the corpus and in the KR may diverge, which would lead to discrepancies, e.g. a very specific concept in the KR that appears frequently in the corpus. In some cases, considering these inconsistencies might be desired, but in most cases intrinsic ICs are good estimators instead. As an example, Seco's IC (Seco et al., 2004) is calculated by using the descendants of a given concept $c \in \mathcal{C}$:

$$IC_{Seco}(c) = \frac{\log(\frac{|desc(c)|}{|\mathcal{C}|})}{\log(\frac{1}{|\mathcal{C}|})} = 1 - \frac{\log(|desc(c)|)}{\log(|\mathcal{C}|)}, \tag{1.3}$$

where $desc(c)$ is a set containing c and all of its descendants and $\mathcal{C}$ is the set of concepts of an ontology. This function is bounded on an interval $[0; 1]$; it behaves like extrinsic ICs: the more specific the concept, the higher the IC value. The other ICs used in our work (Sánchez and Batet, 2011; Zhou et al., 2008; Harispe et al., 2015a) all satisfy these properties while providing some

other specificities. For instance, Zhou et al. (2008) propose to consider both the depth of a concept and the number of its descendants to assess its IC. Now that the information concepts convey has been defined, let us explain how it is used in IC-based SSMs.

1.3.2.4. IC-based semantic similarity measures

The idea behind IC-based SSMs is to model the similarity of two concepts by relying on the amount of information they carry and their similarity according to the KR. For example, let us take two pairs of concepts that are equally distant in terms of path in the graph (see Figure 1.4). The pair of less specific concepts (e.g., {BIRD, MONKEY}), i.e. with a low value of IC, should have a lower similarity value the pair of more specific concepts (e.g., {GRAPEFRUIT, PERSIAN LIME}). This rule refines the definition of shortest path similarity measures solely based on the taxonomy. Here, the shortest path is coupled with the specificity of the pair of compared concepts so that the similarity of {MONKEY, BIRD} is different from that of {GRAPEFRUIT, PERSIAN LIME}.

Resnik (Resnik, 1995) is the first to implicitly define the MICA notion that stands for Most Informative Common Ancestor. The MICA is a way to represent the shortest path of two concepts in a DAG. By maximizing an (extrinsic) IC function, he proposed to compute the similarity of a pair of concepts by calculating the IC of the MICA:

$$\mathrm{sim}_{\mathrm{Resnik}}(c, c') = \mathrm{IC}(\mathrm{MICA}(c, c')), \tag{1.4}$$

where c, c' are two concepts. The drawback of such calculation is that some

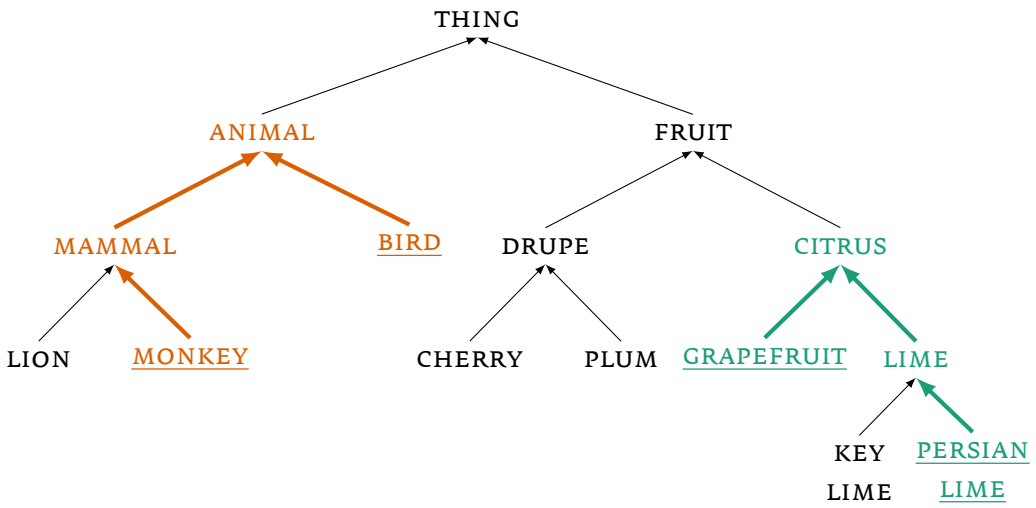


Figure 1.4.: Identical path distance of two pairs of concepts should lead to different values of similarity that take into account the specificity of the pair of concepts.

pairs of concepts would have the same similarity whereas they should not. By assuming the partial order provided by the taxonomic relationship *is_a* of an ontology, let us assume $c' \preceq c$, where the concept c' is a subclass of the concept c according to the definition of the hierarchy $\mathcal{H}_C$. In Figure 1.4, MAMMAL $\preceq$ ANIMAL, MONKEY $\preceq$ MAMMAL, BIRD $\preceq$ ANIMAL. In this case, $\text{sim}_{\text{Resnik}}(\text{MONKEY}, \text{BIRD}) = \text{sim}_{\text{Resnik}}(\text{MAMMAL}, \text{BIRD})$, while we expect that $\text{sim}(\text{MAMMAL}, \text{BIRD}) > \text{sim}(\text{MONKEY}, \text{BIRD})$.

In order to tackle this issue, many authors like Lin (1998) proposed enhanced functions. Lin's SSM, the most used SSM in this thesis, is defined as follows.

$$\text{sim}_{\text{Lin}}(c, c') = \frac{2 \times \text{IC}(\text{MICA}(c, c'))}{\text{IC}(c) + \text{IC}(c')} \tag{1.5}$$

As a result, the similarity depends on the IC of the MICA and the individ-

ual ICs of the compared concepts. The Lin measure is extensively used in our work because it is said to be a neutral measure as compared to some other proposals that meet different requirements that we do not cover in this thesis. For example, the Lin measure respects the property of identity of the indiscernibles that consists in assuming that if the compared concepts have exactly the same properties, then they are identical and the similarity should thus be maximal. That is to say, for two concepts c, c', $\text{sim}_{\text{Lin}}(c, c) = \text{sim}_{\text{Lin}}(c', c') = 1$—except for the root because its IC is 0. In some case, one may prefer the non-respect of this property, leading to the choice of a less neutral SSM as in (Schlicker et al., 2006). Note that we study the impact of the choice of several SSMs by relying on an abstract framework that can instantiate many of them in §2.6.2.

So far, we described the pairwise semantic similarity measures but we often have to compare two groups of concepts. Two strategies have been followed, called direct and indirect groupwise semantic measures. The former consider the sets of features of both sets of concepts while the latter aggregate individual pairwise values. The Jaccard index for example may be applied to create a direct groupwise SSM (Gentleman, 2010). Let us define the set $\text{anc}(c)$ that corresponds to c and all of its ancestors. Say A, B are two groups of concepts and $A^+ = \bigcup_{c \in A} \text{anc}(c), B^+ = \bigcup_{c \in B} \text{anc}(c)$. The semantic similarity of A and B can be computed by

$$\text{sim}_{\text{Jaccard}}(A, B) = \frac{|A^+ \cap B^+|}{|A^+ \cup B^+|}. \tag{1.6}$$

Despite the fact that some other works propose direct groupwise semantic similarity measures (Pesquita et al., 2007; Mistry and Pavlidis, 2008), they

are all hampered by a higher computation time than for indirect ones. Indeed, for any application, the whole list of pairwise similarities can be precomputed and stored for a given ontology. Then, accessing these similarities is in constant time and only the aggregation has to be computed. Additonally, indirect approaches may offer possibilities for optimizations. For example, $\mathrm{sim}(\{c_a, c_b\}, \{c_c, c_d\})$ must not be very different from $\mathrm{sim}(\{c_a, c_b\}, \{c_c\})$ if this similarity is based on an arithmetic mean of pairwise values. This means that when we need to compute many groupwise semantic similarities, we may not have to recompute the whole aggregation but derive it from another already computed similarity, therefore, our choice was to go with indirect SSMs. Apart from classical aggregators (minimum, maximum, arithmetic mean, geometric mean ...), a few more refined ones have been proposed (Schlicker et al., 2006; Pesquita et al., 2007). The Best Match Average (Schlicker et al., 2006) is a composite average between two sets of concepts, here A, B, which we mainly decided to employ throughout this thesis:

$$\mathrm{sim}_{\mathrm{BMA}}(A, B) = \frac{1}{2|B|} \sum_{c \in B} \mathrm{sim}_m(c, A) + \frac{1}{2|A|} \sum_{c \in A} \mathrm{sim}_m(c, B), \qquad (1.7)$$

where $\mathrm{sim}_m(c, B) = \max_{c' \in B}(\mathrm{sim}(c, c'))$ and $\mathrm{sim}(c, c')$ is any IC-based pairwise SSM such as those detailed previously. It is thus the average of all maximum similarities of concepts in A regarding B and vice versa. Its definition is further detailed in §2.4.3.3 along with the several computational optimizations it allows.

1.4. Objectives and context of the thesis

So far, this chapter detailed the importance and utility of indexing and clustering for information retrieval along with the possibilities offered by KRs, particularly the computation of SSs. Nowadays, IR is not limited to Google and its concurrents. NCBI, for example, proposes a federated IRS called Entrez (Sayers et al., 2011) to look for proteins, genes, species or scientific papers. In other words, it relies on several specific—gene-specific, protein-specific, etc.—IRS to retrieve information out of more than 30 databases. This shows that depending on the type of documents we are looking for, we use a different IRS that may rely on different paradigms. The main difference lies in the features that are considered to match a query with the corpus. For text-based IR, documents are annotated with keywords, where the query contains words; for image retrieval, the features are based on patterns of pixels, where the query may be a picture; etc. The emergence of such federated systems shows that there is a need for more generic systems that are able to answer an information need from many sources. The same need for more flexible approaches can be identified in the clustering field as well, as showed by some studies like Chebel et al. (2015), which try to cluster multilingual documents.

The main goal of this thesis is to contribute to the field of knowledge-based systems, especially for indexing and clustering, by promoting the use of KRs. Although KRs are very well used in several fields, we think they are not fully exploited. For example in semantic indexing, people usually use only the concepts for annotating. That is, the methods barely use the taxonomy or other relationships of the ontology and prefer to focus on map-

ping, for example, texts to a set of concepts by using their labels. Despite the very good results obtained with these approaches, we felt something was missing.

As indexing and clustering are key processes for many applications such as IR, we decided to explore the possibilities in these domains. They are two-fold:

- relying on knowledge representations through the use of semantic similarity measures might lead to better predictions or decisions,
- using semantic annotations may allow to build generic approaches instead of creating federated systems.

A whole field of AI is dedicated to the assessment of the similarity of pairs or groups of concepts by relying on the structure of the underlying ontology. These measures are used in various domains, e.g. IR (Lin and Wilbur, 2007) for the computation of the relevance score. The second item is motivated by the fact that relying on semantic annotations should enable to create multi-domain approaches. We noted that many methods in different fields do the same thing, applied to a slightly different kind of data. They use the same machine learning algorithms, on nearly the same features, to the same aim.

In light of the literature, we wonder what impact the use of semantic similarity measures has regarding indexing and clustering. In order to answer, this thesis provides an **extended study of the implementation of SSMs in these processes**. It deeply covers the algorithmic aspect of building systems solely upon SSMs and details an analysis of robustness of a semantic similarity-based technique towards numerous SSMs. Most of all, the rele-

vance of an SSM-based indexing system is tested through an international large-scale indexing challenge.

Our application of this paradigm to the clustering area rose another question: how to label the clusters? More generically, is there a difference between annotating a single document and annotating a group of documents? If so, to what extent? We propose to deal with this question by adapting our indexing technique and suggest **a semantic summarization algorithm** that aims at factorizing a group of concepts into a meaningful summary. This approach has been designed and tested in the context of cluster labeling.

Finally we want to study the role of the user in these two processes. In the literature, users are requested in the last step of both indexing and clustering. Therefore we conduct a study on **the consequences of (not) relying on the user** by creating fully or semi-automatic approaches and comparing them.

> This thesis is an attempt at the creation of generic indexing and clustering approaches, in tune with one of the goals of the Semantic Web: "*The Semantic Web provides a common framework that allows data to be shared and reused across application, enterprise, and community boundaries*"[4].

1.5. Chapter outlines

This thesis focuses on two areas of research, namely indexing and clustering. Both of them are investigated within a semantic context: more specif-

[4] http://www.w3.org/2001/sw/

ically, by relying on semantic annotations available for the documents and by using semantic similarity measures. This chapter has broadly introduced these areas and defined the vocabulary, the mathematical objects and other definitions that will be useful throughout the thesis.

A detailed overview of existing approaches in this chapter would have been confusing due to the richness of each explored area. As a result, each chapter starts with a thorough review of the previous works, their upsides and their limitations.

Chapter 2 is dedicated to the semantic indexing field. We present the comprehensive work behind USI (User-oriented Semantic Indexer). This includes the definition of objectives, the modeling of these objectives and their implementation by using an optimized heuristic. We also explain the limits of this approach, particularly the fact of relying on a user, and we propose several options to tackle them. We conclude by presenting several real-life applications, one of which is our participation to BioASQ 2015, a large-scale semantic indexing challenge.

Chapter 3 is an extension of the work in Chapter 2. We follow the same idea of proposing a generic approach for a common task—here, clustering and cluster labeling—on the basis of semantically annotated documents. While Chapter 2 shows several direct applications of our approach, this chapter explores how deeper adaptations can allow to create novel approaches in close fields. To do so, we build a clustering tool for generating a benchmark for the evaluation of hierarchical clustering of semantically annotated documents. We then evaluate our results by using this benchmark. We also expose a more general thought on potential uses of hierarchical clustering and labeling as we propose it.

The conclusion chapter ends this thesis by summarizing our contributions in indexing and clustering. More specifically, it provides more insight on the impact and utility of the use of semantic similarity measures in those contexts. It also suggests novel conclusions regarding the creation, feasability and relevance of generic systems. After detailing the general limits of our work, the chapter gives some research perspectives in the domain of semantic applications, particularly those relying on semantic similarities.

2

Semantic indexing

Contents

2.1. Abstract

This chapter focuses on the association of documents with semantic annotations which describe them. While the literature proposes ad hoc techniques that are highly dependent of the application context, we build a generic method that is applicable to any semantic domain. To do so, we explore the sole use of semantic similarities through our tool, USI.

CONTRIBUTIONS RELATED TO THIS CHAPTER

Fiorini, N., Ranwez, S., Harispe, S., Montmain, J., & Ranwez, V. (2015). USI at BioASQ 2015: a Semantic Similarity-Based Approach for Semantic Indexing. In Working Notes for the Conference and Labs of the Evaluation Forum (CLEF), Toulouse, France.

Fiorini, N., Ranwez, S., Montmain, J., & Ranwez, V. (2015). USI: a fast and accurate approach for conceptual document annotation. BMC bioinformatics, 16(1), 83.

Fiorini, N., Ranwez, S., Montmain, J., & Ranwez, V. (2014). Coping with Imprecision During a Semi-automatic Conceptual Indexing Process. In Information Processing and Management of Uncertainty in Knowledge-Based Systems (pp. 11-20). Springer.

Fiorini, N., Ranwez, S., Ranwez, V., & Montmain, J. (2014). Indexation conceptuelle par propagation. Application à un corpus d'articles scientifiques liés au cancer. In CORIA 2014, COnférence en Recherche d'Information et Applications (p. 187), France.

A generic semantic indexer, USI: http://usi.nicolasfiorini.info

A user-oriented biomedical semantic indexer: http://bio.usi.nicolasfiorini.info

A user-oriented movie semantic indexer: http://movies.usi.nicolasfiorini.info

2.2. Related work

In Chapter 1, we discussed the growing need for semantic annotations on which rely many tasks such as conceptual information retrieval or recommendation. Semantic indexing consists of associating concepts from a knowledge base to documents, thus describing their contents. To face the overwhelming amount of documents to be annotated with concepts of ever growing ontologies, automated solutions are required. For example, PubMed has been gathering more than 1,000,000 new papers per year since 2011[1], while the MeSH[2] size is growing yearly, reaching more than 27,400 descriptors (or concepts) in 2015. Manually establishing a list of concepts correctly characterizing a document has always been challenging because of the expertise required both in the domain of concern and the afferent ontology used to annotate it, but the number of documents and the size of the ontology now make this process almost impossible. In order to automate the process, many researchers investigated the methods that can suggest annotations for a document. Those methods rely on document features that are mostly related to the type of the documents. Some works were proposed for annotating images (Carson et al., 1999; Zhang et al., 2014), texts (Jimeno Yepes et al., 2012), audio documents (Turnbull and Barrington, 2008) or videos (Tseng et al., 2008). These studies show how useful conceptual annotations are for efficient data-driven applications. They all proceed with the same general steps: (i) analyzing the content of the document (through text-mining, sound analysis, sequencing etc.), (ii) mapping extracted information to an ontology and (iii) defining among the list of potential concepts those that are the most relevant.

[1] http://www.michaeleisen.org/blog/?p=1654
[2] Medical Subject Headings, the thesaurus used to annotate the papers on MEDLINE

2.2.1. Extracting concepts from documents

Conceptual approaches have proven efficient in several domains, particularly in the biomedical field where conceptual indexing is used for information retrieval (e.g. to find scientific publications by avoiding ambiguous terms, to identify protein-protein interactions by comparing gene annotations, etc.). This is why in this domain we observe many ontologies that have been designed over the last decades (Gene Ontology, Medical Subject Headings, SNOMED CT, etc.) to annotate a wide range of documents: scientific papers, genes, proteins ... However, using ontologies to automatically annotate documents is a difficult task in terms of computing power, sometimes seen as a fuzzy multi-label categorization problem (Névéol and Shooshan, 2009). The challenge lies in the fact that selecting the optimal set of concepts (of unknown size) from an ontology to annotate a document is, for non trivial criteria, an NP-complete problem that thus requires an exponential computation time to find an exact solution.

The emergence of a lot of dedicated semantic annotation methods is then not surprising. MetaMap (Aronson, 2001) is one example, for which the basic idea is to filter textual contents and map them to concepts of an ontology. Obviously, a mapping based on a perfect textual match with the labels associated to the concepts would not be sufficient because of the polysemy and synonymy in the language. Aronson (2001) explains that ambiguity is the main problem that MetaMap faces. MaxMatcher (Zhou et al., 2006b) tackles this problem by including a disambiguation algorithm based on the words that surround the extracted textual content in the text. Jonquet et al. (2009) proposed the NCBO[3] annotator Web service that can be

[3]National Center for Biomedical Ontology

used to annotate texts with concepts from the UMLS[4] and NCBO BioPortal ontologies. The underlying method uses the structure of ontologies it relies on to improve the conceptual annotations of a text. Neves and Leser (2014) provide a comprehensive survey of concept extraction for biomedical text documents, including the specificity of each method. Among them, Machine Learning methods are particularly prominent.

2.2.2. The rise of Machine Learning and its limits

In this semantic indexing context, Machine Learning (ML) is famous for being able to provide good solutions quickly thanks to the definition of features and the learning—mostly supervised, in this field— of their behaviour on a learning set. Some ML approaches have been applied to the problem of annotating biomedical papers with the MeSH.

2.2.2.1. Description of ML approaches

The aim, when using ML to index documents, is to find relevant features to accurately predict concepts representing a given document content. Several ML approaches have been experimented such as gradient boosting (Delbecque and Zweigenbaum, 2010) or Reflective Random Indexing (RRI) (Vasuki and Cohen, 2010). The choice of the considered features is crucial and constitutes a key difference in those approaches (see §2.2.2.3).

Delbecque and Zweigenbaum (2010) highlight that co-authoring and citations contribute to some of the top features of their approach. This work

[4]Unified Medical Language System

shows that scientific paper annotations can benefit from co-authoring and citation information. Some of those features are *FreqDistConcept* and *RefFreq*. The former stands for the number of occurrences of a MeSH concept in cited papers divided by the number of distinct concepts annotating them. The latter represents the proportion of cited papers in this document annotated by this concept. Another feature, *MeanFreq* is the average frequency of a given concept annotating previous publications of co-authors.

The approach presented in Vasuki and Cohen (2010) is a k-NN (k-Nearest Neighbors) approach like most hybrid approaches presented in the next section. They suggest to use an alternative to the Latent Semantic Analysis (LSA) method called Random Indexing (RI). Let us first explain how LSA models work in general before exploring their limits and how RI can overcome them. LSA—or LSI, for Latent Semantic Indexing in the context of Information Retrieval—consists of starting from a sparse matrix F where rows w_i represent words and columns c_j represent their context, e.g. documents or phrases where those words appear (Figure 2.1). Each cell $F_{i,j}$ thus contains the frequency of occurrence of w_i in the context c_j. As a result, F contains a vector of occurrences for each term of the corpus and for each document (in a document-based co-occurrence matrix). This allows to compute the similarity of terms-documents by relying on well-known mathematical operations based on vectors such as the cosine similarity. Because the matrix quickly becomes unusable as the corpus size increases, LSA relies on the Singular Value Decomposition (SVD) approach to factorize F and to benefit from a smaller matrix size. The main problems of the factorization is that it is costly in terms of computation time and it has to be done each time a new entry is added to the corpus.

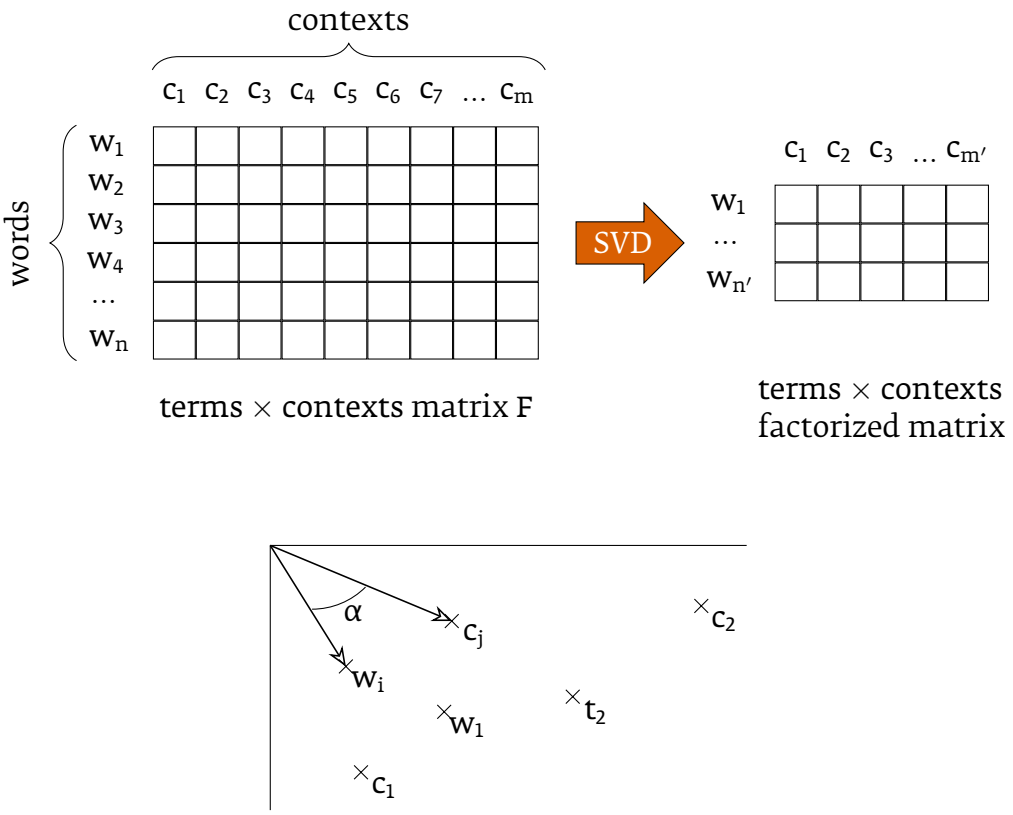


Figure 2.1.: Overview of the Latent Semantic Indexing (LSI). A sparse matrix of terms × contexts is created, each cell containing the number of occurrences of the corresponding word in the corresponding context. It is then factorized by the singular value decomposition (SVD). This allows for a reduction in size of the matrix before computing the term-term, context-context or term-context similarities in a vector space.

The Random Indexing technique described by Sahlgren (2005) consists of associating a random vector of dimension d with each context (co-occurring word or document) that contains a few $+1$ and -1 randomly distributed among many 0. Each vector thus characterizes a context and these vectors are summed up for each word, that is for each row of F. Consequently, a matrix F' is created, where each word is represented by a context vector of dimension d. The motivation of RI is to get a *nearly* orthogonal matrix of F, as Hecht-Nielsen (1994) demonstrated that in a high-dimensional space, there are many more orthogonal directions than truely orthogonal ones. By choosing random directions in the high-dimensional space, one can thus approximate orthogonality. The dimensionality of the stored data is thus controled and adding new documents to the corpus only requires to create a new random vector, sum it to the contexts of words occurring in it that already exist in F' and add the new words to F'. This approach benefits from a good scalability while erasing the issues of LSA. Finally, Cohen et al. (2010) explain that RI fails to derive semantic connections between words when they do not directly co-occur. For example, if "tumor" and "carcinoma" do not directly co-occur, RI would not correctly assess their similarity. Vasuki and Cohen (2010) propose to use a method called RRI, an iterative version of RI that can *learn* implicit associations even when the terms do not directly co-occur. The first step is the same as in RI, i.e. creating random document vectors. Then, each term is associated with a term vector that is the sum of all document vectors of documents it occurs in. RRI uses the term vector to recalculate the document vectors and the term vectors iteratively. The cosine similarity then allows to compute the similarity of documents. Neighboring documents and their associated MeSH terms are retrieved. Finally, the concepts are ranked according to a score

that is the sum of similarity between the documents to which they are associated with the target document (represented by its abstract).

These approaches rely mostly on ML solutions to index documents. Some authors propose to combine concept extraction with ML to provide enhanced annotations.

2.2.2.2. Towards hybrid methods

Concept extraction *per se* is an interesting idea that can actually be coupled with other acknowledged approches to annotate documents. The Medical Text Indexer (MTI) (Aronson et al., 2004) for example is built upon a two-fold strategy to fulfill this task. During a phase A, it identifies candidate concepts while in phase B those concepts are associated with a relevance score and ordered accordingly. This approach is similar with every hybrid approach, although the method used for each phase may vary.

The phase A of MTI is actually made of a concept extraction tool presented in §2.2.1: MetaMap. The title and abstract of the paper to be annotated are given as input to MetaMap to extract UMLS concepts. UMLS being broader than the MeSH, an algorithm, Restrict-to-MeSH[5], is applied to convert this list into MeSH headings.

MTI enriches the set of identified concepts by using a k-Nearest Neighbors approach. Indeed we can make an intuitive assumption that documents that are similar in content in the corpus are likely to be annotated the same way. Therefore, when annotating a new document, it seems legitimate to look for similar documents that are already annotated and use those

[5] http://ii.nlm.nih.gov/MTI/Details/RTM.shtml

annotations to enrich the set of candidate concepts. PMRA (PubMed Related Articles) (Lin and Wilbur, 2007) is an algorithm designed to recommend papers to readers of PubMed. This solution is used by MTI to identify the k-NNs of the considered documents. When someone reads a paper on PubMed, PMRA proposes a list of similar papers based on a similarity calculus between articles. This algorithm relies on two metrics. There is a text similarity based on the textual content of papers—for example, the use of words or keywords—and a semantic similarity of their respective MeSH annotations. Since the target document provided as input in MTI is not annotated yet, only the first calculus (text similarity) can be used during k-NN identification and PMRA has been modified in this way. We further refer to this modified version of PMRA as PMRA*. Several methods propose different algorithms for the phase B but still use the PMRA* algorithm to identify the k-NNs (Huang et al., 2011; Mao and Lu, 2013; Mao et al., 2014).

The goal of phase B is to filter the candidate concepts and to propose only the relevant ones. To do so, MTI uses several outputs from phase A. For example, each concept extracted from the text with MetaMap is given with a confidence score according to the quality of the word-concept term match and the possibility of ambiguity. Also, when PMRA* returns similar documents, they are associated with a neighboring score. Those scores are used to assess the relevance of each concept. The results are then ordered, a cut-off is applied and the list is output. ML has been extensively used to provide the results of phase B.

Researchers compared several indexing frameworks such as k-NN, concept extraction and vector space models. Yang (1999) and Trieschnigg et al. (2009) came up to the conclusion that k-NN approaches are the only scal-

able methods that give good results. The k-NN discussed in those papers is very simple: each neighbor acts like a voter for each concept it is annotated by and the algorithm counts how many times the concepts have been voted up. Concepts are then ranked according to those votes. MTI has been compared to such a k-NN approach in Trieschnigg et al. (2009) and showed less good results than such a simple method. However, more recent work on hybrid systems—that is, relying on concept extraction besides the k-NN part—gave better scores (Huang et al., 2011). This new system provides both efficiency and effectiveness. It relies on MetaMap and PMRA* for phase A, and it uses a LTR (Learning-To-Rank) algorithm (Cao et al., 2007) to rank the concepts. In this implementation of LTR, the features used to determine the score of each concept are:

- concept frequency in k-NNs, like in Trieschnigg et al. (2009);
- word unigram/bigram overlap between a concept label and the title (and abstract) of the paper;
- query likelihood scores between MeSH terms and words in the title or abstract by using information retrieval models;
- translation probability between two languages (author language versus expert language).

This approach only ranks the concepts, a cut-off must be applied to provide a result to the user. They proposed a list of 25 concepts. Their approach is more elaborated than a simple k-NN frequency-based one, thus leading to better annotations.

Later on, the NLM team behind the work in Huang et al. (2011) improved their system for participating at the BioASQ 2013 and BioASQ 2014 chal-

lenges. In 2013, they updated the LTR algorithm, they implemented dynamic cut-offs for the number of documents to retrieve with PMRA* and for the number of concepts to suggest (Mao and Lu, 2013). They also considered the baseline called MTIFL provided for the challenge. This baseline is the draft output of MTI before human experts modify it by adding/removing MeSH terms. In 2014, they updated the LTR algorithm again, improved the dynamic threshold of the number of concepts to return and added a new feature based on the results of a binary classifier (Mao et al., 2014). Some participants in the challenge built slightly similar methods (Liu et al., 2014), while other proposed multi-label classification approaches (Papanikolaou et al., 2014).

2.2.2.3. Limits of ML approaches and general drawbacks

Although hybrid approaches showed more promising annotations, they suffer from several limits. Jimeno-Yepes et al. (2012) note that ML methods behave differently depending on the problem and the dataset. For example, annotating a paper with the MeSH and the Gene Ontology would require to use different ML approaches, with different learning sets. They hence suggest to use meta-learning, that is the system learns to choose the ML approach depending on the input data.

This improvement seems to overcome the specificity problem, but at the cost of a high expertise needed to implement such a system and an extensive learning phase. This learning phase is also a limiting requirement regardless of the ML algorithm. Authors of tools including a learning phase report to train it on huge amounts of training data compared to the number of test documents. Jimeno-Yepes et al. (2012) mention a learning set

of 200,000 documents for 100,000 tests. Vasuki and Cohen (2010) report a training on more than 9,000,000 documents for a test set of 200 papers. Névéol and Shooshan (2009) used 100,000 articles to train the system for 100,000 tests. Every time, the size of the training set seems important compared to that of the test sets (it is at least the same, sometimes much higher). Besides, when the number of parameters modelled by the system is high, it is prone to overfitting (i.e. the system models very specific features of the training set). This effect is even more important after an extended training phase, so performance on bigger test sets should be evaluated as well as the predisposition of these systems to overfitting.

The main limit of all approaches presented in this thesis is their lack of genericity. As a result, this whole section is focused on the indexing of biomedical papers, while semantic annotations are not limited to this field. More generally, almost all described approaches are text-specific whereas this task is useful to annotate many kinds of documents. So far, there has been no work on a generic approach for annotating documents irrespective of their type.

2.2.3. Evaluation datasets and metrics

The evaluation of indexing methods consists of comparing their results with gold standard ones. These gold standards are made by experts in the field, in general by the same team that actually annotates documents every day at the NLM library. The aim is to assess how close the proposed annotation of each paper is from the gold standard, and by extension, evaluate the results on a whole dataset.

2.2.3.1. Benchmarks

Although several methods were proposed for concept extraction or document annotation, there has been a lack of proper benchmarks before the first BioASQ challenge in 2013. Usually, in order to evaluate their system, researchers created a gold standard dataset extracted from PubMed and compared their results with the gold standard. Several small datasets have been created and shared such as NLM2007 (Aronson et al., 2004) or L1000 (Huang et al., 2011). However, some authors evaluated their results on datasets that are not accessible anymore as in Trieschnigg et al. (2009). Finally, there are even studies that do not share any dataset (Jimeno-Yepes et al., 2012), making the comparison to their approach impossible. Indeed, the source code of annotation tools is rarely available so it is impossible to test the efficiency of older approaches on new benchmarks. Note that this is quite contradictory with the initial aim of such work: building new methods in order to help the indexing community and make proper advances in this domain. Since we encountered these problem, we decided to make all our work available online (softwares, results, datasets, etc.). In order to evaluate a new system, one should thus choose between using a very small benchmark (200 tests for NLM2007, 1,000 tests for L1000) or make a new dataset knowing that the results would not be comparable with previous work.

BioASQ bridges the gap by bringing proper evaluation sets (with a correct size) and by allowing teams to submit results obtained with several variants of their tools during several months and compare their results. After the BioASQ 2014 edition, Mao et al. (2014) made available a 5,000 tests dataset from BioASQ named BioASQ5000 and the result of their method

MeSH Now, winner of the 2014 challenge. Therefore, it is now easy to compare a new method to state-of-the-art ones by using this dataset and MeSH Now results.

2.2.3.2. Metrics

Several classical measures have been proposed in the literature for evaluating results of automatic annotation against a gold standard. Precision, recall and F-measure are the most common ones as in Huang et al. (2011). Say the annotation $G \subset \mathcal{P}(\mathcal{C})$ is the expected result, which is the gold standard for a document, and $O \subset \mathcal{P}(\mathcal{C})$ the observed result. Precision P is the fraction of the set of proposed concepts that is present in the gold standard, P is maximal when the observed annotations are all in the gold standard:

$$P = \frac{|G \cap O|}{|O|} \tag{2.1}$$

and recall R is the fraction of the gold standard recovered by the method, R is maximal when all annotations in the gold standard are in the observed annotations:

$$R = \frac{|G \cap O|}{|G|}. \tag{2.2}$$

The F-measure F is the harmonic mean of precision and recall thus providing a global assessment of the result, that is

$$F = \frac{2 * P * R}{P + R}. \tag{2.3}$$

Since the outputs of automatic indexing used to be proposed as a ranked list, authors also used the MAP (Mean Average Precision) to emphasize the

importance of the order (Trieschnigg et al., 2009; Huang et al., 2011). This measure relies on the computation of an average precision (AP) for each document d of the test set. AP is defined as follows:

$$AP_d = \frac{1}{|G|} * \sum_{t=1}^{n} (P_t * I(t)), \tag{2.4}$$

$$P_t = \frac{|G \cap O_t|}{t}, \tag{2.5}$$

where $t \in \mathbb{N}$ is a position in the results of size n, P_t is the precision at this specific position, O_t is the set of terms cut after the position t. $I(t)$ is an indicator function for which value is 1 if the term at position t is relevant, that is it belongs to G, 0 otherwise. AP is thus computed for a finite list of annotations, which means that a cut-off is applied. The MAP is then computed for the whole test set D with

$$MAP = \frac{1}{|D|} \sum_{d \in D} AP_d \tag{2.6}$$

Although those measures represent the metric that were mostly used for evaluating automatic annotation, some papers used several variations. Gay et al. (2005), for example, use the F-measure as presented above as well as the F_2-measure that puts more weight on recall. In fact, a general expression of the F-measure exists:

$$F_\beta = \frac{(\beta^2 + 1)P * R}{\beta^2 P + R} \tag{2.7}$$

where $\beta = 2$ in the case of the F_2-measure and $\beta = 1$ for the classical F-measure.

When annotating documents with concepts that come from a structured knowledge representation, precision, recall, MAP and F-measure may seem unadapted. In fact, those measures do not take into account the underlying structure of the concepts. For example, say a paper is automatically annotated MAMMAL while the expected concept was DOG. This annotation is not perfect, but we know with the MeSH that a dog is a mammal. Therefore, giving a precision of 0 for this annotation seems inappropriate as it does not capture the fact that MAMMAL is a much better choice than the completely out of scope concept BICYCLE for instance. As a result, it is pretty hard to compare the results of different approaches that are evaluated according to these metrics as they do not reflect well their quality in this specific semantic context. Some authors thus investigated better fitted metrics as in Névéol et al. (2006). They conclude that semantic similarities give better scores than precision and recall because they consider the structure. They advise to use them when evaluating automatic semantic annotations. However, to the best of our knowledge, no paper followed up with this idea until the BioASQ challenge.

The BioASQ challenge proposes two evaluation metrics. They call them flat and hierarchical. The flat measure is a simple F-measure, while the hierarchical one called LCA-F follows the concepts presented in Kosmopoulos et al. (2013). It is a sort of F-measure that takes into account the hierarchy of concepts by considering their ancestors. The idea is the same as in Névéol et al. (2006), that is, using the relations among concepts to provide a better assessed score.

2.3. Motivation & positioning

There are several points that we question when semantically annotating documents. The methods in the literature propose to output an ordered list of concepts for a target document. We wonder whether order matters and, more generally, if ordering is the right way to propose an annotation. Basically, scoring each concept individually may lead to some biases in the annotations (see §2.4.2.1).

Studies state huge amounts of training data that is needed to get their respective results. There is also the risk of overfitting when numerous parameters are extensively trained. Apart from those downsides, every tool proposed in the literature is type-specific, e.g. it only applies on biomedical documents. Some may be extended to more general textual documents, but there is no work dealing with any type of document. Even in text-based application, Huang et al. (2011) report that full texts are not always available, making the annotation task more difficult.

Finally, to the best of our knowledge, there is no study reporting anything about algorithm complexity or running times for this task except some scalability tests in Trieschnigg et al. (2009). Considering the increasing number of documents to be annotated, those parameters become crucial.

2.4. USI: a generic User-oriented Semantic Indexer

The following sections describe the steps and choices we made to create a k-NN-based annotation framework and its implementation in an optimized

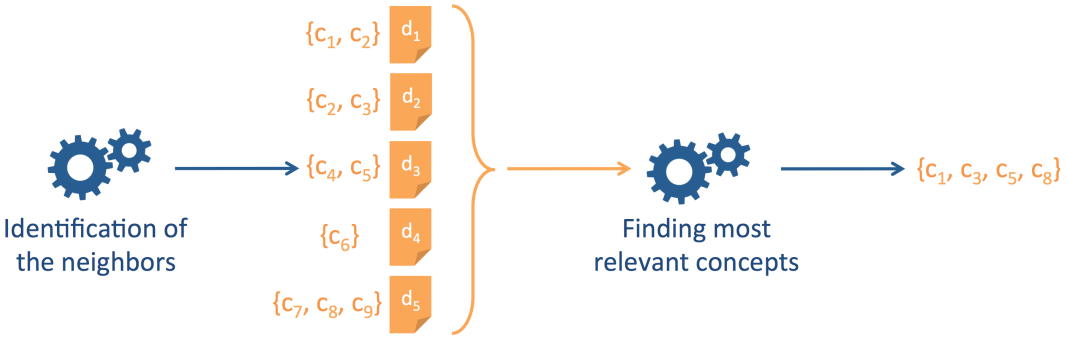


Figure 2.2.: The two main processing phases of USI. First the neighboring documents are found by using an information retrieval system. Second, annotations of the neighbors are processed to find the most relevant concepts among them.

algorithm. In order to keep the description of this work consistent, we rely on an example that will be this chapter's main theme: the annotation of biomedical papers with MeSH descriptors.

k-NN approaches are motivated by the assumption that documents close in content should be close in their annotations. Their implementation thus consists of identifying, thanks to approaches based on Natural Language Processing (NLP) for example, a neighborhood of close documents and use those neighbors to annotate the current document. Figure 2.2 depicts the pipeline USI follows for annotating documents.

2.4.1. Selection of neighboring documents

Identifying close neighbors of the document to be annotated can be seen as an information retrieval task, since the aim is to find documents in a collection that match with a query. Here, the query would be the content of the document to annotate—or its keywords, or any representation of its content. This process relies on the content of the document, so it is type-

specific: IRSs based on text work differently from the ones based on audio documents. We further discuss the genericity of neighbor documents selection in section 2.5.1 where the role of the user is emphasized. This step can be assimilated to an IR task and many tools already exist to do this job. Although it is absolutely needed in k-NN approaches, it is quite independent from the proper annotation process. Therefore, we had to make a choice on the system to use to get documents similar in content. For our biomedical paper annotation example, we chose a state-of-the-art approach, PMRA*, the textual variant of PMRA described in §2.2.2.2.

2.4.2. Modeling the objectives

Let us assume a neighborhood of documents has been identified for the target document. This neighborhood is a set K of k documents. Each document $d \in K$ is annotated by a set of concepts $A_d \subseteq \mathcal{C}$ such as

$$\begin{aligned} A : K &\mapsto 2^{\mathcal{C}} \\ d &\mapsto A_d. \end{aligned} \tag{2.8}$$

The set of annotations of all documents of K is a family of sets denoted $\mathfrak{A}_K = \{A_d | d \in K\}$. Once the neighbor documents are gathered, the system identifies the concepts that best summarize their annotations. Testing all subsets of concepts would lead to a solution having an exponential time complexity that would not be able to deal with large ontologies. We hence limit our search space to the subset of concepts A_0 defined as $A_0 = \bigcup_{A_d \in \mathfrak{A}_K} A_d$. The optimal solution to the indexing problem, A^*, is obtained by maximizing an objective function $f(A)$, that is $A^* = \operatorname{argmax}_{A \subseteq A_0}(f(A))$. The next sections describe the choices we made to model this objective function.

2.4.2.1. Individual concept scoring versus annotation scoring

Classically, the concepts are individually scored according to several features: Is this concept common in the neighborhood? Is this concept ouput by several methods, e.g. both NLP and the neighborhood? The concepts are then ranked according to their score(s). This approach however may produce redundant annotations. Say a paper describes a study about carcinoma affecting dogs. An NLP analysis would certainly extract the concepts DOG and CARCINOMA as top concepts among others. Say the documents in the collection, therefore in the neighborhood, do not discuss the specific case of the dog but carcinoma among mammals in general. Then, CARCINOMA would be highly rated since both NLP and k-NN exploration return it. DOG and MAMMALS would follow in the list. The order would basically depend on the weight given to NLP or k-NN. A dog is a mammal, so returning the two concepts is a form of redundancy that the user would like to avoid. This is due to the fact that each concept is scored independently of other proposed concepts and often regardless of the ontology structure, leading to possible annotations containing parents and their children concepts.

Set scoring may allow to overcome this redundancy. Instead of scoring each concept, the idea is to globally evaluate a set of concepts considered as a potential annotation of the document based on the neighborhood, concept extraction, etc. Besides, it allows to assess the *synergy* of concepts altogether. That is, for example, comparing the sets {DOG, CARCINOMA} with {DOG, MAMMALS} w.r.t the objective function. The way we chose to assess the quality of a group of concepts is the semantic similarity. As described in §1.3.2.4, many semantic measures are available. Groupwise measures allow us to compare two sets of concepts, for instance an annotation sug-

gested by the method with any element of $\mathfrak{A}_K$. It thus seems that semantic similarities would provide a more accurate annotation, certainly at the cost of higher computation time. Indeed, finding an optimal set of concepts is obviously more complex than individually scoring and ranking the concepts but also closer to the intuitive idea of evaluating an annotation suggestion. USI follows the set scoring strategy at the cost of an algorithm complexity that will be discussed and optimized in §2.4.3.3.

2.4.2.2. The consistency criterion

Our approach only relies on the neighbor documents, so the system must be able to score the annotation it proposes by relying solely on those neighbors. We model the consistency of an annotation by its average similarity with the neighbors annotations. The idea is that if an annotation is very close to that of all documents in K, then it is likely to be accurate. The similarity between the proposed annotation and the neighbor annotations can be modelled by using a groupwise semantic similarity.

Let us generically denote $\mathrm{sim}_g(A, B)$ a groupwise semantic similarity returning a similarity score in $[0; 1]$ for two sets of concepts A and B. We want to explore the search space A_0 to find the set of concepts that is the most similar to the neighbor annotations. This is modelled by the following consistency objective

$$\text{consistency}(A) = \frac{1}{k} \sum_{A_d \in \mathfrak{A}_K} \left(\mathrm{sim}_g(A, A_d) \right). \tag{2.9}$$

We do not fully discuss the choice of the measure in this section (see §2.6.2).

Note, however, that depending on the properties of the function chosen to assess the similarity, the output may suffer from several downsides. For example, let us analyze this very naïve similarity function:

$$\mathrm{sim}_g(A, B) = \max_{\substack{a \in A \\ b \in B}} \Big(\mathrm{sim}_p(a, b) \Big). \tag{2.10}$$

where $\mathrm{sim}_p(a, b)$ is a pairwise similarity function between concepts a and b. In that case, many optimal solutions A^* exist since as soon as one concept of A_d is present in A^*, their similarity is maximal. Therefore it suffices for A^* to contain at least one concept of each A_d of the neighborhood to be an optimal solution. Obviously, such a groupwise similarity is not satisfying as A_0, or even $\mathcal{C}$ would be optimal solutions while being uninformative due to a lack of accuracy and a high redundancy. This emphasizes the importance of choosing $\mathrm{sim}_g(\cdot)$ and $\mathrm{sim}_p(\cdot)$ carefully.

2.4.2.3. The concision constraint

The optimal annotation should be both consistent with the neighborhood and concise. These two criteria may bring into focus the precision and recall metrics for evaluation in information retrieval. By returning many concepts we have more chances of having good recall but often at the cost of a low precision and vice versa. We define concision as a penalty on the number of concepts in the annotation:

$$\mathrm{penalty}_c(A) = \mu |A|, \tag{2.11}$$

where $\mu \in [0;1]$ is a parameter controlling the importance of the constraint. Let us write the full objective function $f(A)$:

$$f(A) = \text{consistency}(A) - \text{penalty}_c(A) \tag{2.12}$$

$$f(A) = \frac{1}{k} \sum_{A_d \in \mathfrak{A}_K} \left(\text{sim}_g(A, A_d) \right) - \mu |A|. \tag{2.13}$$

In fact, here μ represents the decrease of the average similarity that the user accepts for removing a concept. Here is an example of two candidate sets: $A_1 = \{\text{DOG, CAT, RABBIT, MAMMALS}\}$ and $A_2 = \{\text{DOG,CAT,RABBIT}\}$. Say A_1 has a consistency score of 0.82 while A_2 has 0.79. Say $\mu = 0.05$, the penalty for A_1 is 0.2 and the penalty for A_2 is 0.15. The final score for each is thus respectively 0.62 and 0.64. A_2 is preferred with this value of μ since the user accepts to lose up to 0.05 in the average semantic similarity for removing one concept and improving concision. The question of the value of μ thus arises. Its value is nearly impossible to define by hand. In all of our applications, we optimized it by fitting the expected average result size of a test. It is important to note that this is so high-level (it reflects the desired annotation lengths) that a small test set is sufficient for this task. Also this could be combined with a fixed upper bound of size of returned annotations (algorithm 1 implements this feature).

2.4.2.4. Discussion of other alternatives and leads

Before going further, let us briefly describe the other alternatives that we have imagined and sometimes explored before abandoning them. Indeed,

so far we have detailed the search space and an objective function that relies on two criteria, but one can imagine different ways to calculate them.

2.4.2.4.1. An enriched search space

A_0 is defined as the union of concepts annotating the documents of the neighborhood. At first, we also imagined a definition that includes their ancestors. Given $A_0 = \bigcup_{A_d \in \mathfrak{A}_K} A_d$, this alternative search space A'_0 would be defined as $A'_0 = \bigcup_{c \in A_0} \text{anc}(c)$, where $\text{anc}(c)$ is the list of all ancestors (or superclasses, or hypernyms, etc.) of c. This alternative definition would allow the system to propose annotations including more abstract concepts and factorizing the annotation for example. However, factorizing the set of concepts annotating the neighbors is not the purpose of indexing and an annotation containing specific concepts is generally preferred. In the next chapter, we further detail an application where considering the ancestors has better odds to be relevant.

2.4.2.4.2. A specificity criterion

As we said an annotation had to be specific, one could suggest to emphasize the specificity of the concepts to find A^*. In other words, creating a criterion that would favor specific concepts over more generic ones. In fact, as we limit the scope of SSMs to the ones defined in the previous chapter (see §1.3.2.2), specificity is inherently favored. Indeed, we saw that if we consider two pairs of concepts distant by the same number of nodes in the graph, the pair with higher indivual IC—thus specificity—is considered to

be more similar. Besides, the choice of A_0 over A'_0 is another way to favor specific concepts.

2.4.2.4.3. Preprocessing the neighbor annotations

When observing the documents that are already annotated in a few corpora, particularly in the biomedical domain, we identified what we considered to be inconsistencies. The most common phenomenon is the presence of parent-child concepts, for instance a document annotated by {ANIMALS, MAMMALS, DOGS}. We thus thought about preprocessing such annotations in the neighborhood to make them less redundant and avoid noise in our algorithm. Such annotation would be replaced by {DOGS} here as it implicitly encompasses others. After some research we saw that human indexers have to obey some rules in order to keep the index consistent despite the number of human experts curating it. What we thought to be an inconsistency was in fact an indexing rule for the corpus. As a result, we decided not to put any constraint on the annotations of the neighbors because the essence of our algorithm is to mimic a human expert annotation.

2.4.3. ALGORITHM DETAILS

The algorithm is the final important choice to make. Finding an exact solution according to the objective function is not feasible because of the exponential amount of time required as there are $\mathcal{O}(2^{|\mathcal{A}_0|})$ possible annotations to consider. In fact, the problem seems to be NP-complete and close to the subset-sum problem for which a description is available in Cormen (2009),

Chapter 35.5. We thus chose to rely on a heuristic algorithm to keep the execution time low even if the size of the search space A_0 increases.

2.4.3.1. An intuitive heuristic

An intuitive heuristic that could come to mind to solve the problem of selecting the best subset of concepts among A_0 is a hill-climbing heuristic. Starting from an empty set, the algorithm adds concepts from A_0 as long as the objective function score increases. The opposite is also possible: removing concepts from A_0 following the same condition. Let us study the complexity of this algorithm by proceeding step by step. In the worst case, the optimum is A_0 and we added all concepts to A. Or for the opposite, the optimum is only one concept and we had to remove all concepts but one from A_0 to get to the solution. Let us define $n = |A_0|$, the algorithm updates the annotation n times and, if there is each time a single concept leading to an increase of $f(A)$ and it is tested after all others, then $f(A)$ is called $n + (n-1) + ... + 1$ times hence $\mathcal{O}(n^2)$.

The choice of an agglomerative or subtractive approach actually matters a lot. For example, let us consider the following A_0 ={MAMMAL, DOG , CARCINOMA} and a trivial neighborhood of two documents annotated by {MAMMAL, CARCINOMA} and {DOG, CARCINOMA}. We assume that the algorithm explores A_0 in its original order. With a subtractive approach, $A_0 \setminus$ {MAMMAL} would be evaluated and, depending on the value of μ, MAMMAL may be definitely removed from A_0. The other concepts would certainly be kept because their removal has poor chances of increasing the objective function value—again, it depends on μ. Now with an agglomerative strategy the algorithm would consist of iteratively adding concepts of A_0

to A, starting with $A = \{\}$. MAMMAL would be added first, then $A \cup \{\text{DOG}\}$ would be tested. At this point, the objective function may not increase, in which case $A^* = \{\text{MAMMAL}, \text{CARCINOMA}\}$. This means that by choosing an agglomerative strategy, we might pick a generic concept for an iteration that might prevent the selection of more specific ones afterwards. However, a subtractive strategy is not exempt from reaching a local optimum either since it is very sensitive to the order provided in A_0. We have yet not explored all the subsets of A_0 because of the use of a heuristic, so the possibility of reaching a local optimum should absolutely be minimized.

2.4.3.2. An improved heuristic

A good algorithmic alternative to the simple one-way exploration of A_0 is to find the best concept to remove at each iteration (see algorithm 1). This means evaluating all possibilities at each iteration. For each concept removal, all concepts of A—the current state of the set—have to be tested to find the best one to remove, that is: $n + n - 1 + ... + 1 = \frac{n(n+1)}{2}$. Let z be the size of A during the process, then $f(A)$ is called $\sum_{z=n}^{1} z$ times in any cases. The time needed to evaluate $f(A)$ for each concept at each iteration highly depends on the groupwise similarity that is used. Let us then study the time complexity of such an algorithm and explore some optimizations.

Algorithm 1: Approximate $A^* \subseteq A_0$ using a set of documents K

```
1  Function Annotate (K, μ, th, θ)
       Input  : The set of neighbors K, a real number μ ∈ [0;1], a maximal
                size of annotation th ∈ ℕ, an ontology θ
       Output : A set of concepts A
2      bestScore ← −∞;
3      A ← ⋃_{A_d ∈ 𝔄_K} A_d;
4      objectiveScore ← f(A, μ, 𝔄_K, θ);
5      while objectiveScore > bestScore or |A| > th do
6          bestScore ← objectiveScore;
7          maxTemp ← −∞;
8          maxConcept ← null;
9          foreach c ∈ A do
10             tempScore ← f(A\c, μ, 𝔄_K, θ);
11             if tempScore > maxTemp then
12                 maxTemp ← tempScore;
13                 maxConcept ← c
14             end
15         end
16         if maxTemp > bestScore or |A| > th then
17             A ← A\maxConcept;
18             objectiveScore ← maxTemp
19         end
20     end
21     return A
22 end
```

2.4.3.3. Study of complexity and optimization

Complexity of an unoptimized algorithm

In order to accurately assess the complexity of the whole algorithm, we must take into account the time needed to compute $f(A)$ defined in equation 2.12. It relies on the number k of selected neighbors and the computation of a groupwise semantic similarity.

In fact, we have to choose among three strategies for calculating the similarity of two sets of concepts that are (i) using a simple indirect groupwise measure, (ii) using a direct groupwise measure or (iii) using an elaborated indirect groupwise measure. The first strategy is for example to take the average of all parwise similarities of the two sets. The problem of such strategy is that it does not capture the similarity as well as direct groupwise measures. However, as explained in the first chapter, the disadvantage of a direct groupwise approach is its high computation time. The last option that we decided to choose is an elaborated indirect groupwise measure. It provides similarities that are closer to the direct groupwise measures. As for the direct measures, its computation is heavier than a simple groupwise measure but it allows algorithm optimizations because it uses pairwise similarities so that some computation can be saved when updating the annotation. The similarity measure we thus choose in order to evaluate the closeness between two groups of concepts is a composite average of pairwise similarities called Best Match Average (BMA) (Schlicker et al., 2006) defined as follows:

$$sim_{BMA}(A, A_d) = \frac{1}{2|A_d|} \sum_{c_d \in A_d} sim_m(c_d, A) + \frac{1}{2|A|} \sum_{c \in A} sim_m(c, A_d), \qquad (2.14)$$

where $sim_m(c, A_d) = \max_{c_d \in A_d}(sim_p(c, c_d))$. This function thus requires to compute every pairwise similarity between A and A_d. This means that for each evaluation of $f(A)$, all pairwise similarities of concepts from A and $\mathfrak{A}_K$ are considered. Let us assume that each pairwise similarity $sim_p(\cdot)$ is precomputed and accessed in constant time in this complexity expression. Indeed, it would be counter productive to actually repeatedly compute the same pairwise similarities. Instead, those similarities can be retrieved from a database that stores the semantic similarities of all pair of concepts for a given ontology. Say $S_{d_{max}}$ is the maximum size of annotations in $\mathfrak{A}_K$ and $z = |A|$. In the worst case, one groupwise similarity is thus computed in $\mathcal{O}(zS_{d_{max}})$ and the computation of such similarity is done k times, once per neighbor, according to the objective function. The computation of $f(\cdot)$ at line 10 (cf. algorithm 1) is thus done in $\mathcal{O}(kzS_{d_{max}})$. As it is done z times—once per iteration of the for each loop—, the complexity of the inner loop (l.9-14) is thus $\mathcal{O}(kz^2S_{d_{max}})$ and that of the overall algorithm (while loop) is:

$$\mathcal{O}(\sum_{z=n}^{1} kz^2S_{d_{max}}) = \mathcal{O}(kn^3S_{d_{max}}) \tag{2.15}$$

Optimizations

Computing BMA groupwise similarities is the most time-consuming task as it is done repeatedly.

Let us focus on how the BMA is calculated. Equation 2.14 shows that it requires to find, for each concept of A_d, the most similar concept in A and vice versa. This supposes the constitution of a matrix of pairwise similarities. The way A_0 is constructed implies that $\forall A_d \in \mathfrak{A}_K, A_d \subseteq A_0$. Therefore, building a matrix M_{ps} with all n^2 pairwise similarities of A_0 is the first step

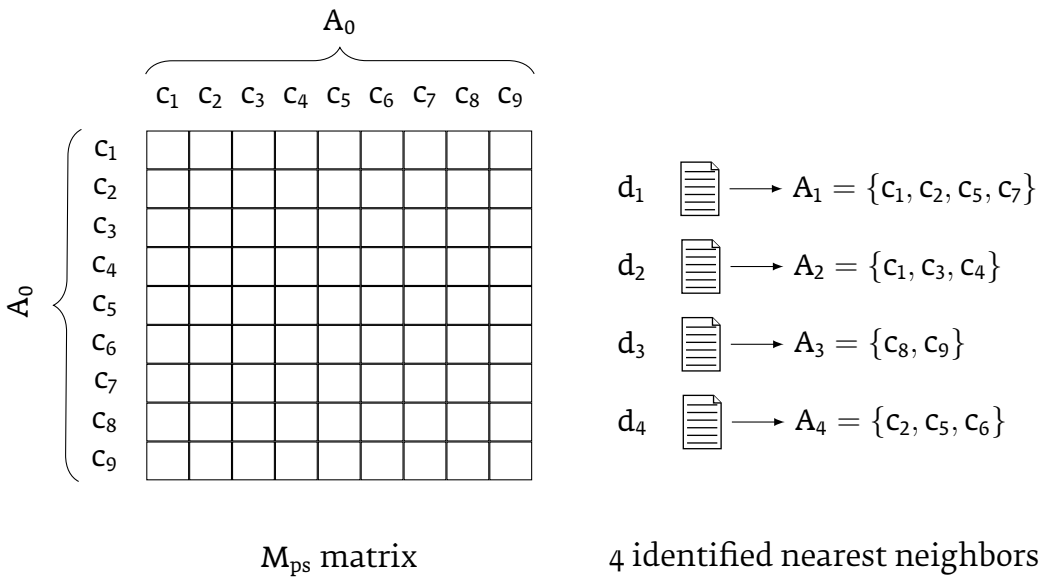


Figure 2.3.: Minimal example of the structure used prior to optimization.

to finding maxima and sum them up further on. Assuming that access to all pairwise similarities is done in constant time, initialization of M_{ps} is done in $\mathcal{O}(n^2)$ (see Figure 2.3).

Now, when calculating $sim_{BMA}(A, A_d)$, we must restrict the matrix to the submatrix $M_{ps}(A, A_d)$ to avoid browsing the whole M_{ps} matrix. This restriction is done in $\mathcal{O}(|A| + |A_d|)$ by simply identifying the indexes of the relevant rows/columns without duplicating the submatrix. In the rest of this section, M_{ps} is a shorthand notation referring to $M_{ps}(A, A_d)$. M_{ps} columns are concepts of A while its rows are concepts of a given document $d \in K$ as showed by Figure 2.4. The optimization of USI lies in the fact that when a concept c_r is removed, the score difference of $f(A \backslash c_r)$ can be efficiently derived from the data used to estimate $f(A)$ by updating the few intermediate values impacted by the removal of c_r.

The BMA measure relies on what we call SumMaxCols and SumMaxRows

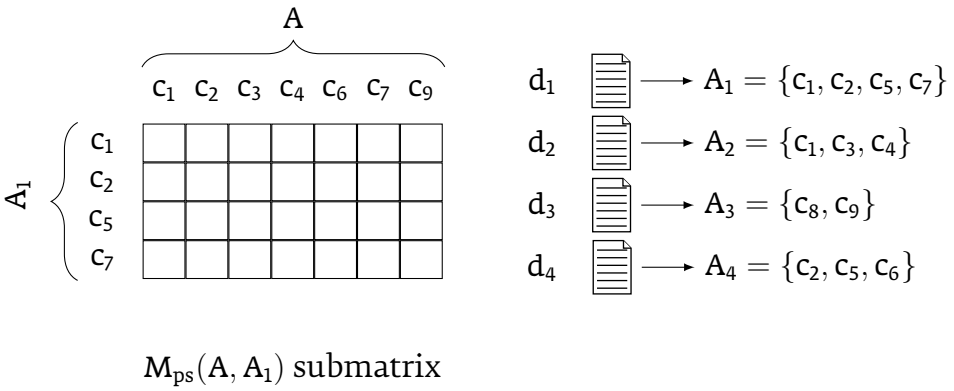


Figure 2.4.: Restriction of the M_{ps} matrix to handle d_1 when considering $A = \{c_1, c_2, c_3, c_4, c_6, c_7, c_9\}$.

functions. The average of the values of those functions gives the result of the BMA. In our case, when computing the similarity between A and the annotation of a single document A_d, $\text{SumMaxCols}(M_{ps}) = \sum_{c \in A} \text{sim}_m(c, A_d)$ and $\text{SumMaxRows}(M_{ps}) = \sum_{c \in A_d} \text{sim}_m(c, A)$. Let $\text{col}(c_r)$ denote the column of M_{ps} representing the concept $c_r \in A$ to be removed. When c_r is removed, SumMaxCols result can simply be updated by substracting the value of $\text{sim}_m(c_r, A_d)$. Therefore, we can compute once and for all k lists called MaxCol_d of maximum similarities for A_0—one for each document, see Figure 2.5—and substract the value of the maximum similarity that corresponds to c_r for a given document in constant time when needed. The computation of all the maxima is done for all concepts of A_0, for every document in K, in $\mathcal{O}(knS_{d_{max}})$ and can be updated in $\mathcal{O}(k)$ each time a concept is removed from A. When a concept is removed—that is, $\frac{n(n+1)}{2}$ times—the SumMaxCols values are updated so the overall management of the SumMaxCols values is in

$$\mathcal{O}(knS_{d_{max}} + \sum_{z=n}^{1} zk) = \mathcal{O}(knS_{d_{max}} + n^2k). \tag{2.16}$$

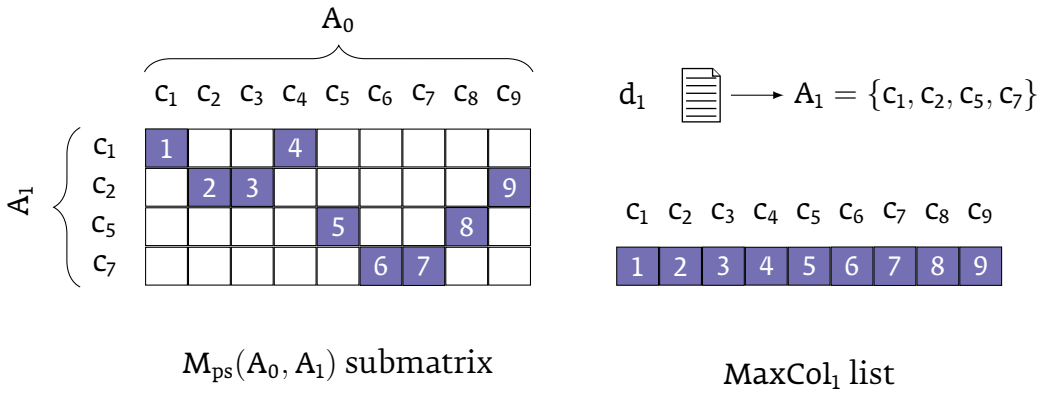


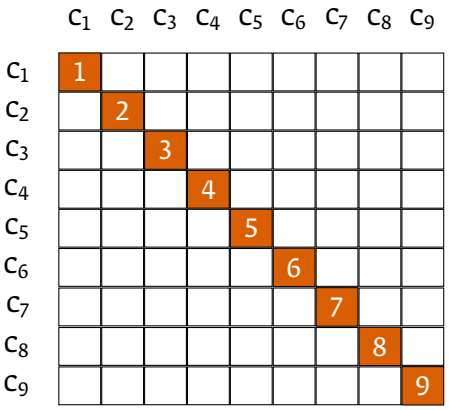
Figure 2.5.: Restricted matrix and MaxCol_1 list for d_1. Cells in purple in the restricted matrix contain the maximum similarity of each column, so for each concept of A. MaxCol_1 associates, for each column, the cell value that corresponds to the maximum similarity.

When testing a concept removal (l.10), SumMaxCols is stored before modification so that it can be restored in $\mathcal{O}(1)$. Updating the SumMaxRows value may require some more computation. The difference is that it may happen, when removing a concept c_r from A, that it was precisely this concept that gave the maximum value of a row and the new maximum must be calculated. The idea behind the optimization for this part is that we can detect when it happens. Figure 2.6 illustrates its implementation, detailed thereafter. For each concept $c \in A_0$, we store the list Inverse_c of documents in which it appears. We also store for each concept $c \in A$ the rows in which they currently give the maximum value in a list denoted Corr_c. As for SumMaxColumns, we compute the list MaxRow_d of all maximum values per row, that is, for each concept of $A_d \in \mathfrak{A}_K$ we find the concept in A_0 that gives the maximum similarity. This, and the creation of the Inverse_c and Corr_c, is done with the same complexity as SumMaxColumns, so $\mathcal{O}(\text{knS}_{d_{max}})$. After this initialization step, MaxRow_d and Corr_c may need

to be updated when A is modified, while Inverse_c never needs to. When a concept is removed from A, we know which rows to update thanks to the Corr_c list. Updating the SumMaxRows is easy as it only requires to find the new maximum for each concerned row, to calculate the difference between the old maximum value and the new one, and to add this difference to SumMaxRows to update it. In order to find the concept to remove at each iteration of the while loop (l.5-19), the maximum value for each of the n rows needs to be updated exactly once—since all concepts of A are tested once. Updating one is done in $\mathcal{O}(z)$ as the new maximum value needs to be found. The SumMaxRows value for a document $d \in K$ needs to be updated in $\mathcal{O}(1)$ each time a MaxRow_d value of one of its concepts is modified, i.e. at most $|A_d| < S_{d_{max}}$ times at each iteration of the while loop. This happens for all neighbors and for each iteration of the while loop, so in $\mathcal{O}(\sum_{z=n}^{1} kS_{d_{max}}) = \mathcal{O}(nkS_{d_{max}})$. Hence the management of SumMaxRows during the whole algorithm consists of the initialization of SumMaxRows, the update of MaxRow_d and the update of SumMaxRows and its complexity is respectively:

$$\mathcal{O}\left(knS_{d_{max}} + \sum_{z=n}^{1}(nz) + nkS_{d_{max}}\right) = \mathcal{O}\left(knS_{d_{max}} + n^3\right). \tag{2.17}$$

As for SumMaxCols, this complexity analysis only details removal of concepts. Indeed, restoration of a concept c that has been removed for testing A\c does not need computation. Indeed, USI caches all impacted values before modifying them when testing a concept removal so the complexity does not increase because of the restoration. It follows that the overall time complexity of this algorithm is defined by the complexities to create



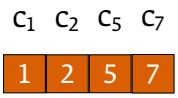
M_{ps} matrix

$Inverse_{c_1} = \{d_1, d_2\}$
$Inverse_{c_2} = \{d_1, d_4\}$
$Inverse_{c_3} = \{d_2\}$
$Inverse_{c_4} = \{d_2\}$
$Inverse_{c_5} = \{d_1, d_4\}$
$Inverse_{c_6} = \{d_4\}$
$Inverse_{c_7} = \{d_1\}$
$Inverse_{c_8} = \{d_3\}$
$Inverse_{c_9} = \{d_3\}$

$Inverse_c$ lists

$Corr_{c_1} = \{c_1\}$
$Corr_{c_2} = \{c_2\}$
$Corr_{c_3} = \{c_3\}$
$Corr_{c_4} = \{c_4\}$
$Corr_{c_5} = \{c_5\}$
$Corr_{c_6} = \{c_6\}$
$Corr_{c_7} = \{c_7\}$
$Corr_{c_8} = \{c_8\}$
$Corr_{c_9} = \{c_9\}$

$Corr_c$ lists



$MaxRow_1$ list

Figure 2.6.: Structures for optimizing the computation of SumMaxRow

and update M_{ps}, SumMaxColumns and SumMaxRows, that is

$$\mathcal{O}(knS_{d_{max}} + n^3), \tag{2.18}$$

which is more desirable than a straightforward algorithm in $\mathcal{O}(kn^3S_{d_{max}})$. Besides, the time complexity is independent from the size of the corpus[6] and that of the ontology, which guarantees the scalability of our approach.

We also studied the space complexity of this approach to make sure it is still scalable in space. USI is built on top of the Semantic Measures Library[7] (SML) (Harispe et al., 2014a), which loads the entire ontology when loading. Say the ontology is composed of $|\mathcal{C}|$ concepts. Semantic similarities we work on rely on the hierarchical relationships (see §1.3.2.2), so the space needed to load this ontology is in $\mathcal{O}(\mathcal{C}^2)$. The algorithm mainly stores three objects: M_{ps} in $\mathcal{O}(n^2)$, maximum values for rows and maximum values for columns both in $\mathcal{O}(n)$. Hence the overall space complexity of USI is

$$\mathcal{O}(n^2 + n + |\mathcal{C}|^2) = \mathcal{O}(|\mathcal{C}|^2), \tag{2.19}$$

because $A_0 \subseteq \mathcal{C}$, so $n = |A_0| \leq |\mathcal{C}|$. If all pairwise semantic similarities are precomputed and stored in a database, this space complexity becomes $\mathcal{O}(n^2)$. As USI is designed to be flexible and easy to use, it relies on a more appropriate compromise and lets the user only input an ontology instead of the list of semantic similarities. Therefore when USI has to compute a pairwise semantic similarity, it does it once and for all by caching its result. This is done in $|\mathcal{C}^2|$ (if all parwise similarities have to be calculated) and it

[6] In fact, the size of the corpus is important for the retrieval of neighboring documents but a lot of efforts have been devoted to the creation of scalable and efficient IRS to this end and PMRA is one of them.

[7] http://www.semantic-measures-library.org/sml/

Method	F-score	Semantic score	Running time (sec)
LTR	0.467	0.768	0.169
USI	0.521	0.776	0.003

Table 2.1.: Comparison of USI with LTR regarding the F-score, semantic score and processing time.

allows USI to benefit from quick calculation of semantic similarities in the algorithm—in fact, in real cases as fast as having all pairwise similarities in a database.

As we wanted to have an idea of this complexity in a real use case, we measured the processing time of USI on the same dataset as for LTR (Huang et al., 2011). The authors kindly provided us with the running time of their algorithm[8]. Table 2.1 shows the results of the comparison between USI as detailed above and the state-of-the-art system, in terms of F-score, semantic similarity and average processing time per document in seconds. Clearly, USI is faster than LTR (by a factor of 50) and the semantic score (Névéol et al., 2006) and F-score of USI are significantly better than those of LTR ($p < 10^{-6}$, although the semantic score is very close).

2.5. Including the user in the task

Annotating documents is an important task for upcoming processes such as information retrieval or decision making. Introducing a bias in the annotation would lead to inaccurate search results. Therefore, experts are often needed to validate the annotation. In this section, we describe when and how experts may intervene.

[8]Note that the running times or LTR are obtained by using a somewhat comparable configuration but on a different machine.

2.5.1. Prior to annotating

So far we have described the neighborhood selection as an automatic task, for example by using PMRA* to retrieve similar scientific papers. However, the constitution of the neighborhood is key in a k-NN approach. It is even more important for USI, since it solely relies on the neighborhood for genericity purposes whereas all other approaches also use features from the document itself. We thus imagined a neighborhood definition interface.

2.5.1.1. Interactive interface

We propose that the experts manually but easily restrain the neighborhood. Instead of automatically selecting the top k papers that PMRA* returns as the neighborhood for example, we take the first 100 papers it proposes. All of those papers are already annotated, so we can compute all similarities of pairs of documents by using the SML and one of the implemented groupwise semantic similarity measures. In order to be consistent with the indexing process that will follow, the semantic similarities used to build the similarity matrix are the same as in USI, i.e. Lin's pairwise measure with Seco's IC and Schlicker's BMA. The matrix created is then used by the MDSJ library[9] (Pich, 2009) to build a 2D semantic map. MDS stands for Multidimensional Scaling, an algorithm that makes a 2/3D projection of data so that the distance reflects on the map as much similarity as possible, i.e. semantically close items should be gathered on the projection while semantically non similar ones should be distant on the map. The output of this method is a set of coordinates that we use to build a map displayed

[9] http://www.inf.uni-konstanz.de/algo/software/mdsj/

to the user as shown on Figure 2.7. Each dot represents a paper associated with the name of its first author below it. Hovering over a paper displays a tooltip with the full name of the paper.

We then ask the expert to point the location where the paper to be annotated should be on this map. The expert should understand such a map and be able to accurately select a position for the document. Once the user clicked, the k closest documents on the map (using the Euclidean distance between the click location and the document coordinates on the map) define the neighborhood that is passed to the USI algorithm to determine the annotation of the new document.

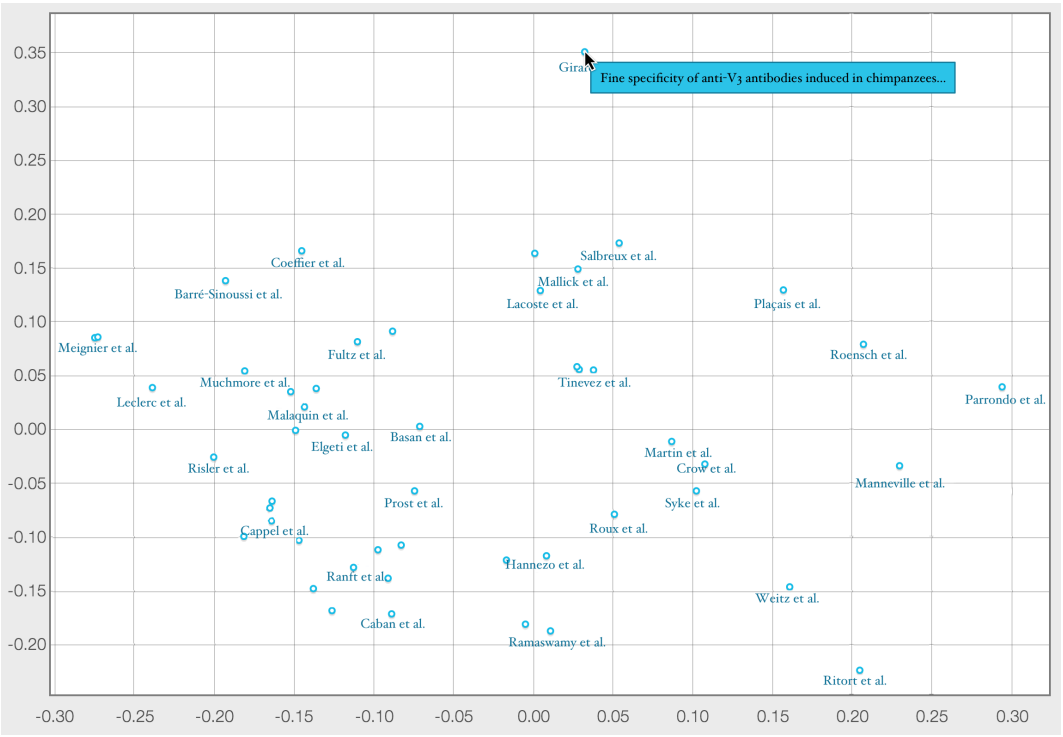


Figure 2.7.: Example of map displayed to the user.

2.5.1.2. Impact on the results

In order to test the benefits of such an approach, we evaluated the method by simulating clicks at the right position on the map. Indeed, evaluation datasets provide expected annotations, so for each document we can

- define a set of 100 neighbors using PMRA*,
- build the map with the document—by using its annotation—and its 100 neighbors,
- define a narrower neighborhood according to its location,
- run USI with this restricted neighborhood

Results of this analysis on the L1000 dataset are displayed in table 2.2. The conclusions are two fold. First, the F-score is better when using a fully automatic method. However, as explained in §2.2.3.2, it does not take the structure of the ontology into account and is thus useful as a relative score to compare two methods but not as an absolute score as it strongly penalizes small imprecisions (e.g. annotating MAMMAL where the gold standard is DOG gives an F-score of 0). The table shows how the F-measure can fail in accurately evaluating conceptual annotations. Second, scores according to the semantic measure are much more satisfying and certainly better represent the reality of the output. We observe that the results obtained by correctly clicking on the map are significantly better[10] than those of a fully automatic approach. This suggests that if the annotations are of great importance, e.g. concerning critical fields such as medicine, then relying on a human expert for defining the neighborhood might be a serious option.

[10] Significance has been tested with a paired t-test, for which $p < 10^{-6}$.

Method	F-score	Semantic score
USI automated	0.521	0.776
USI with map	0.509	0.807

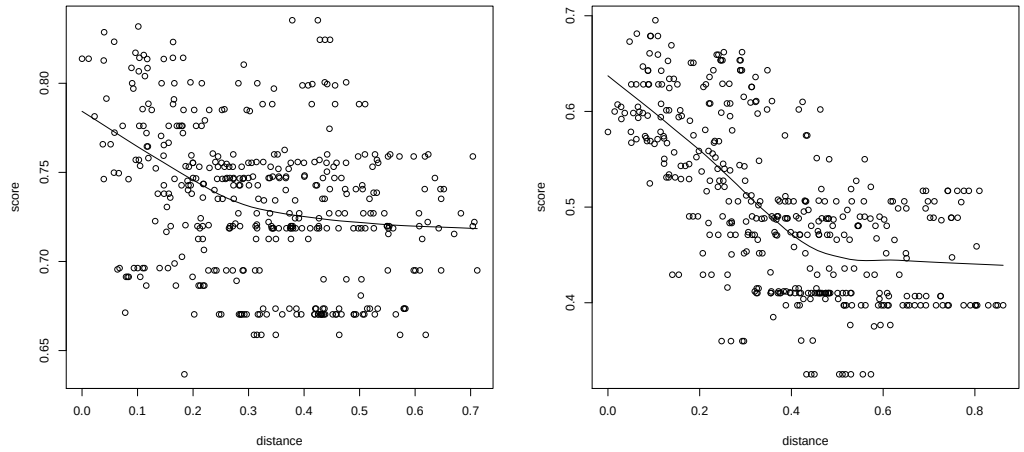
Table 2.2.: Scores obtained with and without the map. The semantic score is slightly better when relying on the map.

2.5.1.3. Limits of relying on an expert

Although this approach shows better annotations—according to the semantic score—, it suffers from several downsides. First of all, one may wonder how easy it is to point the location where the document should be. The expert is supposed to be able to do it quite easily since he/she would have a rich knowledge of the published papers in the domain, but we could not experiment on a real case to check whether or not this assumption is true.

It is certain that, whatever the time it takes for an expert to point the location on the map, it will be slower than a fully automated approach that outputs an annotation in a few milliseconds at most. One can thus question the need of an expert to manually define a neighborhood. USI provides both solutions and depending on the use case, more or less time should be spent on the definition of the neighborhood. We think that for difficult cases of sensitive applications it is critical to make sure the neighborhood is accurate, while in most cases the automatically generated one should be fine enough.

Finally, we studied the impact of imprecision when clicking on the map. Indeed, what if the user clicks 100px next to the correct location? 50px? 3px? In other words, it may be useful to know and inform the expert on the sensibility of the tool when displaying the map. There are in fact two



(a) Score variation in a homogeneous context.

(b) Score variation in a heterogeneous context.

Figure 2.8.: Semantic score variation in different contexts. Distance is computed according to MDS coordinates (usually in $[-0.5; 0.5]$).

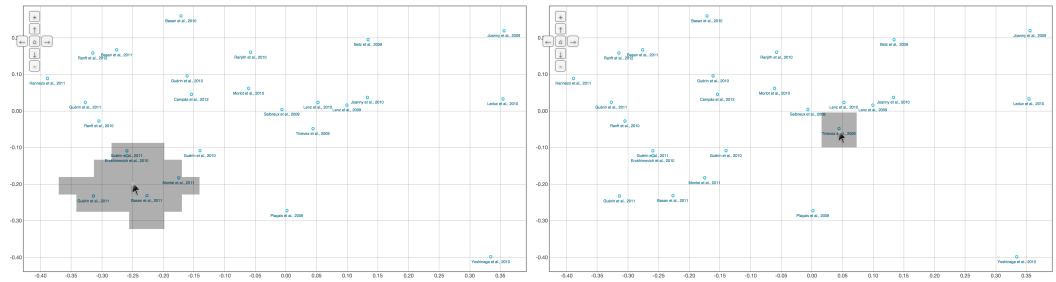
cases for which sensibility may be different. Sometimes maps (or zones of a map) contain documents with homogeneous annotations, while sometimes the documents are very different. For example, if a new topic has been barely studied, it is likely that few documents only would be similar, others would be returned anyway to reach the expected neighborhood size but they will have highly heterogeneous annotations. Figure 2.8 shows the variation in score of the output depending on how distant the click was from the correct location. This analysis is performed in two different contexts: a heterogeneous and homogeneous zone. It appears that depending on the context, the impact of imprecision of the click highly varies. In homogeneous zones, imprecision is well tolerated while in heterogeneous zones it leads to bad annotations.

Since the sensibility can vary depending on the map location, it is useful

to provide the user with information on the impact of click imprecision. Hence, we propose that when a map is computed, it is split in several tiles (their number depends on the expected granularity). A click is simulated at the center of each tile to define a tile neighborhood and a set of concepts that would be associated to a document placed in the center of this tile with the USI algorithm. For each tile, we compute the semantic similarity of its center with those of other tiles. We then associate, for each tile, the neighbor tiles which have a semantic similarity higher than 0.95 with this one. When a user hovers over any tile of the map, those that are similar are highlighted such that he/she easily knows how homogeneous the zone is and how careful he/she should be when clicking. Figure 2.9 shows an example of this process. When the grey zone is large (a), it means that click imprecision does not matter because the result will be roughly the same anywhere in this zone. Conversely, when the grey zone is small (b), the user should be careful when clicking since any imprecision would highly impact the suggested annotation. Of course, computing annotations of the center of the tiles is again time consuming, so this kind of use case would be much slower than a fully automated one. However, it would be faster for the user than an interface that does not help him/her and who would spend a lot of time thinking where exactly the click should be on cases when it does not matter due to annotation homogeneity.

2.5.2. After annotating

Despite efforts to automate the processes, experts are solicited for some applications such as annotating papers for PubMed. Although the annotation proposal is automated, it is sometimes crucial to rely on experts to



(a) The user is hovering an area with homogeneous annotations.

(b) The user is hovering an area with heterogeneous annotations.

Figure 2.9.: Representation of areas that let the user know how imprecision of a click would impact the results. On the left, the users do not have to worry about imprecision while on the right they need to be careful.

validate or to amend this proposal. This task is easier than a fully manual annotation as they only have to confirm the most suggested concepts and remove or add a few ones. Nonetheless, a rather inaccurate concept would rarely be replaced. For example, say a paper should be annotated by Carcinoma, Basal Cell. If the method returns Carcinoma, would the expert intuitively change it to pick the most accurate Carcinoma, Basal Cell? We expect from those experts to have an excellent knowledge of the thesaurus, but memorizing the whole structure and its 27,000 concepts is nearly impossible.

One solution to overcome this situation is to display the list of concepts and for each of them, to propose related concepts by using semantic similarities. The pairwise similarities can be stored in a database as explained in §2.4.3.3. Or, even more efficiently, the closest neighbors of each concept can be stored (e.g. the twenty closest neighbors of each concept according to the Lin SSM) and proposed to the user. In any case, as long as the user is part of the loop, some effort has to be invested to ease and speed up their

System name	Description
`USI 10 neighbors`	Default version of USI where 10 neighbors are selected
`USI 20 neighbors`	Default version of USI where 20 neighbors are selected
`USI abstract`	"`USI 10 neighbors`" where semantic similarity is chosen using an abstract framework
`USI baseline`	"`USI abstract`" integrating the provided baselines

Table 2.3.: Description of the systems submitted to BioASQ 2015.

work because it will always be the limiting factor in the process.

2.6. Evaluation of the approach: the BioASQ 3a task

We took part in the 3a task of the 2015 BioASQ challenge. This task consists of annotating biomedical papers given several inputs: the PMID—an identifier on PubMed—, the title and the abstract of each paper. Each participating team can submit results of up to 5 systems. We thus participated with the system described above and we created some variants presented in table 2.3 to investigate several questions regarding USI. The following sections detail several upgrades we made for those variants.

2.6.1. The optimal number of neighbors

In a previous study (Huang et al., 2011), the authors already estimated the optimal number of neighbors to consider for a k-NN approach. We conducted a similar study to see whether or not our application needs more, less or the same number of neighbors to give the best scores. While many approaches identify the neighbors to define a pool of candidate concepts and then rely on NLP strategies to score them, USI actually uses them to define the candidate concepts but also for selecting the relevant ones. Therefore, a too rich neighborhood may lead to some noise in our method and

less accurate annotations, while too few neighbors may not provide enough candidates.

It thus seems crucial to study the impact of the size of the neighborhood in our applications instead of solely relying on the previous analysis. We tested a neighborhood ranging from 5 to 40 documents. To do so, we ran USI 100 times on the BioASQ5000 dataset at each value of k and took the F-score and the average processing time. The results of this analysis are presented in Figure 2.10[11]. It shows that a plateau of F-score performance is reached at 10 neighbors. It stays stable up to 20 neighbours where the F-score starts decreasing. We explain this behaviour with the fact that considering too many neighbors induces noise. The processing time seems to increase linearly, which can be explained by the fact that the algorithm complexity is linear in k. However, we would expect that the size of A_0 (n, in the complexity details) leads to a non-linear increase. This can certainly be explained by the fact that since the neighbors are close to the document, they are certainly close in their annotations, so the size of A_0 does not increase much when adding similar documents in the neighborhood. The same study has been performed regarding the semantic similarity score. The resulting curve is flatter, showing once again that in this context, evaluating the results by using the underlying structure of the KR is more robust than the classical F-measure. Still, the score decreases passed 20 neighbors. We conclude that it is counter productive to take too many neighbors as it may decrease performances while increasing computation time (even linearly). For the BioASQ challenge, we thus test a USI variant (`USI 10 neighbors`) considering 10 neighbors as it seems to give the

[11] Those results have been obtained with an UNIX machine with a 3.4GHz microprocessor and 16GB of RAM.

best scores in terms of F-measure and processing time on the BioASQ5000 dataset. We also submitted a system considering 20 neighbors (`USI 20 neighbors`) since this gives the best results for the semantic score.

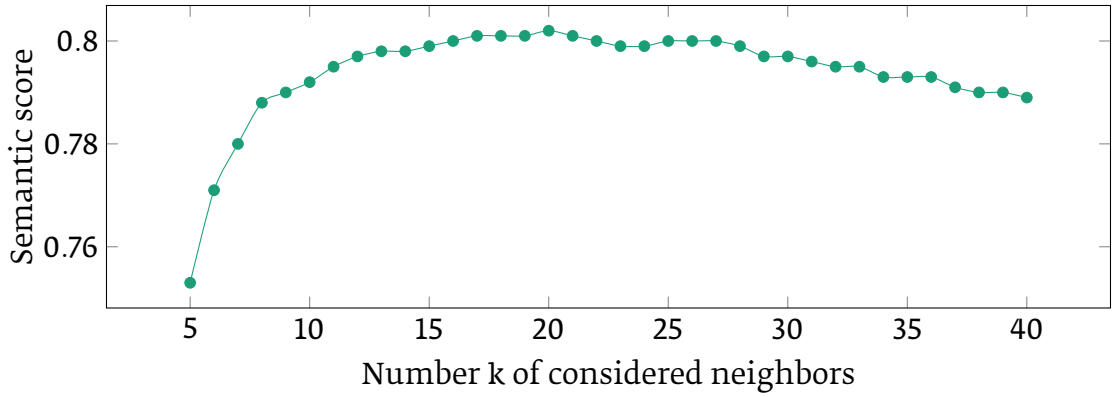
2.6.2. Questioning the system measures

One main question USI may raise concerns the impact of the semantic similarity measure on the performance of the system. Can we rely on the same semantic similarity measure irrespective of the application, or is it so important that an analysis should be performed prior to any real case indexing based on USI? Those questions may also bring up some engineering conclusions. For example, if semantic similarity matters substantially, then USI should be configurable to use as many measures as possible.

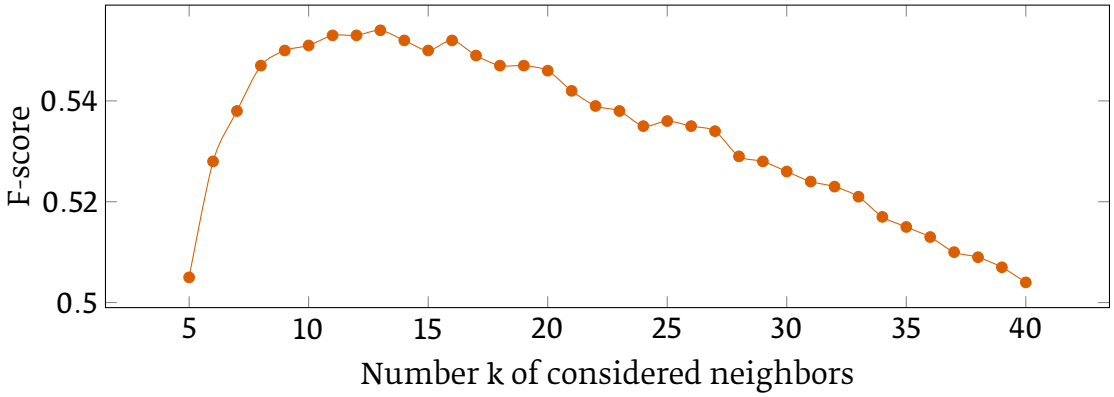
We used some provided test sets to make experiments on several semantic similarity measures. We chose to keep the aggregation formula (BMA) because it is quite a neutral composite average and there is nothing that seems to justify the use of more complex aggregating functions for this task. Therefore, the experiments have been made on several pairwise similarity measures and ICs, both of which can have an impact on the results. It has been recently showed that many measures proposed in the literature can be viewed as an instantiation of a handful of abstract models (Harispe et al., 2014b). For example, the Lin similarity is an instantiation of the ratio model sim_{RM} proposed by Tversky (1977):

$$sim_{RM}(a, b) = \frac{f(A \cap B)}{\alpha f(A \backslash B) + \beta f(B \backslash A) + f(A \cap B))}, \tag{2.20}$$

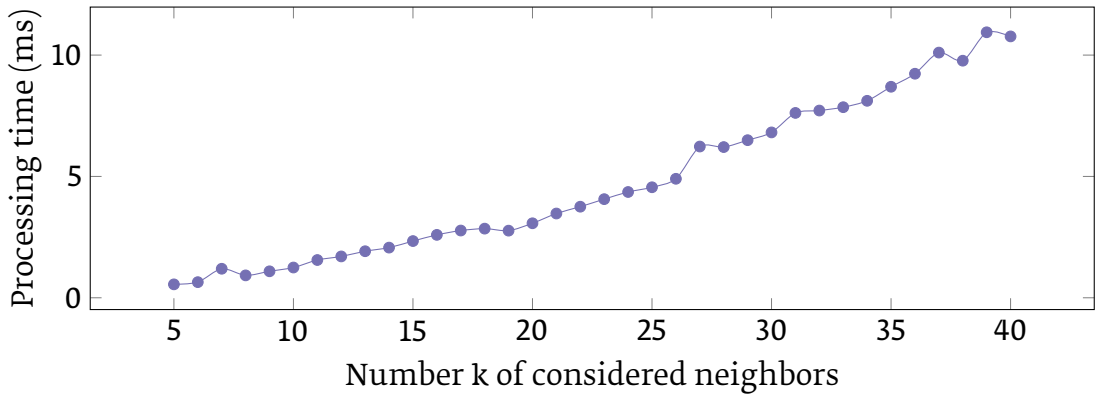
where a, b are concepts, A, B are their respective sets of features, f is a func-



(a) Semantic score as a function of k.



(b) F-score as a function of k.



(c) USI processing time (in ms) as a function of k.

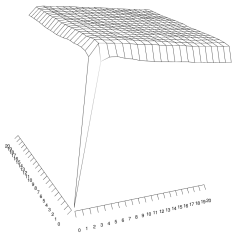
Figure 2.10.: The impact of changing the number k of neighbors between 5 and 40 on the the semantic score (a), on the F-score (b) and on the processing time (c) of USI.

tion defined on the sets of features and α, β are two parameters. For the Lin measure, $\alpha = 0.5$, $\beta = 0.5$, f is an IC function. It is easy to implement and test several variations of the ratio model by using the SML. We tested every variation of this model with a minimum value of 0 and a maximum value of 20 for α and β, with a step of 1. The choice of an IC may also have an impact on the results. We thus tested this model using 5 different ICs: Seco (Seco et al., 2004), Zhou (Zhou et al., 2008), Sanchez (Sánchez and Batet, 2011), Sanchez adapted [12] and a simple IC based on the number of ancestors. Figure 2.11 shows the impact of changing the parameters of the ratio model. Each subfigure represents the variation of α (on the x-axis) and β (on the y-axis) for a given IC metric and their impact on the final F-score of USI (on the z-axis) for the BioASQ5000 dataset. It seems there is not much variation in the annotations. The value at $\alpha = \beta = 0$ is an exception and always leads to a score of ~ 0.38. Indeed in this case, $sim_{RM}(a, b) = \frac{f(A \cap B)}{f(A \cap B)} = 1$, therefore all the concepts are similar to each other and USI randomly removes concepts from A_0, which leads to an inaccurate result. Otherwise, annotations are rated with an F-score that ranges in $[0.56; 0.60]$.

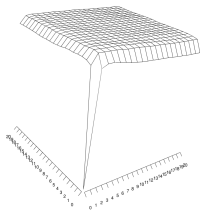
Table 2.4 completes the Figure by proposing, for each IC measure, the maximum and minimum F-score obtained—after removal of the $\alpha = \beta = 0$ value—and their corresponding α, β values.

Two interpretations can be drawn from the observation of the Figure and the table. In general, the choice of the SSM is not very important for USI. Someone who wants to implement USI should not spend a lot of time on this and simply make sure that the Lin SSM is appropriate. However, in the context of a challenge or a sensitive application, no increase of per-

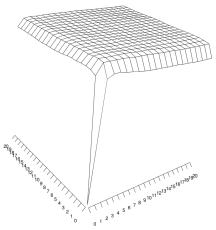
[12] This is from another formula in (Sánchez and Batet, 2011)



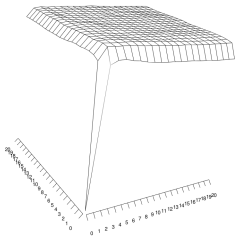
(a) Seco's IC (Seco et al., 2004).



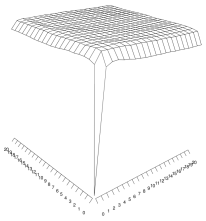
(b) Sanchez's IC (Sánchez and Batet, 2011).



(c) Zhou's IC (Zhou et al., 2008).



(d) Basic IC based on the number of ancestors.



(e) Adapted Sanchez's IC^{12}.

Figure 2.11.: Impact on the F-score for the BioASQ5000 dataset of changing the IC measure and the α, β parameters from 0 to 20 with a step of 1.

	Maximum			Minimum		
IC metric	F-score	α	β	F-score	α	β
Seco	0.5887	2	1	0.5627	0	20
Zhou	0.5903	2	2	0.5654	0	20
Sanchez	0.5872	3	4	0.5668	0	1
Sanchez adapted	0.5870	3	2	0.5626	0	20
Ancestors	0.5899	1	1	0.5650	0	20

Table 2.4.: Maximum and minimum F-score and their corresponding values of α, β, obtained for each IC. The value of $\alpha = \beta = 0$ is ignored.

formance should be neglected so we created a USI variant called `USI abstract` that relies on the triplet α, β, IC that provided the best score on the learning BioASQ5000 dataset: $\alpha = 2, \beta = 2$ and IC function is Zhou's (Zhou et al., 2008).

2.6.3. Including the baselines

In this BioASQ 2015 challenge, a slight variation of the F-score can play a drastic role as top systems often provide F-scores different by 0.01 or less. Therefore, we decided to adapt `USI abstract` to this challenge by taking baselines into account and we created the `USI baseline` variant. `MeSH Now` (Mao et al., 2014) is a baseline system this year and was the winner of last year's challenge. The aim when creating a new system is, then, at least to obtain better results than `MeSH Now` knowing that results of `MeSH Now` are available for the BioASQ5000 dataset.

First, we analyzed the differences between the baseline annotations and ours. The MeSH thesaurus contains headings called "*check tags*" [13]. There are 33 of them and they are widely used in annotations of papers, for example: HUMAN, ADULT, ANIMAL, etc. In our outputs, we looked for the concepts that are frequently wrongly predicted or frequently missing from our predictions and most of them were *check tags*. USI is actually pretty bad at predicting those tags. The fact that USI is a generic approach—it solely relies on the neighbors and not on the text features, for example—implies that it may lack some precision in some applications, particularly in comparison with very powerful approaches such as NLP. The problem of the *check tags* directly comes from this limit as it is difficult to predict the correct *check tags* when only the neighbors are used. For example, consider a paper

[13] http://www.nlm.nih.gov/bsd/indexing/training/CHK_010.htm

to be annotated for which the expert annotation is CARCINOMA and HUMAN. Assume for this example that closest neighbors are papers annotated by CARCINOMA, MOUSE, etc., rarely mentioning HUMAN. If the system uses no clue from the abstract or title text, it will barely be able to predict those tags. Since the baseline includes NLP tasks (with MetaMap) and seems to better predict the *check tags*, `USI baseline` systematically adds the check tags proposed by `MeSH Now`.

Second, we thought that the pool of candidate concepts may be too poor and increasing the number of neighbors would not help (see §2.6.1). `MeSH Now` outputs propose concepts coming from neighboring documents and concepts extracted from the text. Therefore, we enriched A_0 with the concepts proposed by `MeSH Now`. This modification led to the same F-score as `MeSH Now`—although the annotations were different. Finally, we defined a simple set of rules to combine MeSH Now and USI outputs. Let us define a set of concepts $A_{\text{processed}} = A^* \cup A_{\text{baseline}}$ where A^* is the ouput of `USI abstract` and A_{baseline} is the output of `MeSH Now`. Then, for each concept of this set, `USI baseline` keeps it if

- it is present in both A^* and A_{baseline},
- or it is present in A^* only but removing it from A would decrease the objective function score by more than $\varepsilon \in [0;1]$,
- or it is present in A_{baseline} only and it is in the $t \in \mathbb{N}$ top concepts of the list,
- or it is present in A_{baseline} and it is a *check tag*,

otherwise it is deleted. ε and t are optimized by using the BioASQ5000 dataset. This process only requires to store the value of $f(A)$ when look-

System	F-measure on BioASQ5000
MeSH Now BF	0.608
USI 10 neighbors	0.604
USI abstract	0.608
USI baseline	0.615

Table 2.5.: Summary of results obtained by USI systems on the BioASQ5000 dataset in comparison with the baseline, MeSH Now.

ing for the concept to remove. During the loop, USI tries to remove each concept and for each removal, it computes $f(A)$. This value is stored and updated for each concept of A each time USI tries to remove it.

2.6.4. Results of the challenge

Let us first consider the results obtained on the BioASQ5000 dataset by our variants presented in table 2.5. Interestingly, this table shows that USI abstract performs as well as the best system of last year's challenge, MeSH Now BF. While MeSH Now BF is designed for annotating biomedical papers, we see that a generic method can reach the quality of such a specific system, mainly because of the use of semantic similarities. By taking the MeSH Now BF baseline into account, USI baseline shows slightly better scores on BioASQ5000 than USI abstract. This proves that combining generic and specific approaches is a relevant perspective of research for designing new systems.

As for the challenge, since USI baseline systematically outperforms other variants, we simply refer to it as USI. We expected such result as a similar conclusion could be drawn from the table 2.5. The results of the challenge are split in three datasets of 5 batches each and the systems are eval-

uated once per dataset. The best batch-level results obtained by USI is on the batch 1 week 2, where it ranks second among 10 systems participating to this dataset[14]. In terms of dataset-level results, USI gets its best scores on the first dataset as it ranks 3rd. Balikas et al. (2015) summarize the participations to the tasks 3a and 3b of the challenge. They mention that since two baselines of MeSH Now are proposed this year and that it was the winner of last year's edition, "*[they] expected these baselines to be hard to beat*". Notably, on the first dataset, USI outperforms MeSH Now while on the other ones, `MeSH Now BF` has a slight advantage. These results are outstanding considering the fact that USI has not been originally designed for annotating biomedical papers. Neither the title nor the abstract of the papers to be annotated are used and still, USI gets among the top systems of the challenge. It ranks 4th out of 13 on the second and third datasets, for which `MeSH Now BF` is ahead of USI. Overall, even if not in the top two systems, USI is thus surprinsingly powerful compared to specific approaches. Besides, our participation showed that the method can handle large-scale indexing as a high number of documents had to be annotated for each batch and we managed to submit the results of all four variants of USI for each of them without parallelization, in less than 21 hours.

2.7. Extension of USI to several contexts

We made two applications in two different domains: enrichment of a scientific database with new papers and annotation of movies. Indeed, although this chapter focuses on the example of indexing biomedical papers,

[14]Participation to at least 4 batches out of 5 is required to be evaluated on a given dataset.

the scope of USI is not limited to it. The framework behind USI only assumes that a list of neighbors can be retrieved and that those neighbors are annotated by concepts from a structured knowledge representation. It then uses the neighbors to define a conceptual annotation for a new document. Validating USI on other domains is not as easy as we could not find benchmarks or reference datasets for evaluating automatic indexing of other media. Besides, we think it is important to verify that our method can be easily adapted to another domain with different KRs and different descriptions. Finally, these applications also allow anyone to try the interactive map as described in 2.5.1.

2.7.1. Enrichment of a scientific database: bioUSI

Demonstration at: http://bio.usi.nicolasfiorini.info/

This application illustrates the task detailed in this chapter: semantically annotating biomedical papers. The following sections detail the data used in this application and how it has been implemented.

2.7.1.1. Data

bioUSI relies on a database that has been elaborated by our team, consisting of relations between authors, their papers and conceptual annotation associated to those papers. It contains:

- 99,000 authors,
- 38,000 papers,

- 500,000 annotations, amounting to around 13 concepts per document.

This corpus focuses on the French medicine community dealing with cancer. It was created for the AVieSan (*Agence nationale pour les sciences de la Vie et de la Santé*) in order to help this community in its research. The database contains many other relationships such as teams of authors, research units and labs. It has been used several times to make adapted tools for this community such as OBIRS (Sy et al., 2012), a semantic IRS or CoLexIR (Ranwez et al., 2013), an IR tool that benefits from lexical analysis.

2.7.1.2. Interactive interface

When a user wants to annotate a document, he/she needs to fill a form specifying: (i) the title of the paper, (ii) the authors and (iii) the citations of this document. The author field benefits from an autocompletion tool. The system proposes names from the database that match the first typed letters in the field and the user can easily pick the correct ones (see Figure 2.12a). One can imagine that the form could be automatically filled after uploading a PDF file of the paper to be annotated or that a bibtex file would be parsed for instance.

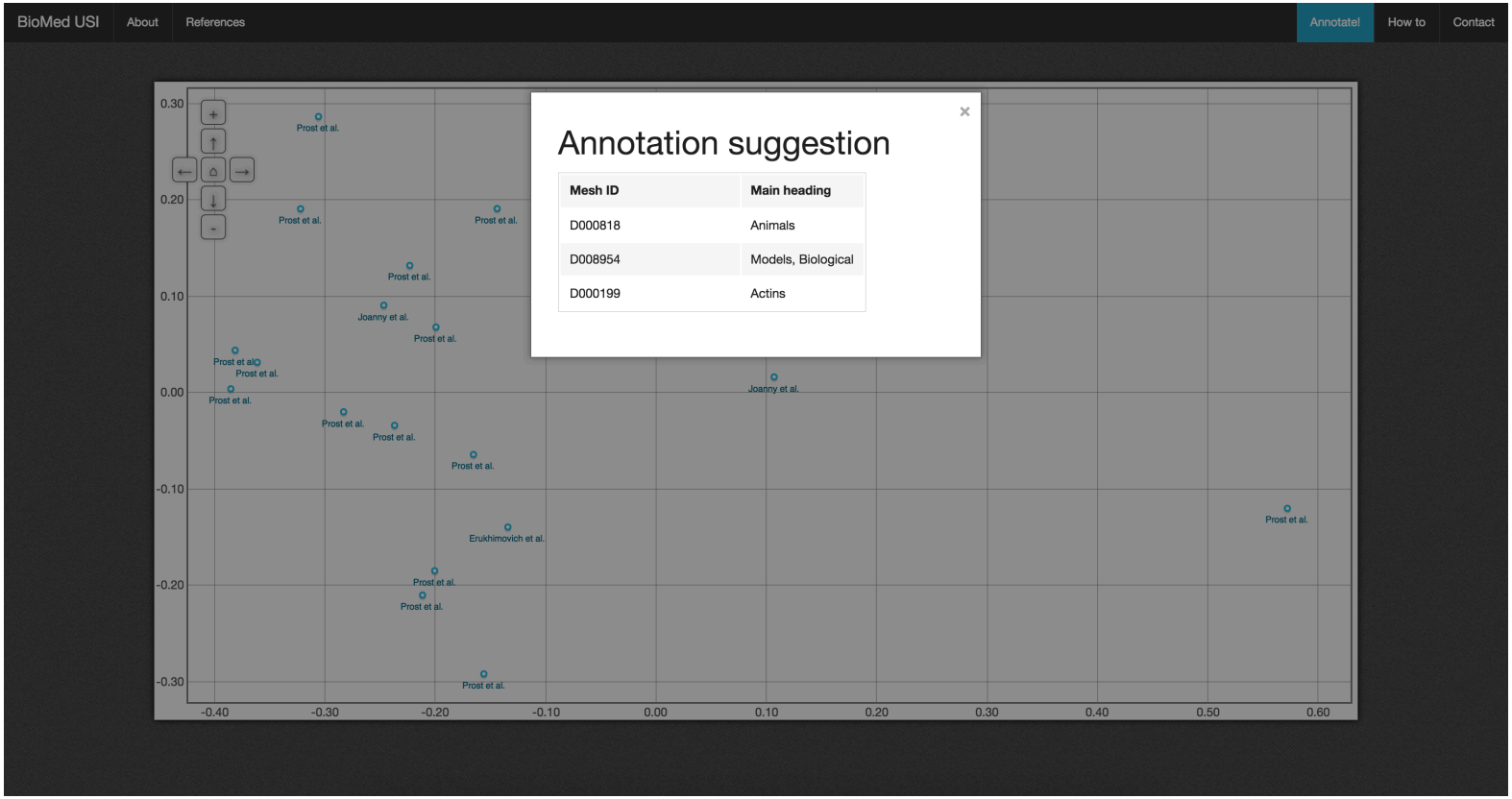
Once the form is sent, bioUSI needs to compute an MDS based on the information the user provided. To this end, bioUSI constitutes a set of papers selected on two criteria, instead of using PMRA*: (i) it selects papers including authors that co-authored the target document; (ii) it fetches the papers cited by the target document from the database or from PubMed depending on their availability. bioUSI only needs their title, list of authors, date of publication and the list of MeSH terms associated to them. This

process follows the assumption proposed by Delbecque and Zweigenbaum (2010) that previous papers of the co-authors and the papers they cite are certainly related to the one to be annotated. In this application, many documents are not indexed in PubMed as some of them are French papers or documents that are not published scientific articles but reports for example. An IRS such as PMRA* would thus not be relevant in this context, except for enriching the map with other documents. Once co-authored and cited document information is retrieved, bioUSI creates an interactive 2D map based on the MDS technique presented in §2.5.1.1 as showed in Figure 2.12b. The user can interact with the map by clicking on it once he/she knows where the document should be located. This triggers the selection of the ten closest neighbors according to the map and the launching of the USI algorithm with these neighbors as input. bioUSI then displays the annotation proposal of USI (see Figure 2.12c).

The server of bioUSI is a Tomcat 7 server that persistently stores the MeSH and provides two Web services. One is the MDS Web service that is called when the user sends the form. This Web service returns a list of coordinates associated with the data of each paper to be displayed on the map. The second one is an annotation Web service based on the USI algorithm.

(a) bioUSI form. Filling it is the first step of the annotation process.

(b) The MDS map presented to the user. Each item represents a paper.



(c) An annotation suggested by bioUSI after the user clicked on the map.

Figure 2.12.: The three steps in bioUSI illustrated.

2.7.2. Annotation of movies: moviesUSI

Demonstration at: http://movies.usi.nicolasfiorini.info/

moviesUSI is a more general application. The challenge here was to find data to annotate semantically that would make sense for anyone. moviesUSI also aims at testing the genericity of USI to see whether or not it is possible to load any knowledge representation and annotate any kind of document. As moviesUSI uses cutting-edge functionalities, it requires a recent browser to work properly (tests have been done on Chrome v43).

2.7.2.1. Data

Freebase[15] is a great database of linked data. For this application, we downloaded a dump of Freebase and extracted all the movies it contains. The set of movies constitutes the documents this application focuses on. We consider that each movie is annotated by several genres, but it is important to note that Freebase associates plenty of information to each movie that we decided not to use for this application. We chose to extract the movie genres as they are all structured. This structure of genres can be seen as a simple ontology: each genre is unique and subsumes/is subsumed by other genres. The database of moviesUSI thus contains:

- 238,000 movies
- 400 genres organized as a DAG
- 800,000 annotations, that is genre $\leftrightarrow$ movie associations

[15] https://www.freebase.com/

2.7.2.2. Interactive interface

This tool is a simple sandbox exploiting USI genericity and we suppose that users could use it to manage their video library containing movies that are in Freebase (hence already annotated) and some that are not.

We ask the user to build their collection of movies. An autocompletion tool helps here too to enter the movie names (see Figure 2.13a). Then, a map representing the collection is proposed to the person. The technologies and techniques used in this application are the same as in bioUSI—that is, Web services, MDS, etc. Note that we could use further details of the movie to annotate in order to build the map. For example, fetching the movies of the same actors, or of the same film-maker. The main difference with bioUSI is that the ontology is now a hierarchy of genres and the map is built solely using the movies the user has in his/her own local database (see Figure 2.13b). A click on the map will again generate an annotation for this location as in Figure 2.13c. We also tried to make the user interaction better by displaying a picture of the movie when available. Hovering over a movie displays a tooltip with its full name and year. It is also highlighted in the list on the left.

With this application, it is much easier to see the consistency of USI, irrespective of the domain ontology and the type of documents. The few users who tried this application gave us good feedback about it. They liked the whole interface and thought that the annotations suggested by the application made sense. The quickness of the whole process has also been appreciated, especially the rendering of the map and the computation of movie genres after a click on the map. Besides, the creation of such an

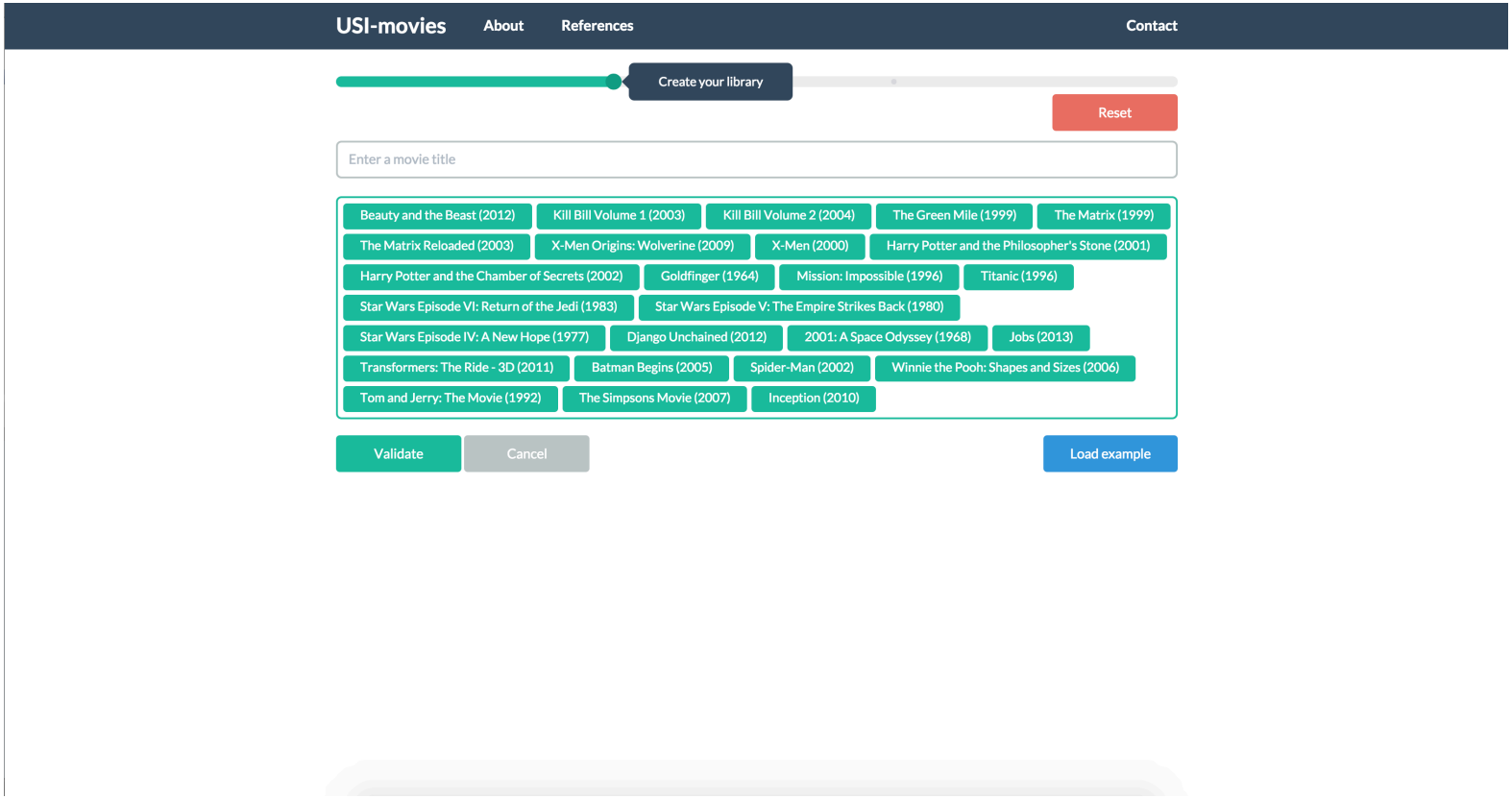
application is in tune with the objective of the LGI2P laboratory which promotes technological transfer towards industrial partners. Such technological transfer has been previously achieved with similar applications and moviesUSI is a serious candidate for creating or enhancing partnerships as it gives a great example of what USI can accomplish.

2.8. Chapter summary

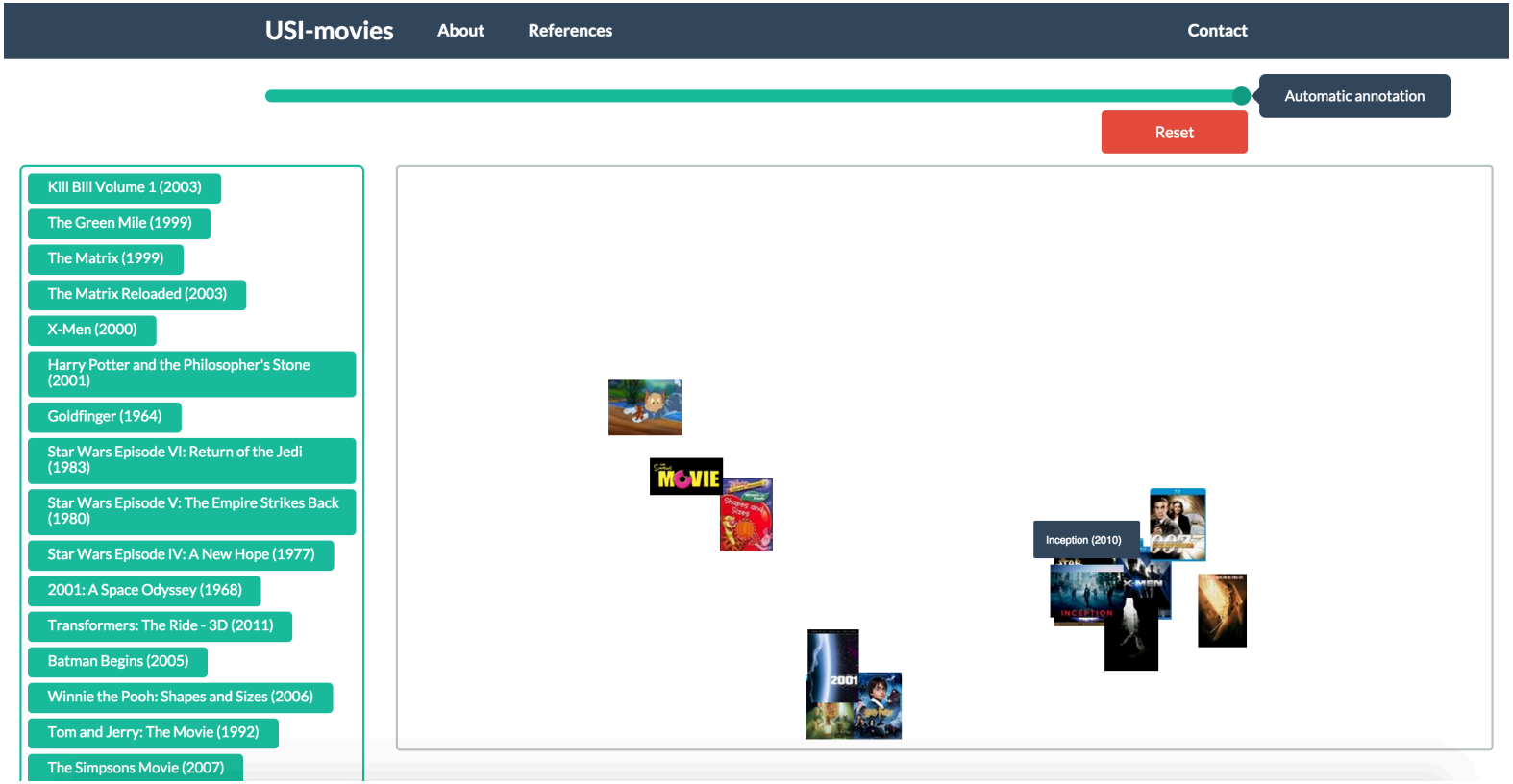
This chapter introduced the thorough work that led to the creation of USI—User-oriented Semantic Indexer. It was motivated by the need, in our opinion, for more generic indexing methods relying on ontologies. USI is an attempt to this end and explores the possibilities of exploiting semantic similarities to replace text-specific applications in the biomedical field.

USI is an indexing technique built upon an objective function modelled to define what is expected from an annotation, considering a neighborhood of semantically annotated documents. A heuristic algorithm implements this objective function to approximate the optimal annotation and a complexity analysis and optimization have been performed to make it faster. The result of this work is a blazing fast, flexible and accurate indexing method. It has been proved to be even faster than current ML approaches, knowing that they are already famous for their swiftness. Besides, while ML-based techniques require a (usually heavy) learning set, USI only has a few parameters that can be optimized on a small set of examples.

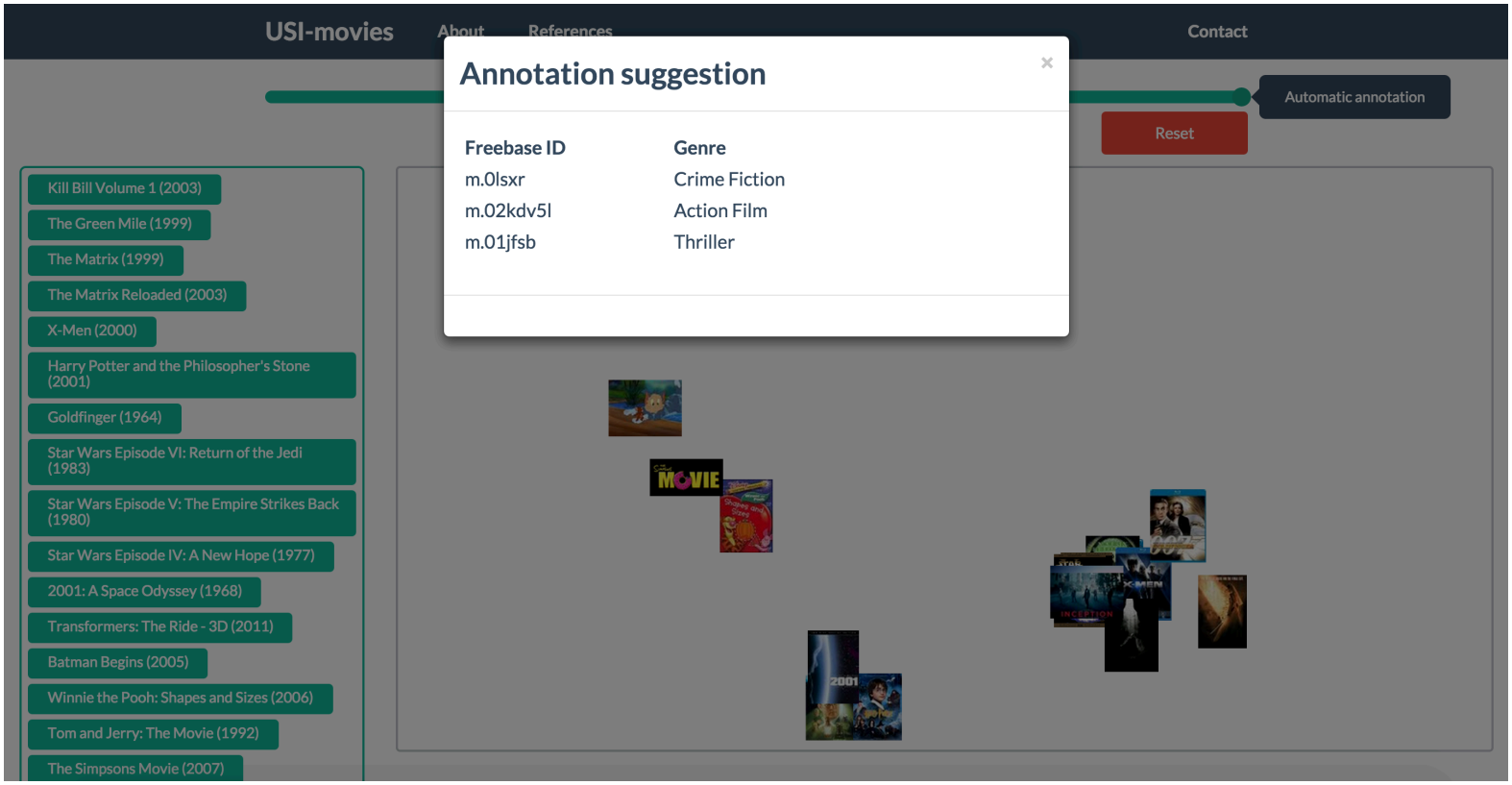
We also successfully put USI's flexibility to test with the BioASQ challenge by implementing four variants of the algorithm. The results are outstanding compared to our expectations: USI ranks in the top systems despite its



(a) moviesUSI form. Filling it is the first step of the annotation process.



(b) The MDS map presented to the user. Each item represents a movie, and a tooltip displays its title when hovered.



(c) An annotation suggested by moviesUSI after clicking on the bottom right part of the map.

Figure 2.13.: The three steps in moviesUSI illustrated.

genericity. This remark stresses that USI seems to be a powerful tool for semantic indexing in general as it performs as well as state-of-the-art approaches for textual documents. However, it is not *the* top system and this proves that text-specific applications still are the best choice approaches for text-document indexing. We note nevertheless that the scores of all systems are very close, which makes us wonder if a plateau has been reached in this field. This statement also questions the relevance of comparing systems when the scores are this close: does the difference mean anything at all? We will certainly have more insight into this question in the next edition of the BioASQ challenge in 2016.

Although USI can be run automatically, it is meant to be user-oriented. We thought that in many fields, the end-user would like to have some control over the annotations of the documents, so USI might be used as a proposal generator. We detailed two ways of including the user in the process. The most common one in the literature is to propose the annotation and let the user alter it. We investigated and proposed a map of potential neighbor documents to the user so that he/she may manually refine the neighborhood. The idea is that automatically defining the neighborhood may introduce a bias at the beginning of the process. We thought about the visualization of the potential neighbors as a map and described a visual solution to help the user when clicking on the map to limit imprecision. The creation of such interactive, ergonomic and flexible interface is in agreement with the LGI2P objectives of creating and proposing new industrial partnerships. Indeed, these tools are ready for technological transfer, as it has already been done many times in the team.

Finally, USI has been implemented in two applications, one related to bio-

medical papers, the other one to movies. Both of them show a good example of the potential of USI for annotating documents after meeting some requirements (such as the need for already annotated documents). Nonetheless, the framework from which USI originated—an objective function and an optimized algorithm—can be at the basis of many extensions. One extension, semantic indexing, has been deeply explored in this chapter. The next chapter shows that another feasible extension of this framework is semantic clustering and labeling.

3

❖

Semantic clustering and cluster labeling

Contents

3.1. Abstract

The last chapter introduced an indexing framework based on semantic similarities. This allowed us to create a generic approach for annotating any type of document. USI has been shown to perform well compared to state-of-the art approaches submitted to the BioASQ 2015 challenge. This chapter presents one of the numerous possible extensions of this framework. The use case presented here is a generic hierarchical clustering of documents with labeling of the clusters. We compare our novel approach to classical ones and study the benefits and limits of the use of semantic similarities in this context.

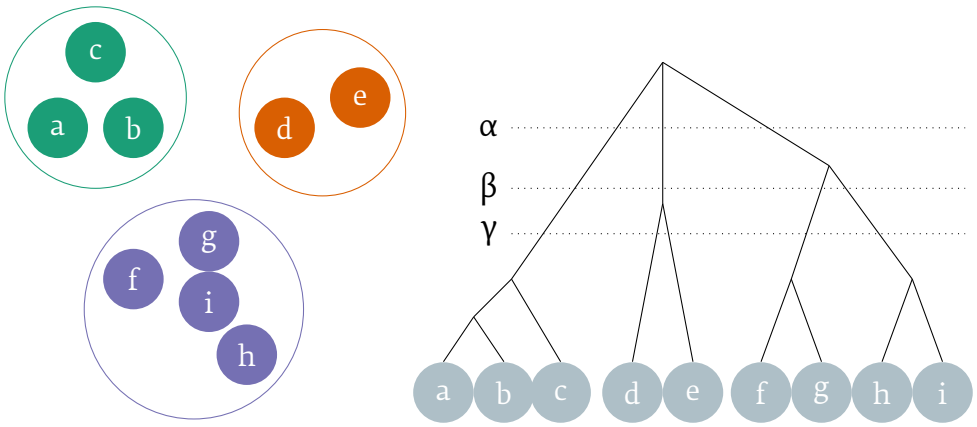
Contributions related to this chapter

Fiorini, N., Harispe, S., Ranwez, S., Montmain, J., & Ranwez, V. (2015). Annotation sémantique de clusters. In 16e conférence ROADEF, Marseille.

A semantic clustering interface: http://clustering.nicolasfiorini.info

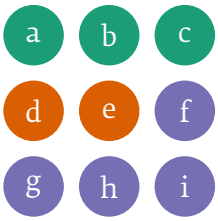
A semantic clustering benchmark: http://benchmark.nicolasfiorini.info

A semantic clustering approach, SC: http://sc.nicolasfiorini.info

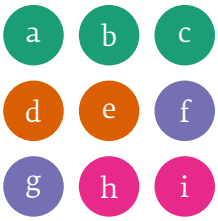


(a) Flat clustering (b) Hierarchical clustering

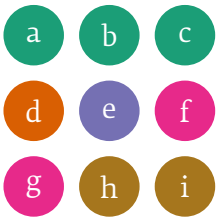
With the α-cut:



With the β-cut:



With the γ-cut:



(c) Clusters obtained with different cuts.

Figure 3.1.: Flat clustering (a) proposes a single partition of data while hierarchical clustering (b) allows to infer several possible partitions depending on the height at which the tree is cut (c).

3.2. General information on hierarchical clustering

As a general description of clustering has been proposed in §1.2.3, this section focuses on hierarchical clustering. The main difference with other approaches such as the k-means is that it provides a hierarchy of classes represented as a tree instead of a flat partition of the data. In other words, several partitions can be drawn from a single hierarchy (see Figure 3.1).

The hierarchical clustering, or equivalently the tree representing it can be obtained by following two strategies: an agglomerative or a divisive strategy. The construction is said to be bottom-up for an agglomerative strategy—the tree is built from the leaves to the root—and top-down for a divisive strategy. Given a list of n documents, top-down approaches require for each iteration to find the most distant pair of subsets among a list of 2^n subsets. Usually, top-down approaches use a flat clustering technique such as the k-means as a subroutine to keep a polynomial time complexity. The bottom-up construction is conceptually easier as it only needs to be able to find the closest clusters at each iteration. In total, we need to compare $\frac{n(n+1)}{2}$ pairs of clusters. For this reason, the bottom-up approach is more widespread and we decided to rely on it. A detail of the method is presented in Figure 3.2. There are two main steps in Hierarchical Agglomerative Clustering (HAC). One, the algorithm is initialized. Each document—more commonly called observations in the clustering community—to be clustered is put in a singleton cluster and a pairwise similarity matrix of all singletons is computed. Two, the closest clusters in the matrix are identified (a) and gathered. This means that a new cluster is created (b) and the matrix is updated by removing the two clusters and adding the new one (c). The similarities of the new cluster with every other have to be calculated as well (d). This second step is repeated until there is only one cluster, that is all clusters have been agglomerated. Note that branch lengths are arbitrary in the schema for the sake of understanding. However, branch lengths are important when partitioning from the tree as in Figure 3.1b and they are usually based on the similarity value of the agglomerated clusters in the matrix. The closer the clusters, the shorter the branches.

The key feature of HAC to define is clearly the way to compare (sets of) sin-

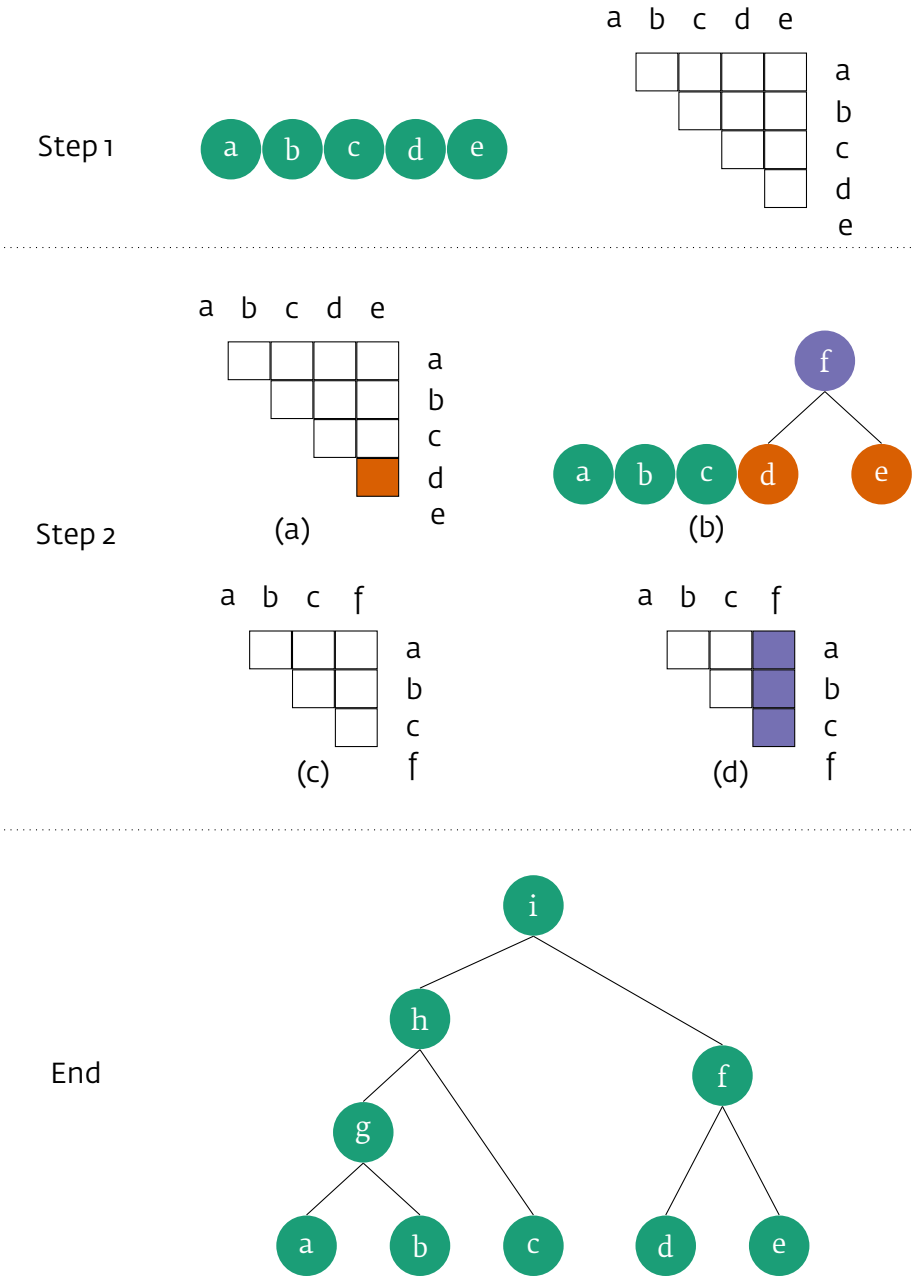


Figure 3.2.: The hierarchical agglomerative clustering. Step 1 initializes the clustering by creating cluster singletons and a pairwise similarity matrix. Step 2 consists in finding the closest clusters in the matrix (a), creating a new cluster f gathering them (b), updating the similarity matrix accordingly (c and d). Step 2 is repeated until only one cluster remains.

gletons that allows to build and update the similarity matrix. In fact, the HAC requires the definition of two functions called the similarity metric and the linkage criterion. The former is used to fill the initial matrix of similarities (step 1). Depending on the kind of data on which the clustering is made, several existing functions can be used for this purpose. For example, an Euclidian distance is appropriate for comparing coordinates and the Levenshtein distance for comparing strings. More elaborated distances can also be imagined, for example based on n-grams (all possible strings of size n of a text) for comparing texts, or on a related domain such as the IR relevance models.

When two clusters are agglomerated, the similarities of the newly created cluster with others are calculated by using the linkage criterion that defines how to compare two sets of observations. The choice of this function may impact the cluster shape and the branch lengths of the resulting tree. Most of the linkage criteria are function of the pairwise similarity metric that is used to compare the singletons. To cite the most common ones, there are the single linkage SLINK, the complete linkage CLINK or the average linkage ALINK

$$\text{SLINK}(A, B) = \max_{\substack{a \in A \\ b \in B}} \{s(a, b)\} \tag{3.1}$$

$$\text{CLINK}(A, B) = \min_{\substack{a \in A \\ b \in B}} \{s(a, b)\} \tag{3.2}$$

$$\text{ALINK}(A, B) = \frac{1}{|A||B|} \sum_{a \in A} \sum_{b \in B} s(a, b), \tag{3.3}$$

where A, B are the two clusters that are agglomerated, a, b are documents within A, B respectively and $s(a, b)$ is the similarity metric used at step 1.

Let us study the complexity of the HAC compared with that of the k-means. As Manning et al. (2008) point out, the time complexity of k-means is $\mathcal{O}(IKNM)$ where I is the maximum number of iterations, K is the number of clusters, N is the number of vectors (one vector represents a document) and M is the space dimensionality. This means that the k-means algorithm is linear in all variables, although the authors also note that M can be high depending on the vector representation, e.g. texts represented as vectors of most frequent words. The HAC on the other hand has a complexity of $\mathcal{O}(N^3 + N^2M)$ for a naïve algorithm and $\mathcal{O}(N^2M)$ for most real case algorithms. Note also that the complexity of HAC highly depends on the linkage function since this is the one that is frequently computed. The pairwise similarity of all documents is computed once and for all at the beginning of the process. Therefore the main benefit of relying on a hierarchical approach is that the output contains more information. The tree structure allows the users to study several granularities, which we explore in this chapter.

3.3. Related work

As described in §1.2.3, much effort has been devoted to the creation and improvement of clustering methods in general. In this chapter, we focus on clustering approaches that take into account semantic data, be it concepts or hierarchical structures in general. We also tackle the problem of annotating clusters, especially after or during hierarchical clustering.

3.3.1. Semantic clustering

Let us have an overview of the work we can find when looking for semantic clustering. It quickly appears that there are some methods called semantic or mixing clustering with ontologies or metadata, however, there is no proper consensus on a field called semantic clustering the same way we define it, which is clustering documents by using their semantic descriptions. Kuhn et al. (2007) for example introduced the concept of semantic clustering as the fact of grouping documents containing the same vocabulary. This approach is called semantic as they try to capture the meaning of the documents to cluster them. The aim of their work is to create a method to understand the source code of softwares in a matter of reducing the time spent reading the code for maintaining it. Although most approaches use the documentation or external data, they claim that semantics are contained in the formal part of source code, i.e. variable names, function names, etc. Their method involves latent semantic indexing (see §2.2.2.1) to compare pieces of code that can then be clustered. The clusters they create can also be automatically annotated by using the data of the latent semantic indexing, that is, identifying for each cluster the terms that mostly contributed to their creation.

Clerkin et al. (2001) propose to use clustering in order to discover and create ontologies—and not using ontologies to cluster documents. They use a conceptual clustering algorithm called COBWEB (Fisher, 1987) and translate the result in RDF (Resource Description Framework), a mean of representing Semantic Web objects. Conceptual clustering algorithm such as COBWEB aim at ordering observations in hierarchical classes with the particularity that each class (here, called concept) is described by a model

Name	BodyCover	HeartChamber	BodyTemp	Fertilization
Mammal	hair	four	regulated	internal
Bird	feathers	four	regulated	internal
Reptile	cornified-skin	imperfect-four	unregulated	internal
Amphibian	moist-skin	three	unregulated	external
Fish	scales	two	unregulated	external

Table 3.1.: A set of observations described by four properties (Fisher, 1987).

that summarizes the attribute-value distributions of objects classified in it. This assumes that, as for HAC, observations are associated with descriptions in the first place, such as provided in table 3.1. The goal and context of conceptual clustering is thus closer to classical clustering than what we want to achieve, although the vocabulary they use to describe the methods is quite similar.

Some other studies are nevertheless closer to the scope of our work. Some researchers have for instance studied the impact of integrating knowledge base information in clustering algorithms (Bharathi and Venkatesan, 2012). To the best of our knowledge, Hotho et al. (2001, 2002, 2003) have been the first to consider this kind of approach. Their series of works consist in enriching document annotations with background knowledge—in the most recent part, WordNet. Everything starts with the association of each document with a vector of term frequencies, further referred to as term vector. This vector is used in classical clustering approaches to compute document similarities. Then, they test all combinations of a few strategies for each step of the process:

First (`prune`) They prune the vectors with a threshold of `0`, `5` or `30`.

Second (`dis`) For each term in the vector space, they request the equivalently corresponding concepts in WordNet. They either fetch all cor-

responding concepts returned by WordNet (`all`); or they get only the first concept returned (`first`); or they disambiguate the term using an algorithm similar to that of Agirre and Rigau (1996) (`context`).

Third (`enrich`) They modify the term vectors by taking the mapped concepts into account. The `add` strategy simply concatenates the concepts with the term vectors. The `repl` strategy replaces the terms by their associated concept(s) from WordNet, if applicable, while the `rmv` strategy builds a vector of concepts only and removes all terms.

Fourth (`hyper`) In order to consider the inherent structure of WordNet, they include either $r = 0$ or $r = 5$ next hypernyms of the concepts that enrich the term vectors. They also update frequencies in the vector such that any occurrence of a concept counts as an occurence of its hypernyms.

Best purity of clusters is achieved with the following combination of strategies: `prune=30, dis=context, enrich=add, r=5`. Overall, the conclusion clearly is that relying on conceptual descriptions of documents improves the quality of the clusters. The authors explain that this improvement is due to the relationships between concepts (and thus the presence of common hypernyms, in their approach), where classical text clustering lacks such relationships. For example, they show that documents about COFFEE and CACAO were gathered in a FOOD cluster while food was never literally mentioned in those documents. (Baghel and Dhir, 2010) also propose an approach based on WordNet. In fact, they proceed very similarly to the previously detailed approach. They rely on concept frequency vectors that are built from texts, except that they use a hierarchical clustering approach. Breaux and Reed (2005); Sedding and Kazakov (2004) propose slightly anal-

ogous techniques relying on a more classical HAC algorithm and k-means, respectively.

Spanakis et al. (2011) prefer to use Wikipedia to test their novel Conceptual Hierarchical Clustering that they abbreviated CHC. They suggest a richer model for representing documents than weighted concept vectors. After having extracted concepts and retrieved their corresponding Wikipedia pages for each textual document, they build a feature set for this document including several distinct features. The weighted frequency *Wfreq* is the feature that is used in WordNet based approaches. They also compute *LinkRank* that measures the importance of concepts in documents by calculating the number of relationships each of them has with the others by observing links between corresponding Wikipedia pages. The *ConceptSim* feature calculates the term-based similarity (i.e., using classical term vectors) between a document and the Wikipedia page corresponding to one of the concepts that are extracted for this document. *OrderRank* is a value associated to each concept that is bigger when the concept appears early in the document. Finally, *Keyphraseness* captures the descriptive power of concepts in Wikipedia by relying on how the concept is referred to in the articles: plain text or a link to an article. The more a concept is used as a link in the articles, the more it has descriptive power. CHC is a hierarchical clustering technique that then relies on a combination of these features to build the tree.

The particularity of work by Yoo and Hu (2006) is that they propose to provide an understandable representation of the clusters that are created. Clustering is made in four main steps in their approach. First, as for most processes detailed so far, document terms are mapped with concepts of KR

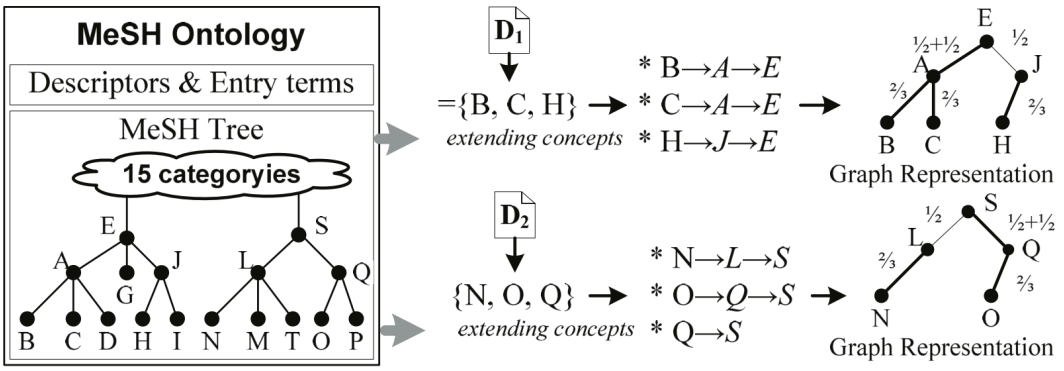


Figure 3.3.: Individual Graph Representations for each document (Yoo and Hu, 2006).

(here, the MeSH). A graph representing each document is built, that consists in the shortest path linking all concepts. In other words, the graph contains all concepts of a document and all their superclasses up to the superclass they share (see Figure 3.3). Second, the individual graphs are merged into a corpus-level graph. It is enriched with relationships of co-occurrences of concepts. An algorithm calculates the number of co-occurrences needed for a pair of concepts to be linked and for each relevant pair, an edge joining them is added to the graph. Third, they partition the corpus-level graph and define classes associated with a subgraph of the corpus-level graph. Finally, documents are associated with classes. The authors claim that graph similarity measures are not applicable in this context as the graphs are too different in terms of vertices and edges, instead a voting mechanism allows the system to associate each document representation—thus each document—to a cluster. In addition to proposing a novel way to use ontologies, this work is a step towards the idea of annotating clusters that will be discussed in §3.3.2. Indeed, the resulting clusters have a graph representation that can easily be understood by the users.

Song et al. (2009) are the first to use semantic similarities as a metric for

clustering semantically annotated documents. The SSM they rely on is that of Li et al. (2003) and is expressed by two factors:

$$\mathrm{sim}_{\mathrm{Li}}(c_1, c_2) = f_1(l) \cdot f_2(h), \tag{3.4}$$

where l is the shortest path length between c_1 and c_2 and h is the depth of their LCA. This reminds some other SSMs presented in the first chapter such as Lin's measure (Lin, 1998). This work features one of the ideas proposed by Hotho et al. (2003) of mapping the term vectors to all the concepts they may be an instance of. For example, if the word JAGUAR is used in the text, the strategy they follow is to associate it with the concepts of the cat, the car and the guitar. Then, the semantic similarity of two words w_1, w_2 represented by their concepts $(c_{1,1}, ..., c_{1,n})$ and $(c_{2,1}, ..., c_{1,m})$ respectively, is assessed as follows: $\mathrm{sim}_{\mathrm{Song}}(w_1, w_2) = \max\Big\{\mathrm{sim}_{\mathrm{Li}}(c_1, c_2)\Big\}$, $c_1 \in \{c_{1,1}, ..., c_{1,n}\}, c_2 \in \{c_{2,1}, ..., c_{2,m}\}$ and that of two documents is the average of similarities of their words. The clustering method they proposed is quite uncommon compared to all other papers as it implements a genetic algorithm. This kind of algorithm provides a near-optimal solution by randomly searching the space of solutions using principles analogous to natural selection and heredity. They argue the drawback of such an algorithm when applied to clustering is that it requires to set a number of clusters, and provide a way to remove this limit.

The use of knowledge for enhancing clustering has also been successfully tried for clustering non-text documents that have been semantically indexed, e.g. genes (Liu et al., 2004; Adryan and Schuh, 2004). Liu et al. (2004) try to tackle the curse of dimensionality[1] encountered in some clus-

[1] Coined by Bellman (2003), the phrase refers to the phenomena that appear when dealing

tering applications by proposing a novel subspace clustering technique. The idea of the task is to find potential clusters in various subspaces, which means that it can capture the fact that some items may belong to several clusters, depending on the considered dimension(s). The DNA microarray technology for example is known to be at the origin of high-dimensional data because it produces a lot of gene expression measurements at once. The authors focus on this use case by using the data of Spellman et al. (1998) that provide the expression of 6,218 genes of *S. cerevisiae* every 10 mins during 160 mins, which represents two cell cycles. They cluster the genes based on their expressions and show that guiding this clustering with the ontology by pruning the search space—so that irrelevant solutions are not explored—greatly reduces the computation time of the algorithm while producing results somewhat comparable in terms of cluster quality.

Finally, clustering has been applied to metadata in Maedche and Zacharias (2002); Lula and Paliwoda-Pękosz (2008). Metadata contain instances of ontology concepts that are also related to each other. Let us consider the Figure 3.4, *Finnland* is an instance of the concept COUNTRY and is related to *Finnish*, an instance of the concept LANGUAGE. Recall Definition 1 in Chapter 1, there are two kinds of relationships called taxonomic ($\mathcal{H}_C$) and non-taxonomic ($\mathcal{R}$). Metadata make use of all relationships to build a graph of instances for an underlying ontology.

They developed several semantic similarity measures based on the relations of the metadata graph to make a hierarchical clustering of the metadata. This approach is the closest to what we aim to do. However, they rely on a complex structure (metadata) that is rarely available because building

with high-dimensional data.

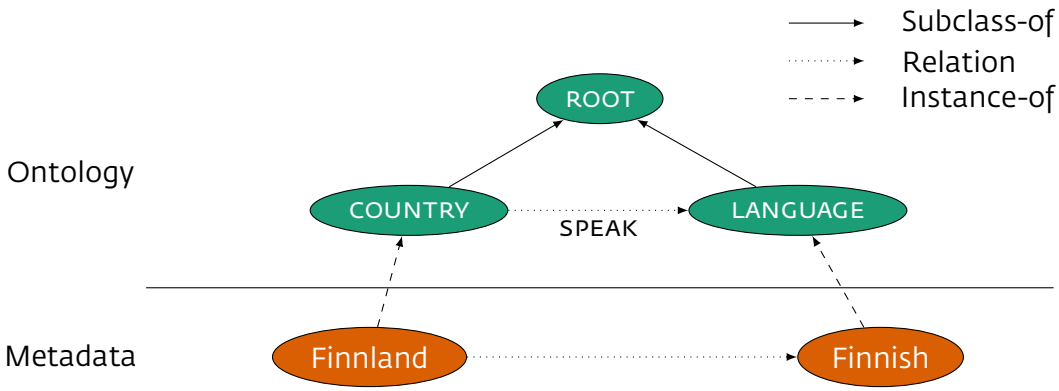


Figure 3.4.: Metadata is a graph of instances of concepts from an ontology. This Figure is inspired by Maedche and Zacharias (2002).

those databases is time-consuming. A still time-consuming but more frequent method is to annotate a document with concepts of an ontology instead of representing this document as a graph. For example, a dramatic movie occurring in France during the World War 2 would be a document annotated by DRAMA, FRANCE and WORLD WAR 2 instead of a graph with relationships ("when", "where", "genre") linking those annotations.

The papers described in this section are more or less related to what we call semantic clustering, ontology-based clustering or ontology-driven clustering. Although it is certain that relying on an ontology helps clustering documents, few works detail how to rely solely on the document annotations and underlying ontology. Besides, hierarchical clustering has been barely explored when considering ontologies e.g. Maedche and Staab (2001); Breaux and Reed (2005) compared to other clustering methods like k-means. Semantic similarities have been exploited in few papers (Maedche and Zacharias, 2002; Song et al., 2009; Shehata et al., 2006), but they did not consider the existing and numerous semantic similarities except in Ovaska et al. (2008), which focuses on genes. Unfortunately, this work has two

downsides. First, it is only applicable to genes, although the core idea is actually generic. It relies on GO (Gene Ontology) concepts—so it cannot load another ontology—to cluster the genes based on the semantic similarity of their annotations. Then, they map the clustering with the expression data as it is meant to be used for gene clustering after a microarray analysis. Second, they rely on a basic hierarchical clustering algorithm with classical linkage functions instead of exploring the behavior of recomputing semantic similarities during the clustering (see §3.2 for more details on this aspect).

As a result, there is no work in the literature that seeks to hierarchically cluster documents of any type by relying on their semantic annotations and semantic similarities.

3.3.2. Semantic cluster labeling

In analysis tasks and user interfaces, clusters have to be rapidly understood by the users. In Information Retrieval, for a query "travel to Germany", the results can be presented as clusters of hotels, restaurants, sightseeing, etc. Another common use of cluster labels is when we want to understand how items have been gathered (Manning et al., 2008). After clustering genes by using their expression data in a few environments for example, one may be interested in what characterizes each group. Consequently, cluster labeling often follows a clustering analysis as it is an important task for cluster analysis (Geraci et al., 2006).

Role and Nadif (2014) state that in most labeling approaches after text clustering, labels are simply terms picked from the texts according to their fre-

quency. The limitation of such a process is that relationships among the words are not represented. They thus propose an approach to make a graph representation of terms for the clusters and give the user a better understanding of the meaning of the clusters. First, they build a document-term matrix as for the LSA and they use a k-means algorithm for clustering this matrix into k clusters of documents. Second, they reduce the document-term matrix by observing the terms that most contribute in the creation of each cluster. Third, they build a term similarity matrix based on cosine similarity of weighted term vectors. Then, they combine the reduced matrix and the term similarity matrix to build a graph, with terms as vertices and similarities in place of edge weights. Finally, the graph is pruned by removing low-weight edges (and thus low-supported nodes) to keep it readable. The result is a graph with top elements—from the reduced matrix—connected to each other when they are highly similar. This representation of clusters is thus called semantic as it provides a structured summary of the content.

As for clustering, the benefit of using external resources such as Wikipedia has been predicted and tested (Carmel et al., 2009). Authors realized that even if the gold standard words were present in the documents of a given cluster, they would rarely be selected to annotate this cluster. In order to improve the results, they define for each cluster the list of best candidates by identifying those that distinguish the cluster from the others by relying on their previous work (Carmel et al., 2006). Instead of directly annotating the clusters with these terms, they use an IRS based on a Wikipedia dump and submit a query containing these terms. A list of Wikipedia articles is returned, from which they extract metadata, that is, the title and the categories of the article. Then, a scoring system evaluates the whole list

of candidates—the initial list of terms and the metadata from Wikipedia—with two metrics. One assesses the mutual information of each term with the other candidates[2]. The other one scores the label by averaging the score of the document(s) to which it is related and that are returned by the IRS. The results show a significant improvement compared to simple term extraction based on frequencies for instance.

Most of the work regarding semantic cluster labeling is related to gene clusters. The reason is that genes are annotated with Gene Ontology concepts that can thus be used for inferring labels to gene clusters. This way, some protein-protein interactions may be inferred, as an example. Besides, gene clustering is useful for researchers to understand which genes are expressed the same way in the same conditions, for instance for a given metabolic pathway. GOstat (Beissbarth and Speed, 2004) proposes to annotate a group of genes by finding overrepresented GO concepts among their annotations. This approach takes the structure into account as ancestors of concepts annotating the genes are considered to be potential labels. GOstat then performs a statistical analysis to pick the concepts that are overrepresented. Several other tools add slight variations to improve the performances and/or quality. While GOstat uses a combination of $\mathcal{X}^2$ with Fisher's Exact test, Lee et al. (2005); Bauer et al. (2008) use different statistical models, respectively the Mann–Whitney U test and the sole Fisher's Exact test. Another approach puts more emphasis on scalability and considers the order of the genes in the list (Reimand et al., 2007). The authors take the example of the constitution of a list by picking a gene of interest and adding other genes ordered by their similarity to the first one in terms

[2]The mutual information aims at measuring the mutual statistical dependence of two random variables.

of expression data. Finally, there is also some work for improving the user experience, for example with providing new covered species (Shah and Fedoroff, 2004) or new visualization tools (Maere et al., 2005; Paquette and Tokuyasu, 2010; Mi et al., 2010).

Although these approaches are relevant in a biological context, the way GO concepts are picked for labeling the clusters in another context can be discussed. Overrepresentation is one way of summarizing a set of concepts by picking those that directly or indirectly appear the most. This kind of approach, however, does not consider the specificity or genericity of the labels. Especially when labeling hierarchical clusters, the specificity/genericity ratio needs to be controlled in order not to give the same labels to successive parent nodes. Besides, the most elaborated approaches for cluster labeling are related to GO while this task is useful to many contexts with various ontologies. In essence, these methods could certainly be adapted to work with another ontology, however, the papers are often in application notes format that do not provide implementation details and the source code is rarely available.

Some papers introduced in §3.3.1 propose to build hierarchical clusters by using novel (Maedche and Zacharias, 2002) or existing semantic similarities (Ovaska et al., 2008). By doing so, they give the first insight of using semantic similarities for clustering and annotating the nodes. When a new node is defined in the tree, it is justified by a high semantic similarity of its children. Semantics thus literally explain the grouping and should be used in order to label the node.

3.4. Motivation & positioning

The previous section shows that although a lot a work has been done in the several aspects this chapter explores, there are currently few studies on clustering and cluster labeling from entities annotated with concepts. Yet this kind of task would be useful in numerous domains, for example semantically annotated images could be automatically ordered in (sub)folders that are named accordingly. While some effort has been devoted to clustering semantically annotated documents the methods lack genericity: inputs are genes with expression data, some others are metadata, there is no choice of the semantic similarity or they are restricted to one ontology.

The clustering field is a wide research topic and we do not want to create new clustering methods. However, we think that when the documents are annotated by concepts, clustering can be thought differently, or at least enriched as in Maedche and Zacharias (2002). To this aim, we conducted a study on the use of semantic annotations for clustering by using semantic similarity measures.

More importantly, we want to explore a new question: how to summarize a set of concepts? The structure linking the concepts altogether is known, so we may be able to find a way to reduce a set of concepts while keeping this set accurate. Some research intented to do so by statistically selecting overrepresented concepts in the set with consideration of their underlying structure, but overrepresentation might not be the best property to use for summarizing.

Such a process has a very concrete application, cluster labeling. Previous works emphasize the fact that cluster labeling is helpful, if not mandatory,

to understand the results of a clustering. When documents that are annotated with concepts are grouped into a cluster, one can think we can use those concepts to annotate the whole cluster, especially because the grouping is based upon their semantic similarity. Some concepts might be irrelevant for the whole group and should be removed while some others could be generalized—DOG and CAT as DOMESTIC ANIMALS for example, as suggested by some studies that consider hypernyms (Hotho et al., 2003). Two potential problems may arise when one intuitively tries to fulfill this task. Using all the hypernyms of each concept may lead to a lengthy description and solely using the documents' annotations would produce a too specific description of the clusters. Consequently, we want to propose a model that derives from our annotating framework.

We seek to make a method that can cluster semantically annotated documents while labeling those clusters instead of having two independent tasks. Indeed, clustering groups the items together for a reason that can be explained at the moment they are grouped, so this information should be used at the same time for labeling the clusters. Indeed, inconsistencies between the clusters and their annotations may be encountered when the two task are done independently, because the clusters would be obtained according to different paradigms to those of the annotations.

3.5. Benefits of semantic clustering

This section focuses on our contribution to the clustering of documents by using their semantic annotations. As for automatic annotation, the aim here is to only use those annotations so that the final application is generic.

3.5.1. A CONSISTENT AND ACCURATE CLUSTERING APPROACH

As explained in §3.2, the similarity metric is chosen according to the type of data to cluster, e.g. the Levenshtein distance is often used for clustering strings whereas n-grams approaches are favored for texts. It thus seems intuitive to rely on a semantic similarity measure as the elementary metric to build the similarity matrix of documents annotated by concepts. We assume that each document is annotated by one or several concepts. The similarities of all documents needed for creating the initial matrix are computed by using a groupwise semantic similarity (Harispe et al., 2014a). The first feature of our approach is that, when two clusters are agglomerated into a new cluster, we recompute its similarity with every other cluster by following the same similarity we have used for creating the matrix. While classical methods would use an arithmetic function based on elementary similarities such as an average, it seems to be more accurate to rely on the same similarity function. In fact, this proposal guarantees that the whole tree is consistent w.r.t a given SSM, and that there is no bias introduced by averaging (or maximizing, or minimizing, etc.). This idea has already been successfully implemented by Ranwez and Gascuel (2002) and it shows that at a small computational cost, the quality of the clustering can be improved. This feature yields the second novelty in our algorithm in the sense that we propose the clusters as well as the labels that can describe them so that they are easily interpretable. That is, when a cluster is created, it can be annotated by using the concepts of the observations it contains and the resulting labeling is at the basis of the similarity of the cluster with the other ones (see Figure 3.5). Note that again, as our implementation uses the SML, it provides a great flexibility on the SSMs that can be used and a

similar study could easily be conducted with other metrics picked among the numerous ones proposed by the SML. In other words, the hierarchy of clusters and their annotations are all consistent w.r.t. a given SSM, which also implies that the quality of the labels and that of the clustering are mutually important and strongly related.

Unfortunately, the studies presented in the related work section do not provide a satisfying semantic similarity-based technique for clustering semantically annotated documents. They either describe a new way of computing semantic similarities based on another kind of data (Maedche and Zacharias, 2002) or they require a too specific input (Ovaska et al., 2008). In our adaptation, we decide to pursue with the BMA that we already used for annotating (see Chapter 2) as it—or a close variant—has already been used in previous studies for this task as well (Ovaska et al., 2008; Song et al., 2009). As we do not have any pre-requisite about the behaviour of the clustering method, a composite average seems to be appropriate. Depending on the needs of the application, the choice of the pairwise similarity can vary. For our tests, we use the Lin similarity (Lin, 1998) with Seco's IC (Seco et al., 2004) that are both common in the literature and quite neutral, i.e. they do not aim to fulfill any particular requirement (see §1.3.2.4). Besides, this combination has been proved to be relevant regarding the results obtained in the previous chapter.

3.5.2. Labeling the clusters

We still need to define the set of concepts that is used for representing/annotating a given cluster. As for annotating a document, this requires the definition of a selection strategy based on a model of the objective. This section presents the choices we made and the justification for them.

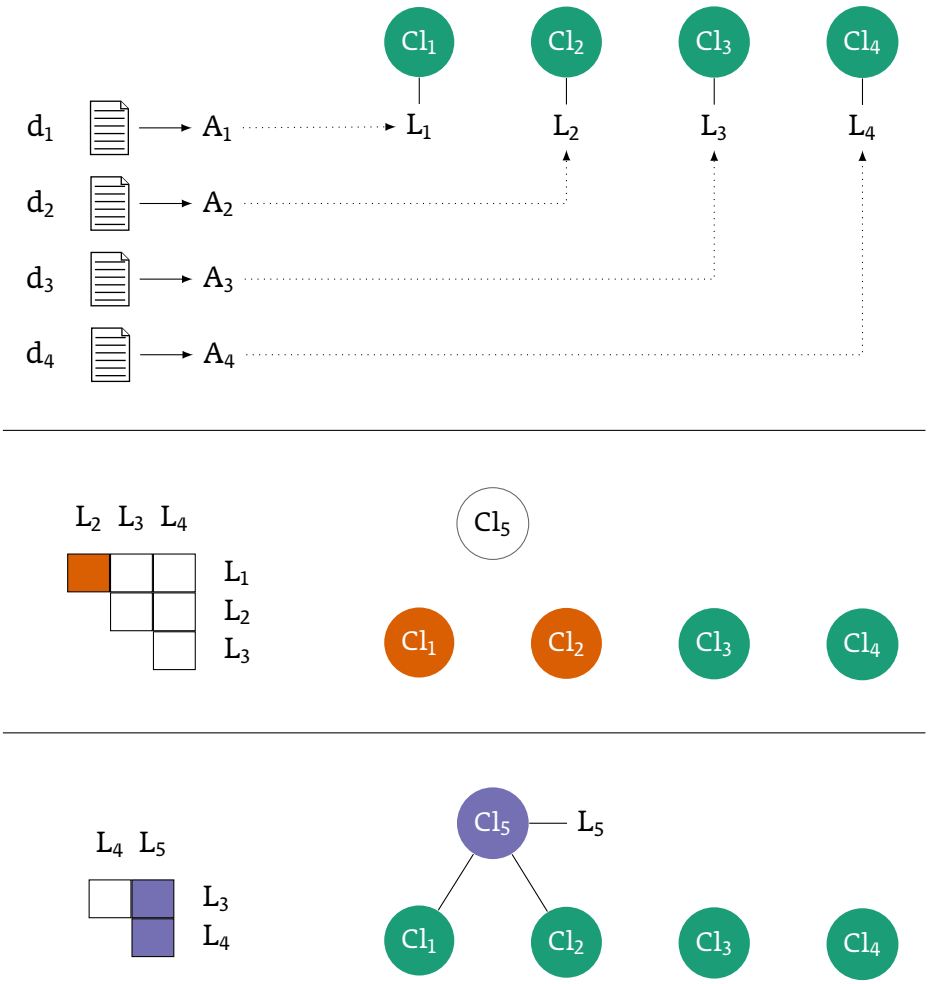


Figure 3.5.: HAC adapted to semantically annotated documents. First all singleton clusters are associated with the annotations of the documents they represent, called their labels. Then a similarity matrix is computed upon the semantic similarities of labels of clusters. It is used to find the closest clusters that (here Cl_1, Cl_2), when agglomerated, lead to the creation of a new cluster Cl_5 which label L_5 needs to be calculated. Once this is done, Cl_5 is added in the matrix and corresponding semantic similarities are calculated.

3.5.2.1. Summarizing versus merging

Let us first expose an example by denoting Cl_1, Cl_2 two singleton clusters (representing one document each), respectively $Cl_1 = \{d_1\}, Cl_2 = \{d_2\}$. Those documents are annotated by A_1 and A_2, two sets of concepts. This is a very early step of the clustering since each cluster represents only one document. Therefore, the labels of Cl_1, Cl_2 simply are $L_1 = A_1, L_2 = A_2$, respectively. Say Cl is a new cluster such that $Cl = \{Cl_1, Cl_2\}$. We need to define the labels L_{Cl} that characterize Cl.

One very intuitive solution is to merge the labels of Cl_1 and Cl_2. That is, $L_{Cl} = L_1 \cup L_2$. As briefly introduced in the Motivation section of this chapter, this solution may be problematic both for the labeling and for the resulting clustering. When merging the labels, the number of concepts will increase quite fast while clusters are agglomerated. As a result, the top level clusters—the ones close to the root of the resulting tree—may be hardly interpretable for the user. Some concepts are very specific to one document and thus should not be propagated to clusters of higher levels. This can also affect the clustering by adding up some noise because as stated in the previous section, the quality of labels may impact that of the clusters and vice versa. Besides, such a strategy ignores the weights of concepts in the agglomerated clusters. That is, if one concept is more represented than the others, it would be wrongly considered as the others. A more appropriate process would be that as the clustering is agglomerating, the labels get more and more generic because the clusters themselves are getting more abstract. Finally, in a basic merging approach, the commonality of labels of the grouped clusters is not taken into account while it should be regarding the fact that labels are concepts from an ontology. Redundancy due to

the commonality may thus be avoided by leveraging the structure of the underlying ontology.

A promising alternative is to summarize the labels. Every time a cluster is created, its labels should be computed according to the two clusters it contains. There are some objectives that can be expected for such an interpretable summary:

- it has to accurately reflect the content of the cluster, i.e. reflect the commonality of the documents belonging to this cluster;
- it must be as specific as possible while proposing some more abstract concepts sometimes;
- it must contain a limited number of concepts.

3.5.2.2. Modeling the objectives

Those objectives remind of the ones we defined for annotating documents. The concision objective has the same goal, that is being easily understandable by a human. Consistency is obvious as the labels should accurately represent the clusters. The main difference regards the specificity of the labels. When annotating documents, we aim at selecting the most specific concepts. Here with clusters, we want to be able to generalize sometimes. For example, if a cluster contains documents related to CAT, DOG, RABBIT and HUMAN, it would be accurate and easier for the user to label it MAMMALS. The loss of information must be minimized during this process so that the generalization is controlled: ANIMAL would be too generic.

The process for labeling a cluster is designed the same way as for annotat-

ing a document. That is, we define a search space L_0, a neighborhood—here, the neighbors are the clusters that have been grouped—and we look for an optimum L^* that maximizes an objective function $g(L)$

$$L^* = \operatorname{argmax} \{g(L)\}. \tag{3.5}$$

The same greedy algorithm as the one presented in Algorithm 1 can be applied as it relies on the same type of input. The same process, i.e. iteratively removing concepts from L_0 seems legitimate too. Only the calculus of the value of the objective function $g(L)$ is different.

Let us denote Cl the cluster to annotate containing two clusters, i.e. $Cl = \{Cl_1, Cl_2\}$. The clusters Cl_1, Cl_2 are already labeled—with L_1, L_2 respectively—since we chose a bottom-up algorithm. We define L_t as a temporary set of concepts containing all concepts in L_1, L_2, so $L_t = L_1 \cup L_2$. The search space L_0 is based on L_t but has to be enriched with novel concepts so that more generic concepts may be proposed in the final labeling:

$$L_0 = \bigcup_{c \in L_t} \text{anc}(c), \tag{3.6}$$

where $\text{anc}(c)$ is the set of all ancestors of the concept c and c itself. L_0 still reduces a lot the search space compared to the whole ontology by limiting it to the concepts already present in the documents and their ancestors. While we questioned the necessity of enriching A_0 with the ancestors in the last chapter, here it is more appropriate. There is a key difference in terms of intention between the annotation process of the previous chapter and the labeling of clusters. Here, the aim is to summarize, factorize, generalize a set of concepts into a smaller one while keeping it meaningful.

To this end, the introduction of concepts of higher abstraction should be encouraged.

The objective function $f(A)$ for annotating documents in Chapter 2 (see §2.4.2) was expressed as an objective and a constraint. $g(L)$ must be somewhat similar as it relies on the same foundations. In an HAC context, the neighbors of the node to label are in fact its two children that have been agglomerated. As a result, the adapted consistency objective is

$$\text{consistency}(L) = \frac{1}{2}\text{sim}_g(L, L_1) + \frac{1}{2}\text{sim}_g(L, L_2). \tag{3.7}$$

The penalty for keeping the list of concepts concise is also adapted as

$$\text{penalty}_c(L) = \mu|L|. \tag{3.8}$$

Finally, there is the objective regarding the global specificity of the set. It can be modeled as another constraint to add so that the final objective function $g(L)$ has a better control over the specificity & genericity of the output. We define it as the average specificity of L:

$$\text{penalty}_g(L) = \frac{1}{|L|}\sum_{c \in L} \text{IC}(c). \tag{3.9}$$

This means that the more the concepts in L are specific, the more penalty_g increases. Therefore, when L is pruned, concepts can be removed for two reasons:

- L is not similar enough to the labels of the inner clusters,
- L is too specific and it increases substantially the average specificity.

The final objective function is the consistency objective under the two constraints defined above, that is

$$g(L) = \text{consistency}(L) - \text{penalty}_c(L) * \text{penalty}_g(L), \tag{3.10}$$

$$g(L) = \frac{1}{2} \sum_{L_i \in \{L_1, L_2\}} \text{sim}_g(L, L_i) - \mu \sum_{c \in L} IC(c). \tag{3.11}$$

The two penalties are not summed but multiplied since we are looking for a label that is concise *because* it is generic. They should thus not be independent from each other. The expected behaviour of this function is the following. If a concept in L is novel, it may or may not be removed depending on the result of the trade off of genericity/similarity. If it is very abstract, $\text{penalty}_g(L)$ will decrease. However, $\text{consistency}(L)$ will decrease as well as it will not be very similar to the concepts labeling the inner clusters. On the other hand, a concept that is very specific will have to be well represented in the inner clusters for being kept in L. Again, this trade off is balanced by μ. With a high value of μ, the labels will tend to be systematically abstract while with a low value, they will remain the same as those of the inner clusters.

3.6. Algorithm and the study of complexity

As we want to provide a scalable method, we need to study its feasibility, particularly by inspecting its time complexity. We do not aim at ameliorating the HAC complexity in any way, since the aim is to explore the possibility of using semantic annotations of documents. Nevertheless, we need to make sure our approach has a comparable complexity. To this aim, this

section details the algorithm we rely on and its complexity.

3.6.1. Algorithm details

The algorithm is a greedy algorithm leading to a binary tree inspired by the very common Neighbor-Joining method in systematic biology (Saitou and Nei, 1987). Algorithm 2 details the whole process. During the initialization, the first clusters and the tree are created: each cluster is a leaf and its label is the annotation of the document it represents (l.5-9). The similarity matrix of trivial clusters is computed by using semantic similarities of their cluster labels (l.10-15). Then, the agglomerative process begins and iterates until there is only one cluster. Each iteration consists in finding the pair of the closest clusters Cl_x, Cl_y in the matrix (l.18). A new cluster Cl_{new} is created (l.19). In order to label the newly created cluster we can rely on the algorithm defined in the last chapter as only the objective function changes without introducing or removing parameters. We thus need to define the cluster's neighborhood. In this case, it is the list of its two children (l.20). This new cluster is then labeled by using the objective function defined in §3.5.2.2 (l.21). We also approximate and store the distance of this cluster to its children by computing the semantic similarity of their labels (l.22-23). This step allows us to build a tree with branch lengths representing the semantic similarities between the nodes. Note that the following equations assume that the SSM is bounded in an interval $[0;1]$, as most SSMs are (Harispe et al., 2015b):

$$d(Cl_n, Cl_x) = 1 - sim_g(Cl_n, Cl_x), d(Cl_n, Cl_y) = 1 - sim_g(Cl_n, Cl_y). \quad (3.12)$$

Finally, the similarity matrix must be updated. The new cluster is added to it and all the similarities with other clusters are computed. The rows and columns corresponding to the two agglomerated clusters are removed (l.24-29).

3.6.2. Complexity analysis

In order not to be redundant, the complexity analysis refers to some notions explained in Chapter 2. The algorithm 2 features $S_{max} \in \mathbb{N}$ as an input that aims at bounding the maximal size of labels associated to the clusters. In practice, S_{max} has been set to 20 in the experiments detailed in §3.9. This is subject to variations depending on the dataset because if documents are heavily annotated, S_{max} should be higher, although a document is more likely to be annotated by ten to twenty concepts in general. It is used as a parameter of *Annotate*$(\cdot)$ in place of the th input parameter. Note that the first lines of the algorithm guarantee that there is no inconsistency between D and S_{max}. Let us detail the few steps preceding the actual clustering of the documents. Creation of initial clusters (l.5-9) is in $\mathcal{O}(|D|)$. Filling the matrix requires to fill $\mathcal{O}(|D|^2)$ cells, each of which is computed in $\mathcal{O}(S_{max}^2)$ for the BMA composite average. Initialization of the algorithm (l.2-15) is thus made in

$$\mathcal{O}(|D|^2 S_{max}^2). \tag{3.13}$$

The clustering process consists of $|D|-1$ iterations during which three main processes occur. At each iteration, the matrix is browsed once to find the best cluster pair (l.18) in $\mathcal{O}(z^2)$ where z is the current number of clusters.

Algorithm 2: Clustering and labeling of a set D of documents

Function *Cluster* $(D, \mu, S_{max}, \theta)$
  Input : The set of documents to cluster D, a real number $\mu \in [0;1]$, the maximum size of cluster labels S_{max}, an ontology θ
  Output : The root node root of the tree
  if *One annotation size is greater than* S_{max} **then**
    print an error and exit;
  end
  clusters $\leftarrow \{\}$;
  for $i \leftarrow 1$ *to* $|D|$ **do**
    Create a childless cluster Cl_i with labels $L_i = A_i$;
    clusters $\cup \{Cl_i\}$;
  end
  Create a matrix M of size $2|D| \times 2|D|$;
  for $i \leftarrow 1$ *to* $|D|$ **do**
    for $j \leftarrow i + 1$ *to* $|D|$ **do**
      $M(i, j) \leftarrow sim_g(L_i, L_j)$;
    end
  end
  new $\leftarrow |D| + 1$;
  while $|\text{clusters}| > 1$ **do**
    Find the pair of remaining clusters Cl_x, Cl_y with the highest similarity;
    Create a new cluster Cl_{new};
    Create a list $K \leftarrow \{Cl_x, Cl_y\}$;
    Create $L_{new} \leftarrow \text{Annotate}(K, \mu, S_{max}, \theta)$ the labeling of Cl_{new};
    Add child Cl_x to Cl_{new} with a distance of $1 - sim_g(Cl_{new}, Cl_x)$;
    Add child Cl_y to Cl_{new} with a distance of $1 - sim_g(Cl_{new}, Cl_y)$;
    clusters $\leftarrow$ clusters $\cup \{Cl_{new}\}$;
    clusters $\leftarrow$ clusters $\setminus \{Cl_x, Cl_y\}$;
    for Cl_i *in* clusters **do**
      i $\leftarrow$ index of Cl_i in M;
      $M(i, \text{new}) \leftarrow sim_g(L_i, L_{new})$;
    end
    new $\leftarrow$ new $+ 1$;
  end
  root $\leftarrow$ cluster[1];
  return root
end

The label of the cluster is then computed by using a modified version of the annotating algorithm for which the complexity is $\mathcal{O}(knS_{d_{max}} + n^3)$ according to equation (2.18); where k was the number of neighbors taken into account, n was the size of the initial set A_0 and $S_{d_{max}}$ was the maximum size of annotation. For cluster labeling, the neighbors are the children of the new node so this value is constant ($k = 2$) and is removed from the complexity. $S_{d_{max}}$, the maximum size of a document annotation, is hereby replaced by S_{max}. The initial set for labeling a cluster, denoted L_0, should be calculated following equation 3.6. However, in order to control the scalability of the algorithm, we introduce a hypernymy parameter h that limits the search among ancestors. That is, $anc(c, h)$ are the h direct ancestors of concept c. As a result, the search space L_0 is in $\mathcal{O}(hS_{max})$. The adapted complexity is thus $\mathcal{O}(2(hS_{max})S_{max} + (hS_{max})^3) = \mathcal{O}(h^3S_{max}^3)$. The third and last important task is the computation of new semantic similarities with the $(z-1)$ other clusters in $\mathcal{O}(zS_{max}^2)$. Consequently, the complexity of each iteration in the while loop is a browsing of the matrix, a labeling of the new cluster and the computation of new similarities, leading to an overall complexity of each while iteration (l.18-29) of

$$\mathcal{O}(z^2 + h^3S_{max}^3 + zS_{max}^2). \tag{3.14}$$

The complexity of all iterations of the while loop, which dominates the one of the initialization and thus the time complexity of the whole algorithm, is hence:

$$\begin{aligned} &\mathcal{O}\left(\sum_{z=1}^{|D|}\left(z^2 + h^3S_{max}^3 + zS_{max}^2\right)\right) \\ &= \mathcal{O}\left(|D|^3 + |D|h^3S_{max}^3 + |D|^2S_{max}^2\right). \end{aligned} \tag{3.15}$$

As complexity is particularly crucial for large datasets, i.e. large $|D|$ and $|\mathcal{C}|$, S_{max} can safely be considered smaller than $|D|$ leading to

$$\mathcal{O}(|D|^3 + |D|h^3S_{max}^3). \tag{3.16}$$

We also investigated another heavier but potentially more accurate approach for clustering documents. Instead of labeling a new node by considering its two children as neighbors, we propose an alternative for which the set of neighbors is the set of all documents contained in this new cluster. Indeed, when a new node is labeled, there is a chance that some bias or noise is introduced by our heuristic. As a result, high-level nodes may suffer from accumulated imprecision that would decrease annotation accuracy, hence impacting the clustering quality. Using all observations for annotating the node is a solution to make sure that full evidence is taken into account when labeling nodes. Let us study the complexity of such an approach by using the one previously detailed. The only difference is the definition of L_0. It now also depends on the maximal size of the new cluster which increases by 1 at each iteration of the while loop, that is, the search space L_0 is in $\mathcal{O}(zhS_{max})$. The adaptation of the labeling complexity is thus $(zhS_{max}^2 + (zhS_{max})^3) = \mathcal{O}((zhS_{max})^3)$ instead of $h^3S_{max}^3$. When plugged into to the equation 3.14, this results to an iteration complexity in

$$\mathcal{O}\left(z^2 + (zhS_{max})^3 + zS_{max}^2\right) = \mathcal{O}\left((zhS_{max})^3\right) \tag{3.17}$$

and the complexity of the whole algorithm is

$$\begin{aligned} &\mathcal{O}\left(\sum_{z=1}^{|D|}\left((zhS_{max})^3\right)\right) \\ &\quad = \mathcal{O}\left(|D|^4 h^3 S_{max}^3\right). \end{aligned} \tag{3.18}$$

From now on, the first approach is denoted by Light Semantic Clustering, or LSC, and the second one is referred to as Heavy Semantic Clustering, or HSC. While LSC has a time complexity rather similar to that of classical approaches (typically, $\mathcal{O}(|D|^3)$), HSC is one order above. However our approach here is exploratory as it mainly aims to identify the gains and losses of relying on semantic similarities combined with automatic labeling. In fact, if HSC turns out to be the most accurate approach, then it can easily be optimized to get a reasonable time complexity. Here are several optimization hints to this end.

1. *Using the values computed in previous iterations.*
   As for the previous chapter, some calculations are needlessly repeated. Since the algorithm follows an agglomerative strategy, some computations can be stored, reused, and updated only when needed (recall the SumMaxCols and SumMaxRows).

2. *Sampling the neighbors.*
   We can also consider to only take a sample of fixed size k of the observations contained in the node to annotate it. This would give a complexity similar to that of LSC while approximating HSC.

3. *Building L_0 differently.*
   The high complexity of HSC is due to the definition of L_0, particularly

because it merges all annotations of the observations in the node to be labeled. However, another way to proceed is to define L_0 as in LSC, that is by taking the union of the labels of the two children of the node. The objective function would still consider all documents within the cluster as neighbors, but the dimensionality of the search space would be greatly reduced.

We are thus convinced that the high complexity of HSC can be reduced. Before studying its behaviour comparatively with LSC and a classical HAC method, let us explain an important post-processing aspect of the output tree and detail the evaluation benchmark.

3.7. Post-processing

The algorithm per se produces a binary tree which is difficultly exploitable by a user. Grouping the clusters by pairs is legitimate for the algorithm complexity, however, a user would expect larger categories to appear instead of a dense binary tree. Figure 3.6 shows the problem. The picture in (a) is an example of output of such an algorithm. As the distance of each node of the tree to its children is computed, it makes it possible to represent branches with various lengths. This would help the user to intuitively identify the groups of clusters that may make sense (b).

A post-processing can automate the cluster definition instead of leaving this task to the user. The post-processing algorithm consists in flattening the tree in certain areas in order to provide a tree as in (c). The distribution of distances in a tree is often bimodal with a peak for low values—

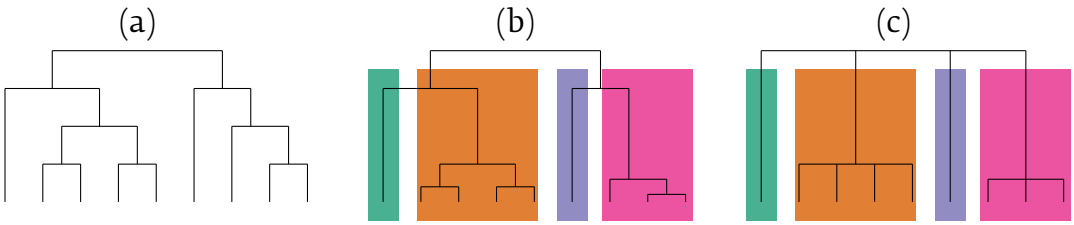


Figure 3.6.: With no branch length (a), it is hard to visually determine the most meaningful clusters. When branch lengths are taken into account (b), clusters clearly appear. The post-processing algorithm we propose produces a tree such as (c) so that the most meaningful clusters are automatically determined.

representing intra-cluster distances—and another peak for high distances—representing inter-cluster distances (see Figure 3.7).

Clusters should be flattened in areas containing extreme values in those peaks. It seems obvious for the low distance areas: when nodes are close to each other, they are merged. However, it is a little trickier for the opposite, that is, when nodes are extremely distant. In fact, despite the fact that some clusters are very different from each other, they were closest in the matrix at some point and they have been gathered. Therefore, they share a common parent. This situation occurs mostly near the root. For example, consider Figure 3.8, a tree with two children at the root. The first child is labeled COOKING, the other one is labeled {COMPUTER SCIENCE, TRAVEL}. In {COMPUTER SCIENCE, TRAVEL}, there are two clusters, COMPUTER SCIENCE and TRAVEL. During the last iterations of the algorithm, the clusters labeled COMPUTER SCIENCE (CS) and TRAVEL where the closest. This does not mean they are close—besides, the labels of their parent are confusing—, and it seems more appropriate to have three clusters at the root, such as COOKING, COMPUTER SCIENCE and TRAVEL. All of them should be on the same level as they all are major topics. We thus alter the tree to remove

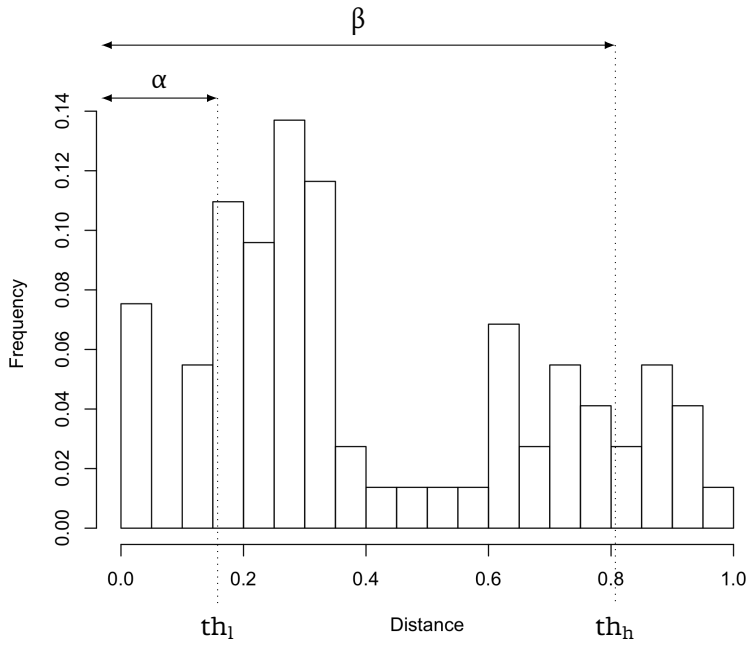


Figure 3.7.: Example of a distribution of distances in a clustering tree from the benchmark dataset (see §3.8) with the representation of the α, β quantiles.

Figure 3.8.: The limitation of relying on an agglomerative algorithm. CS = computer science. The tree obtained on the left is the result of an agglomerative process were the CS and TRAVEL clusters have been gathered although they are very distant (branch lengths are not shown). Instead, the tree on the right is preferable and requires to flatten the tree by removing the {CS,TRAVEL} cluster.

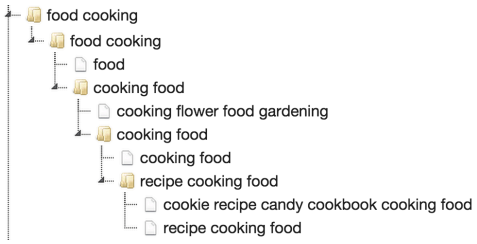
the nodes (here {COMPUTER SCIENCE, TRAVEL}) for which children (here COMPUTER SCIENCE and TRAVEL) and parent (here ROOT) are very different as well. Another reason for flattening is the similarity of cluster labels. Indeed, when two nested clusters have the same labels, the algorithm merges the lower cluster content into the top one. This specific case occurs on two levels while the previously explained one works on three levels.

The algorithm that does this task is pretty straightforward. First, we need to find the threshold values for which distances may lead to a flattening. As the distribution of distances may vary depending on the dataset and the tree, we define two quantiles α, β. Their definition is based on a training set (see §3.8.2.2). For example, with the training set of the benchmark proposed further (see §3.8), best results are obtained with $\alpha = 0.2$ and $\beta = 0.87$. Second, we define two threshold values th_l, th_h for the low and high values below or above which distances will be considered during the flattening. A recursive function browses the tree with a top-down strategy. For each node it browses, if its parent and its children all have low or high dis-

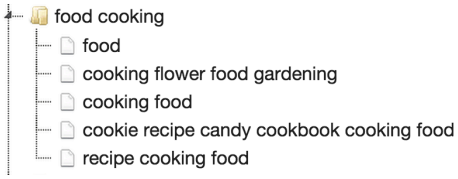
tances with it, then it branches the children nodes to the parent node and removes the current node. Finally, the algorithm stops when it reaches the leaves and returns the processed tree. Figure 3.9 shows parts of a tree before and after this process. We see that some inconsistencies seem to remain in the tree because some items are outliers and cannot be clustered correctly.

3.8. Creation of a benchmark

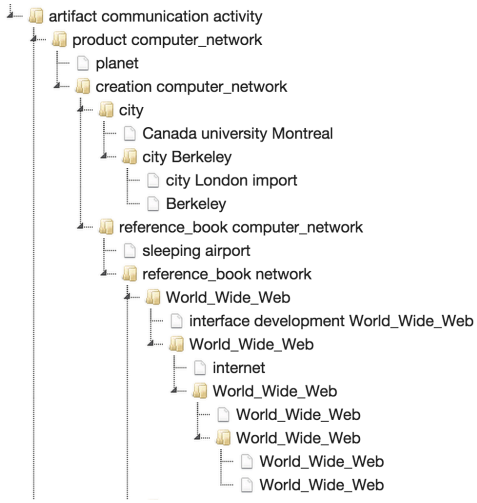
To the best of our knowledge, there is no work that focuses on clustering semantically annotated documents. In fact, the papers presented in the Related Work section do not make any comparison with other approaches on the basis of a common dataset and metrics. We can only identify the novelties proposed by each method without clues as to their comparative performances. Besides, the specificity of each method makes it even more difficult to compare them, as many of them rely on texts with only a few relying on other documents such as annotated genes. As we definitely want to test our approach, we thought about developing a benchmark and comparing several methods, including ours, on this benchmark. Unfortunately, the source code of state-of-the-art approaches is rarely available and often also ontology-specific. As a result, we propose a benchmark built upon previous bookmark annotations, curation and semantification with WordNet. Hopefully, this will provide a useful dataset for comparing clustering methods.



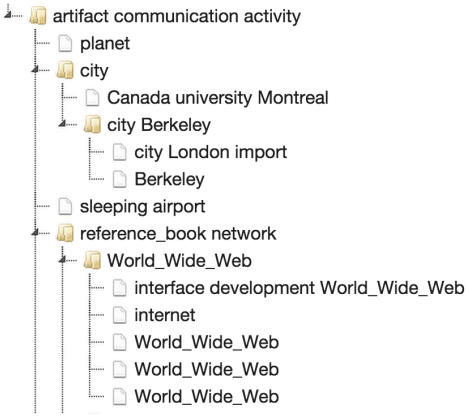
(a) Clusters about food and cooking.



(b) All clusters merged into one.



(c) Clusters of very different topics gathered because of an agglomerative algorithm.



(d) Very distant and close clusters have been flattened. Some inconsistencies remain but the result overall seems better.

Figure 3.9.: Different parts of a tree before and after the post-processing. Folder icons represent clusters and file icons represent documents. The text associated with each document is its semantic annotation.

3.8.1. Original data

Finding relevant datasets for evaluating those approaches turned out to be a very hard task. There are many textual collections used for benchmarking clustering (Rossi et al., 2013) and some papers that evaluate cluster labels (Role and Nadif, 2014; Carmel et al., 2009). In many cases, the information available on the documents to cluster is only their content. The same problem occurs regarding labeling, for which the text is used to extract the labels of a given cluster. On the other hand, increasingly more documents tend to be semantically annotated. The most obvious ones are the biomedical papers for which the annotation has been the main topic of Chapter 2. Evaluating the results of a clustering of biomedical papers would however demand a very high expertise that we do not have in the team. Such results would hence be barely interpretable.

Wetzker et al. (2008) highlighted the fact that social bookmarking may lead to very interesting sources of information. They also stress that it can be subject to spam and propose some guidelines to make sure that the crawled data are not biased by this spam. They created a corpus from a dump of `del.icio.us` that associates users to the bookmarks they created and their annotations. An Italian team (Andrews et al., 2011) followed with this study by proposing a semantic version of a subset of this corpus. The main difference is that all annotations of the bookmarks have been mapped to WordNet, instead of simply being words. This work has been done by hand, providing a high quality corpus at the cost of a limited size.

A real use case application can, however, be designed. Say a user wants to reorder his/her bookmarks. Those bookmarks are annotated by concepts

from WordNet 3.0. Here, the annotation has been made by hand, but one can imagine an automatic concept extraction from the bookmarked website. The clustering approach we propose can create a tree representing the bookmarks of the users with their respective category names. Such an application makes sense to anyone and creating a benchmark based on this dataset seems to be much easier than on biomedical papers.

In order to create the evaluation benchmark, we had to curate the semantified `del.icio.us` dataset. This dataset originally contains 739 URLs—we will refer to them as bookmarks from now on, although a bookmark is generally associated with a user. Those bookmarks are not directly annotated with WordNet URIs so we had to automatically map them from Knowdive, a dataset provided by the authors, to WordNet 3.0 by using the specified *closeMatch* relationship. Since some annotations did not have any close match in WordNet 3.0, we manually reviewed the URLs encoutering this problem. To that end, some bookmarks had to be removed because we were unable to match some of their concepts manually. The rest (591 bookmarks) have been used to create the benchmark, consisting of 8 subsets of about 74 bookmarks each. One of them is a training set that should be used for configuring the method. The seven others are evaluation sets that will be used to score the results.

3.8.2. Obtaining expert data

The use case of the benchmark is pretty easy to understand and anyone with a bit of experience with the Internet should be able to cluster and label a set of bookmarks. Therefore, we created a tool easing those tasks and called for participation in this project.

Figure 3.10.: Example of a bookmark identified by its ID 218110 and annotated by three concepts. The signature on the right gives a simpler representation.

3.8.2.1. Interface details

Tool: http://clustering.creatox.com

Documentation: http://clustering.creatox.com/documentation.html

The purpose of the tool is basically to propose annotated and anonymized bookmarks to the user and ask him/her to classify them. The first screen is a login access so that any volunteer can create a new project or load an existing one. Then, the main page of the project is displayed.

The tool proposes a bookmark as an identifier and a set of concepts from WordNet. There is also the signature of the bookmark, which is a visual representation of the concepts that annotate this bookmark created as follows. For each dataset, a matrix containing the similarities of all concepts annotating the bookmarks is computed. It is then used for a multidimensional scaling analysis that outputs a projection, maximizing the distribution of the concepts in one dimension. The coordinates are used to find a corresponding color by converting them in wavelength and then colors. Finally, for each presented bookmark, a rectangle shows the list of concepts by colored vertical lines (see an example with Figure 3.10). Intuitively, one can quickly understand the color associated to any big topic of the dataset—e.g. COOKING is red or COMPUTER SCIENCE is blue. It also helps to make sure there is no inconsistency in the clusters. If a signature is very different from the other ones in a cluster, the user should wonder if it is well placed.

It is possible to create clusters, delete them and put them into other clusters. Everything is based on drag and drop gestures. There are several representations of the clusters as illustrated in Figure 3.11: (a) hierarchical, as a tree without the bookmarks, (b) pseudo hierarchical, with in each cluster the frequency of concepts and the list of inner clusters and (c) flat, with the list of bookmarks in each cluster. Each representation has it own benefits. (a) gives a nice overview, (b) allows to quickly identify the main concepts in each cluster and (c) allows the user to check for mistakes, inconsistencies or potential refinement. Since all signatures are aligned, it is very easy to identify them, as showed in Figure 3.12. On the left side, a single cluster named "education" is displayed. It appears that some items in this cluster share something else in their signatures, that corresponds to "academy", "university", "administration", "school", "institute". The interface only highlights these facts according to their semantic similarity and the user can chose to split this cluster into two, namely "education" and "organization". The latter would certainly be included in the former as nearly all documents of these clusters refer to "education" or something close.

When a cluster is created, the tool asks for a name. It should be something that makes sense to the user at first. When the clustering is completed, that is, when all bookmarks are ordered in clusters and clusters are hierarchically arranged, the tool requires to semantify the labels of the clusters. For each cluster, one can look for corresponding concepts in WordNet and pick concepts for labeling it. This task is helped by an autocompletion tool. The user simply has to type the beginning of the concept he/she would like to enter and a list of corresponding concepts from WordNet is proposed. This avoids asking the users to search on WordNet by themselves. Once this is done, a small survey asks the user several questions about her expe-

rience with the tool. This includes:

- how long it took to complete the project,
- how confident she is with the clustering,
- how confident she is with the labels,
- how easy or difficult is was to use the tool.

3.8.2.2. Benchmark

12 people answered the call for participation. Some are lecturers (5) or PhD students (3) from the école des mines d'Alès. Several students (4) were also willing to participate. None of them had any knowledge of the content of the dataset prior to participation.

Let us first study their feedback on the process (see Figure 3.13) leading to 19 trees overall. The average confidence in their clustering is 62.1%. This proves that the task is not easy because the items are hard to classify. Users are slightly more confident with their labels (65.9%), some mentioned that finding the accurate concepts in WordNet might be difficult. The other reason for this score is that it may be difficult to give labels to high-level clusters that aggregate groups of various topics. Most of the trees (63.2%) were completed in 30 to 60 minutes. It took between 1 and 3 hours for some others (31.6%) and 3 to 5 hours for the rest (5.3%). The average user-friendliness grade users gave to the tool is 76.7%. This score is quite good but it also shows that some things could probably still be improved. For example, the clusters take a lot of space and users have complained about the need to scroll frequently to access the element they want.

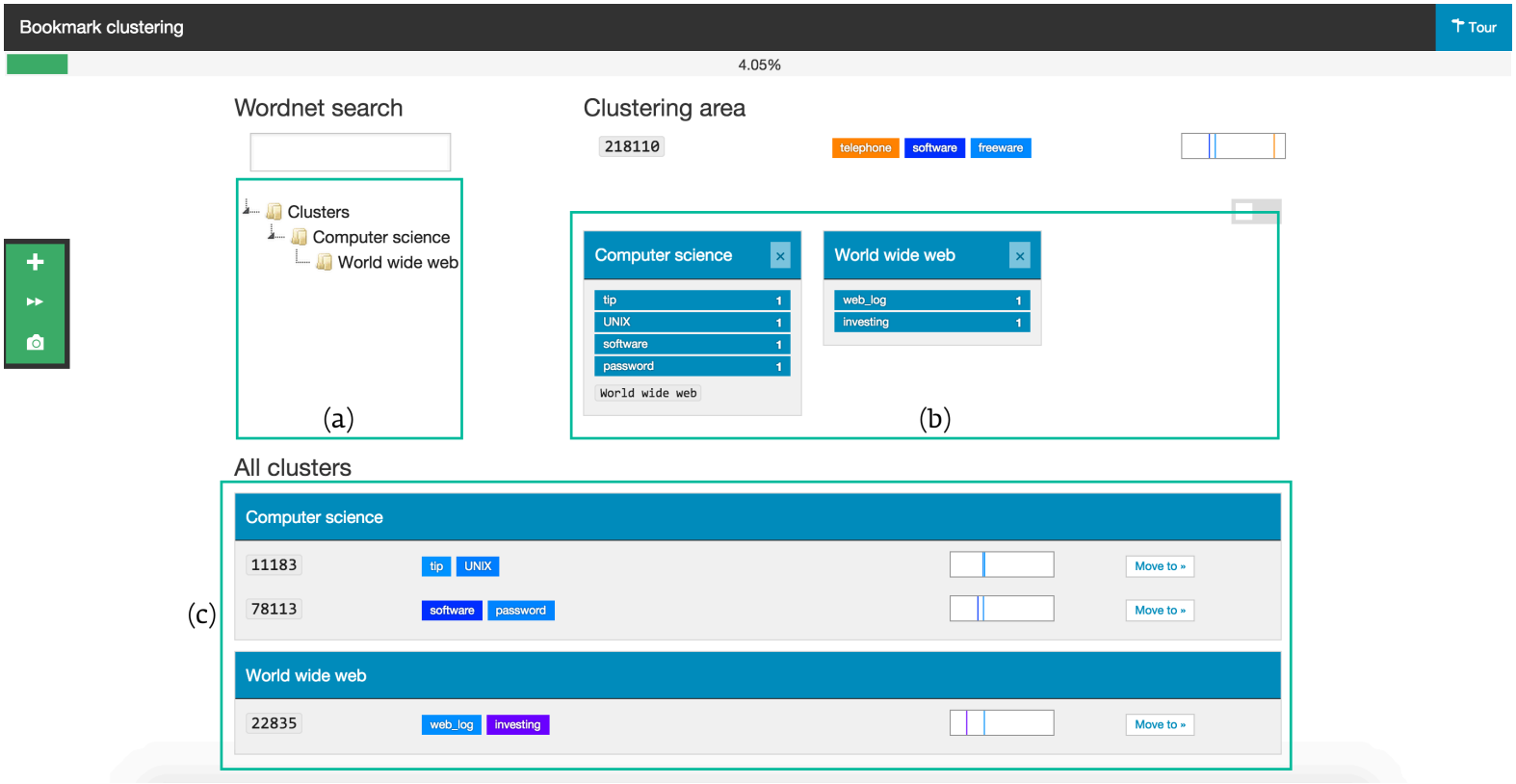


Figure 3.11.: The interface for the creation of the benchmark. Three cluster representations are provided: (a) hierarchical, as a tree without the bookmarks, (b) pseudo hierarchical, with in each cluster the frequency of concepts and the list of inner clusters and (c) flat, with the list of bookmarks in each cluster.

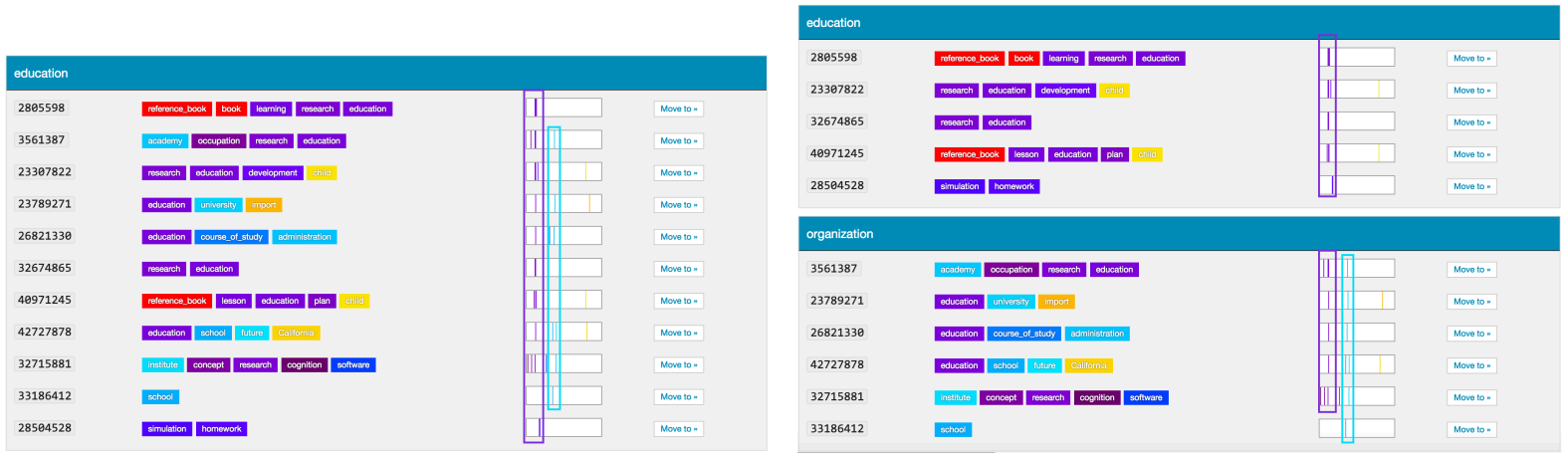


Figure 3.12.: The bookmark signatures helps in identifying new clusters. On the left, the user created a single cluster related to "education". It appears that some bookmarks share something apart from "education" and these could be classified in another cluster labeled "organization" (right side), certainly embedded in "education".

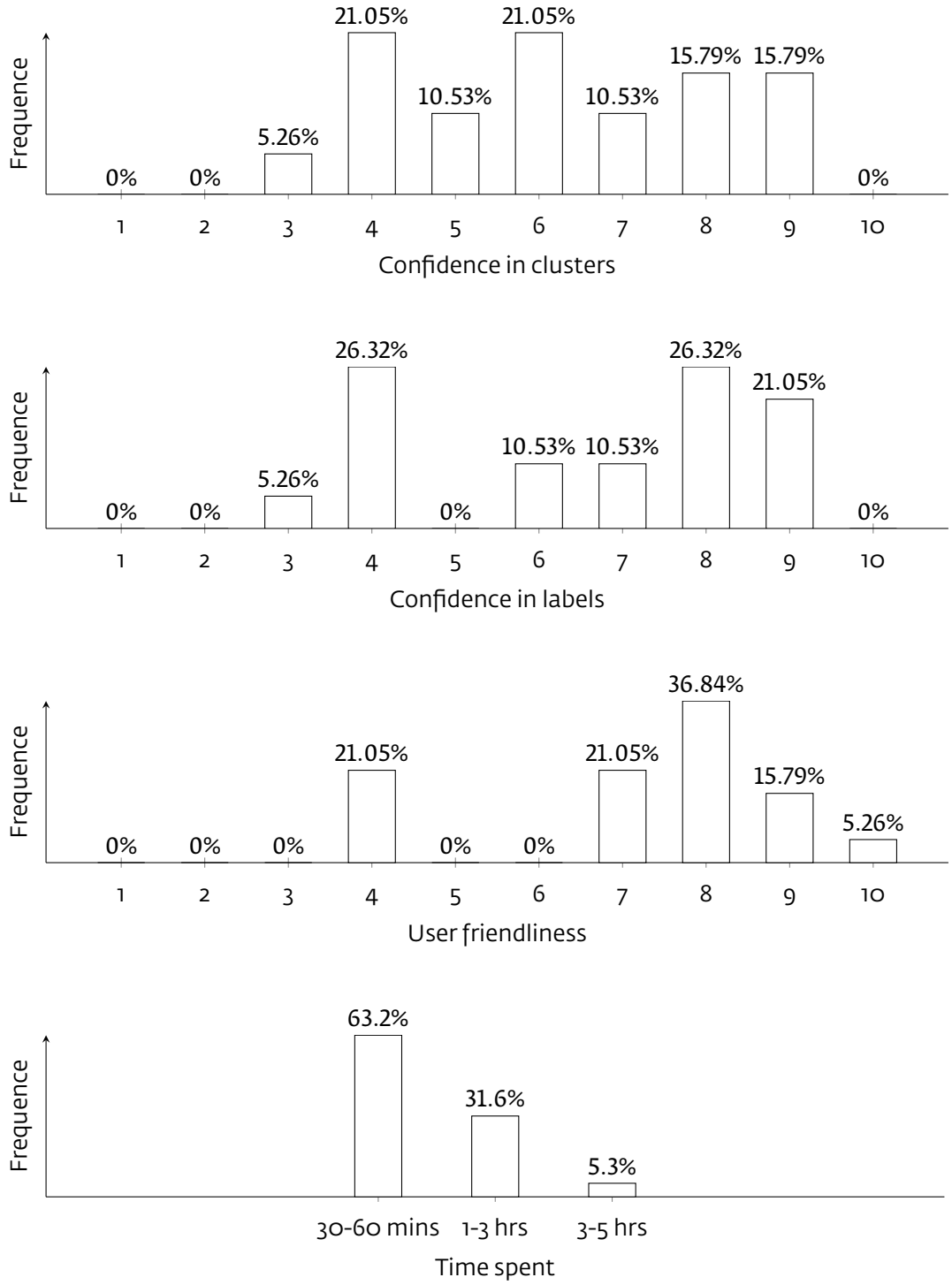


Figure 3.13.: Users feedback on their experience using the clustering interface.

For each dataset, we thus have several clusters and labelings. All datasets have been processed by one-participation and multiple-participation testers. Therefore, every dataset benefits from a point of view similar to some others and from unique insights provided by one-shot testers. The trees have a depth of 4 at most and contain 20 clusters in average.

For evaluating the clusters, we propose to compute the average distance of the computed tree from the expert trees. Precision and recall of clusters also seem obsolete for this task as we aim at evaluating the whole tree structure—and not only the clusters for a given cut-off. Some tree distance algorithms have been elaborated in the literature and we use one of them to calculate how different a pair of trees is. As a result, for each dataset, we suggest to compute the *standard deviation* of expert data, provided that there are about three expert opinions per dataset. If the tree that our algorithm outputs is on average as different from the expert ones as the expert trees are from each other, this means that our method has an efficiency comparable to a human for this task—under the same conditions.

Labels can be evaluated following a more classical method. Here, an F-measure is applicable to score them, although using a semantic similarity would be more appropriate for the same reasons given in §2.2.3.2. As the labels depend on the clustering and vice versa, in order to evaluate the labeling, we must label the expert trees in order to see whether or not those labels are appropriate.

3.9. Evaluation of clustering and labeling

3.9.1. Results of clustering

The benchmark data has been created by using CompPhy, an online platform that proposes many tools for manipulating trees (Fiorini et al., 2014a). It is designed for phylogenetic projects at first but it can well handle any kind of tree as long as the format is respected. Overall, the benchmark comprises eight datasets, one of which is meant to be used for tuning the methods while the others are supposed to be used solely for evaluation. This way, the optimization/tuning of the technique is not made on the evaluation datasets but on an independent learning set. Regarding the evaluation metric, the trees can be compared using the Robinson-Foulds distance (Robinson and Foulds, 1981) that calculates the topological differences of two (non-)binary trees. The output is not normalized and represents the number of basic operations needed to transform a tree into the other one. In order to normalize it, we use the statement in Pattengale et al. (2007) that an unrooted tree of n leaves induces at most $2n - 3$ clusters. As all datasets have a variable number of documents, the similarity of two trees t_1, t_2 is thus estimated as $\mathsf{TreeSim}(t_1, t_2) = \frac{\mathsf{sim}_{RF}(t_1,t_2)}{\mathsf{len}(t_1)+\mathsf{len}(t_2)-3}$, where $\mathsf{sim}_{RF}(t_1, t_2)$ is the Robinson-Foulds distance of t_1, t_2 and $\mathsf{len}(t1) = \mathsf{len}(t_2)$ is the number of leaves (documents) in the trees.

Let us evaluate our approaches called Light Semantic Clustering (`LSC`) and Heavy Semantic Clustering (`HSC`). First, Table 3.2 summarizes the average distance among expert trees (`expert`) and the average distance of `LSC` and `HSC` trees with the expert ones for each dataset. It clearly appears that `HSC`

outperforms LSC on all datasets.

	D1	D2	D3	D4	D5	D6	D7
expert	0.138	0.156	0.168	0.197	0.159	0.131	0.166
HSC	0.131	0.202	0.189	0.195	0.231	0.190	0.193
LSC	0.255	0.271	0.257	0.257	0.259	0.293	0.276

Table 3.2.: The average distance of expert trees from each other and of HSC and LSC (after the post-processing) from the expert trees for each dataset (D1...D7).

We conducted a more thorough study on the behaviours of LSC, HSC and a classical HAC clustering—that we further refer to as the baseline—based on semantic similarities. The baseline is built upon a semantic similarity matrix (the same that we use in our algorithm) but uses the average linkage criterion afterwards instead of the semantic similarity between node labels. Three strategies are explored in the post-processing: without flattening (none), with a flattening of small branches only (half) and with a full flattening (full). By default, the baseline approach provides a cluster tree without branch length or cluster labels. In order to correctly assess the benefit of the post-processing for baseline, its clusters are thus annotated using the same labeling strategies as HSC, that is, based on all contained documents. Figure 3.14 illustrates the results obtained with the two methods, following the three strategies, on the seven datasets. It also shows the standard deviation of expert data to ease the comparison.

The results show that the flattening is an important process for all approaches and that the full process (i.e. merging both short and long branches) outputs a better tree. Note that merging close nodes (half versus none) gives a better improvement than that of merging distant ones (full versus half). With no post-processing, all methods are quite comparable. When close

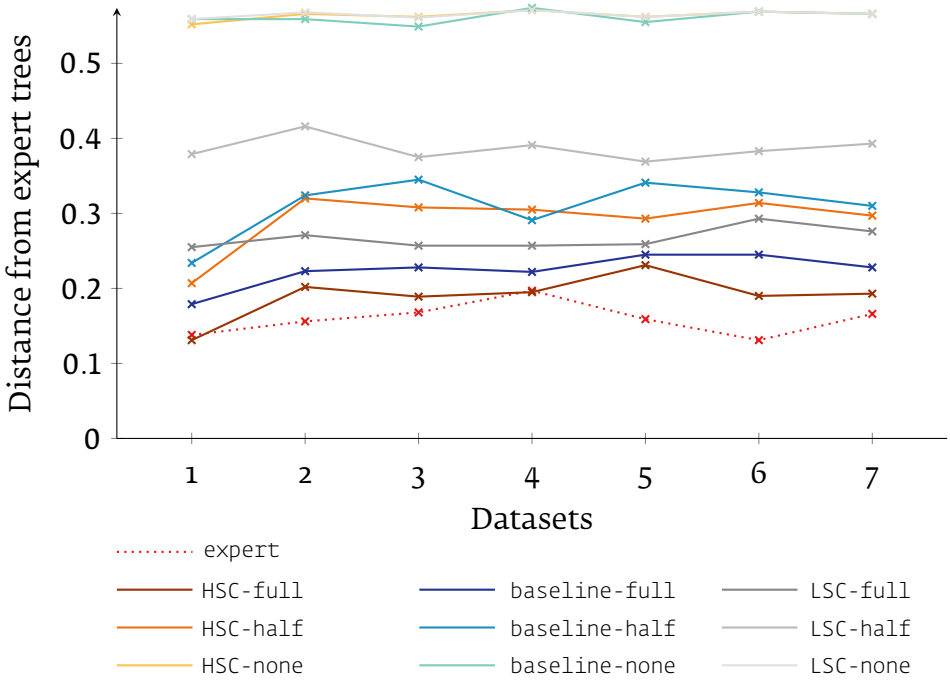


Figure 3.14.: Comparison of semantic clustering (`HSC` and `LSC`) with the classical approach (`baseline`) combined with the three strategies `none`, `half` and `full`. Expert standard deviation `expert` is also displayed. Lower values are better.

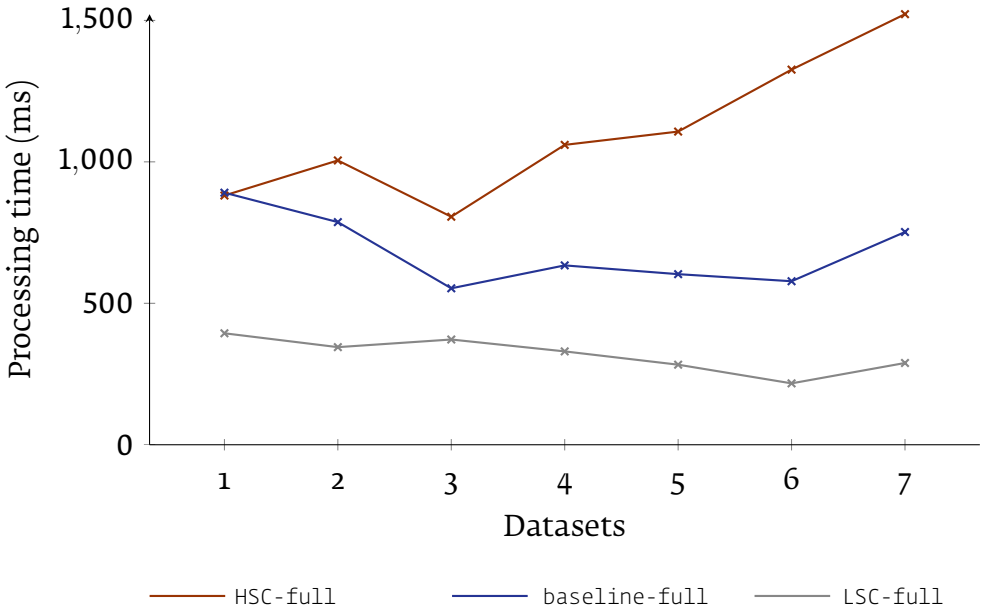


Figure 3.15.: Comparison of the semantic clustering (HSC and LSC) with the classical HAC (baseline) processing times on a 2.9Ghz and 8GB of RAM machine. Lower values are better.

nodes are merged, HSC performs slightly better than the classical approach on some datasets and LSC is not as good as these two. When the full post-processing is applied, HSC systematically performs better than baseline—here again, LSC provides less good clusters. In fact, the results of HSC are very close to the standard deviation that represents the experts' discrepancies. On two datasets (1 and 4), the trees provided by HSC present the same distance to the expert ones, consistent with the standard deviation.

The computation time difference between the approaches is very variable as shown in Figure 3.15. All algorithms include the post-processing and the automatic labeling of the clusters based on all documents within the clusters. The computation times have been obtained by running the algorithm 100 times for each dataset and for each method. As expected, baseline is faster than HSC. Interestingly, LSC is even faster than baseline. This

is due to the fact that the `baseline` approach has not been optimized either and the computation of the average link criterion takes time at each new cluster. Normally, the execution time of this approach should be similar or smaller than that of `LSC`.

Overall, a few conclusions can be drawn from these results. `HSC` outputs better trees than `LSC` and `baseline` by producing better cluster labels at the cost of a higher complexity. However, we are confident that an intermediate variant can be designed to be almost as fast as `LSC` and as accurate as `HSC`. We are currently exploring various leads including algorithmic optimization of `HSC` detailed in §3.6.2 or a variant of `LSC` that takes the two cluster sizes into account during the new cluster labeling. We are hence confident that we will soon be able to propose an approach that would output nice results while being more scalable. Now that clustering has been evaluated on a few criteria, let us study the results of the labeling part.

3.9.2. Results of labeling

Unfortunately, it is impossible to simultaneously evaluate the clusters along with their labels. Indeed, the experts provide different trees with different labels that can thus not be summarized in a gold standard labeled tree for each dataset. Cluster labels are thus evaluated as follows. An expert tree is input in accordance with our method and the part that is supposed to annotate each node is run. The result of such an approach is a label for each node of each expert tree for each dataset. The evaluation of labels follows our previous work proposal (Fiorini et al., 2015c), in tune with other studies that suggested that semantic similarities better assess the quality of a semantic annotation (Névéol et al., 2006). For each expert

tree, we thus compute the average semantic similarity of our labels with the expert ones. Then, the scores for the trees are averaged for each dataset. Consequently, we propose a label score for each dataset.

As explained in the previous sections, it is difficult to compare our approach with other studies because of the lack of availability or because these approaches are too specific. We propose to evaluate our approach `CL` against two baselines that are intuitive alternatives for labeling the clusters. The first proposal `merge` consists in simply taking the union of all concepts annotating the documents within the cluster. In other words, `merge` is L_0 without inclusion of the ancestors and without execution of the algorithm. `flat`, on the other hand, executes the labeling algorithm on the same set as `merge`, that is without including the ancestors. Finally, `CL` executes the algorithm on L_0 as defined in the algorithm details, i.e. considering the ancestors of each concept annotating the documents within the cluster. Results are proposed in Figure 3.16.

At first glance, the scores seem to be pretty close. In fact, it mostly depends on the datasets. The worst scores are obtained with dataset 2 for all methods. `CL` and `merge` perform alike and better than `flat` on this dataset. Overall, `CL` gives better results although it is surpassed by `flat` for Dataset 6. While `merge` is a naïve approach that agglomerates all annotations of the documents in the cluster, `CL` and `flat` both are based on on USI algorithm, which is a more elaborated method. Consequently, these two approaches provide the best labels, but we note that the inclusion of the ancestors in our algorithm leads to better scores except for dataset 6. On average, the semantic score of `CL` is 0.49 and that of `flat` is 0.48. The difference between them is thus small but this result proves that using the hypernyms still im-

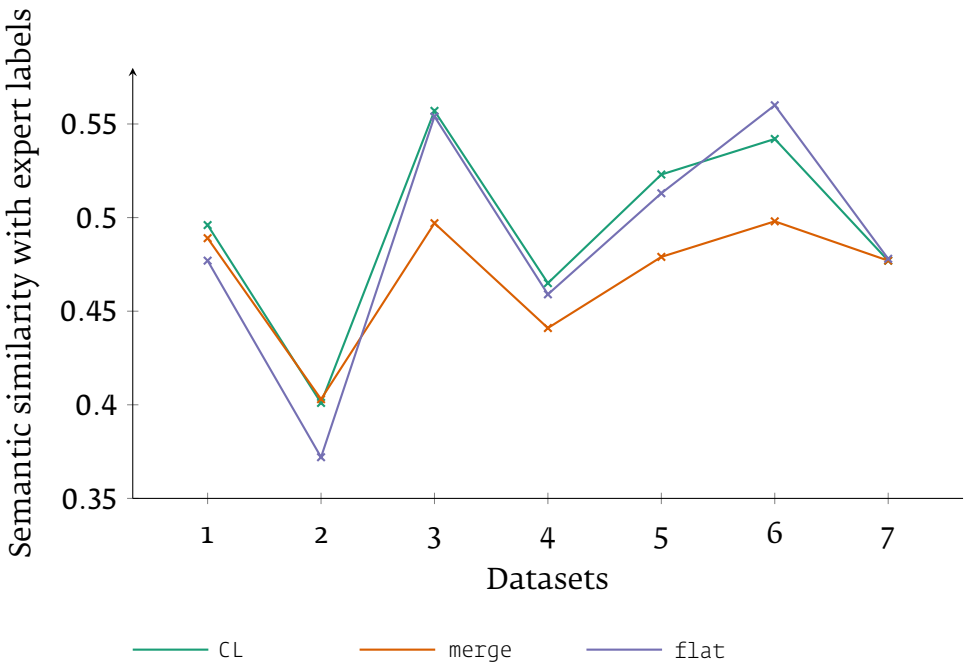


Figure 3.16.: Comparison of the cluster labeling (CL) with the baselines `merge` and `flat`. Higher values are better.

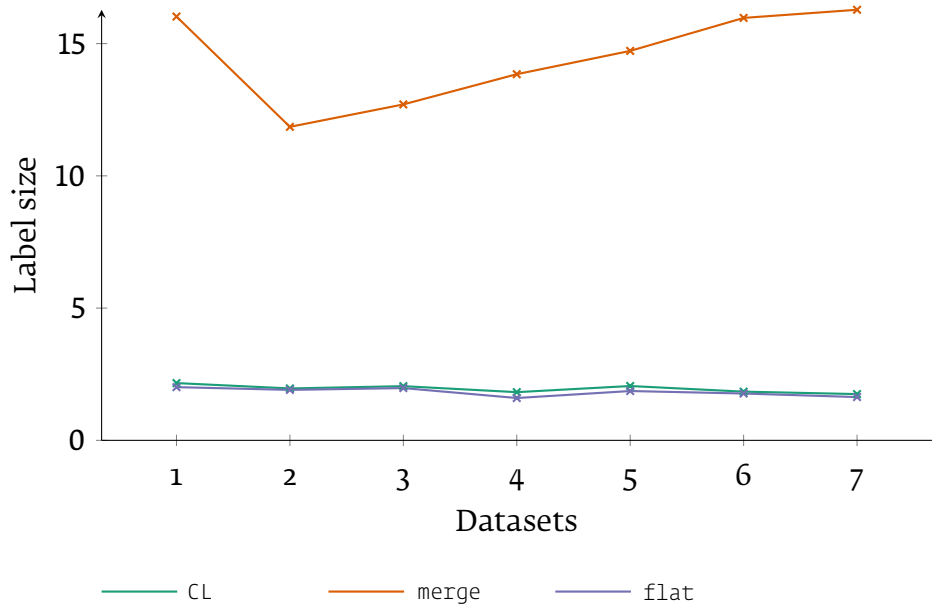


Figure 3.17.: Comparison of the label size of our cluster labeling approach (CL) with the baselines `merge` and `flat`. Lower values are better.

proves the quality of the labels in some cases like cluster labeling. Figure 3.17 illustrates the label sizes for each method. The `merge` approach contains many more concepts than the two others as it does not process them. The sizes of labels for `CL` and `flat` compared to that of `merge` demonstrate that we can clearly summarize 10 to 15 concepts into 2 concepts on average without decreasing the quality of the labels—in fact, it even increases it. Note that even when considering ancestors, the final size does not change much.

We observe that the scores in general are less good than for document annotation, for example, where we usually get semantic scores around 0.8. The reason here is that there are outliers in each dataset that are hardly clusterable because their annotation is vague or has nothing in common with other documents. Some experts gather these outliers in a cluster labeled VARIOUS, some others strive to find them a category. With the former strategy, our algorithm cannot predict this cognitive strategy and fails at labeling the cluster. In the latter, the result is the same because usually, once the expert has put the outlier in a cluster, he/she ignores it to annotate the cluster. When our algorithm annotates a cluster it can possibly ignore outliers when they are really underrepresented—because there must be enough leaves to veil them. When outliers are removed from the expert trees, the semantic score of labels is of 0.78 on average, which proves that our method is hampered by outliers. However, to the best of our knowledge, this is the only approach that proposes to summarize a group of concepts (contained in the leaves) into a smaller set of concepts.

The choice of hierarchical clustering is motivated by further applications. In fact, this work has been done to explore a bigger picture, which is the

complementarity of the nodes in the tree that is conveyed by the cluster labels.

3.10. Complementarity of labels

In some work about clustering or graph partitioning, people are looking for potential overlap after having separated the groups (Crampes and Plantié, 2014). That is, once the groups have been identified, trying to find the items that may belong to several clusters. The same kind of question is relevant for hierarchical clustering. Since the clusters are not definitely separated, we can observe the tree at different levels. For example, two clusters labeled ITALIAN FOOD and FRENCH FOOD may belong to a cluster labeled FOOD.This means that at a given depth or granularity, these are two distinct clusters but they are actually quite close. This reminds of the idea of an overlap for community detection and in a more general way, the concept of complementarity among clusters.

Complementarity of documents may be a useful concept for various applications. For instance, say a company is looking for two people for analyzing patterns in the DNA that may be related to a disease. What profiles does this company need? Can two biologists achieve this goal, or two computer scientists? Even though this example is very simple, it shows that the company will focus on candidates with the best synergy. This means the candidates who have the knowledge that best covers the subject: in DNA sequencing, in algorithms, in NLP, in the specific disease, etc. This also entails candidates who can communicate with each other by sharing some abilities/knowledge. This concept of providing the best coverage of a field

while having an overlap is a goal of hierarchical clustering which this work may help to reach. Imagine you have a hierarchy of skills, rich enough to be used as an ontology. A company could *annotate* its employees with those skills and cluster the employees. This seems feasible as LinkedIn for example uses a graph of skills and mapping skills from CVs to such a structure seems pretty easy. Then, when the company needs to hire a few people, the managers can add the candidates in the company cluster to see where they are branched: do they form new clusters? Are they included in the existing ones? In terms of human resources, this kind of analysis would certainly be extremely useful.

The way the trees we create are presented also allows many inferences in the cluster definition. Let us take the example of clustering users belonging to an e-commerce website. This clustering relies on the items people bought, considering that each item is annotated with concepts describing the company catalog. We can think of an advertisement campaign aiming at promoting some products of a specific brand. The question is about the people to whom the company should send this advertisement. The goal of the campaign is to promote the products only to people that will be interested, otherwise future emails from this company will be blacklisted by users who are annoyed of receiving uninteresting email. Defining the list of interested customers is made easier by such a hierarchical clustering. If the product refers to ELECTRONICS in general, then the advertisement should be sent to anyone of this category. But the granularity of the category can vary, for example with a product that will be of interest for people who like SMARTPHONES only. Besides, the promotion itself differs, as in one case, we present different electronic products and their general use, while, in the other, we specify exactly why those smartphones should

be bought. That is, depending on the group we address, the speech will be different. I too, define my PhD subject differently whether I speak to coworkers or my parents.

This kind of clustering could help exploring this intuitive aspect of human understanding. So far, applications are always domain-specific, which means that they assume everyone using the application understands what is going on. For example, indexing a biomedical paper as in Chapter 2 is specific to a thesaurus. That is, the annotations cannot be generalized easily to make it more understandable to non-expert people. Although our approach does not give a proper answer to this, the underlying matter seems interesting to debate and hierarchical clustering can give good insights for using knowledge representations to generalize/specialize descriptions.

3.11. Chapter summary

This chapter was motivated by previous work in clustering and labeling. Although several approaches tried to make use of semantic data (GO concepts, metadata, concept mapping, etc.) there was no generic approach relying on solid solutions such as semantic similarities.

The studies we presented are thus exploratory and seek to answer or give insight into answers to the following questions. How can we use semantic annotations to cluster documents? Are they sufficient to accomplish this task? Can they be used for labeling the resulting clusters? In order to answer these, the chapter showed an extension of the development in Chapter 2. While type-specific approaches are common in various domains—text mining, video processing, etc.—, we are convinced that semantic similarities can help build more interpretable and meaningful clusters. Actu-

ally, we investigated how they can perform on their own, hence making the approach generic.

As there is no work proposing a benchmark for evaluating hierarchical clustering of semantically annotated documents, we built an interactive interface asking the users to classify bookmarks annotated by concepts of WordNet. This tool allowed us to gather trees of bookmarks, each one being a unique interpretation of a user. By adopting classical phylogenetic tools—manipulating trees is at the core of this field—, we created a benchmark of seven datasets of about 70 bookmarks each. Another dataset can be used for tuning/learning. We evaluated our approach on this benchmark by comparing how different our results were from the expert data. It appears that our method performs better than classical HAC based on a matrix of semantic similarities. Besides, we perform a study of complexity of our algorithm and provide hints in order to optimize it so that it can be as fast as classical HAC. Unfortunately, our results cannot be compared with other methods from the literature because their source code is not available or their method cannot be adapted to this specific benchmark. We thus hope this work may prove useful for future research aiming at making use of semantic annotations for clustering, in trying a semantic similarity, a new clustering algorithm, etc. A perspective to this work is to consider novel metrics for evaluating hierarchical classification as proposed by Kosmopoulos et al. (2015).

We also studied the possibility of labeling a group of documents, i.e. a cluster. Representing a cluster by concepts can be of great use under some conditions: the labels must be specific enough and concise so that the reader quickly gets an idea of the content of this cluster. We extended the USI

framework by adding a constraint to have a control over the genericity of the labels. Depending on the context, the user may want more or less specificity. The context can be represented by the depth of a node in the clustering, for example. We compare our results with two baselines, one naïve and one elaborated, and show that our method performs better than both of them. We also note that it is pretty sensitive to outliers, that human experts tend to group in a single junk cluster. This is an inherent problem of automatic approaches. When outliers are removed or when only considering *clean* clusters, however, the results we get are much more satisfying.

The discussion on the genericity of labels depending on the depth in the tree led us to wonder about a more general matter: complementarity of documents. Even though complementarity is obvious when we think about it (e.g. a toothbrush and toothpaste are complementary), it is difficult to represent it formally. In many cases, being able to assess the complementarity of several items would be useful. For example, for recommending products on a website, where we want to suggest the user items that would fit well together by definition, and not because of a content-based analysis, e.g. because other customers often bought those items together. We explored a way of representing complementarity through a hierarchical clustering. Indeed, observing the tree at different levels may lead to different conclusions. For instance, INFORMATION RETRIEVAL and INDEXING are two different fields, but depending on the context we might group them as ARTIFICIAL INTELLIGENCE. This situation occurs frequently when we talk to someone. If this person works in ARTIFICIAL INTELLIGENCE we will be specific and talk about the two fields separately. However, if we talk to someone who barely knows what those fields are, we will simply refer to them as ARTIFICIAL INTELLIGENCE.

4

Conclusion

Contents

The rather broad title of this thesis is justified by a real wish to investigate a novel way of using documents that are associated with an ontology through semantic annotations. We focused on two wide topics related to (or part of) Information Retrieval, namely indexing and clustering. In fact, the most important motivation of this thesis was to build generic methods to fulfill the tasks needed in these domains.

4.1. On the saliency of knowledge-based systems in IR

In the first chapter, we mentioned that using the Knowledge Representations (KRs), mainly by relying on semantic similarities (see Definition 2), we might be able to bring generic solutions. Indeed, so far, the literature proposes different approaches for different document types, e.g. videos, genes, texts, etc. Even regarding a single document type, say textual, the methods differ depending on the document context, e.g. scientific papers are dealt with differently than novels or patent. It also happens that even for a given document type and context, the content can be related to several topics with different vocabularies (biomedical, chemical, etc.). This means that there are several granularities of specificity and each of them requires specific methods to capture the essence of the documents to annotate them or cluster them. As a result, there are plenty of approaches in the literature that focus on very specific aspects of vocabulary, context and type to provide new indexing or clustering approaches.

The creation and use of such type-specific methods is motivated by the fact that it may perform better than more generic approaches that solely rely on semantic annotations because the latter do not capture as many features

as type-specific methods do (Fiorini et al., 2014c). However, we are convinced that the underlying structure of the knowledge helps to get comparable performances. We tested this idea for indexing first, by building a generic indexing framework called USI (User-oriented Semantic Indexer). Instead of parsing the contents of a document to be annotated, we propose to identify the k-NN (k-Nearest Neighbor) documents in the corpus of already annotated documents. This is done very easily by using an IRS (Information Retrieval System) that looks in the corpus for similar documents in terms of content. These methods are already specific to the type of document and there is an extensive literature regarding them, which means that efficient approaches are available to fulfill this task. Then, we propose to use the nearest neighbors to annotate the target document on the sole basis of concepts annotating the neighbors. To do so, we use a semantic similarity-based objective function that aims at summarizing the set of concepts annotating the neighbors into a smaller set that corresponds to the target document. The quality of annotations provided by our approach is very promising as our annotations are better scored than the literature to which we could compare them. Since other approaches in the literature also consider the neighboring documents, this proves that the use of knowledge truly helps for finding the concepts that are the most relevant regarding a neighborhood of semantically annotated documents.

The use of KRs is often associated with an idea of poor computing performances. Of course, when compared to classical text-based similarities for example, finding the shortest path in a graph seems heavier. Note however that, usually, a conceptual annotation is composed of about 20 concepts at most while textual approaches use high-dimensional vectors to annotate them. Also, a lot of efforts in the Semantic Similarity Measure

(SSM) field helped in creating efficient tools for computing these similarities. That, and the construction of optimized algorithms which use them bring very interesting conclusions in terms of computational complexity. We showed with USI that we can provide a method that is even faster (by about a factor of 50) than Machine Learning ones, which often require a heavy training phase (Fiorini et al., 2015c).

As a result, USI is a successful example of a fast generic indexing approach that relies on semantic similarities. Obviously, this approach also presents a few drawbacks that are generally common to KR-based approaches. First, it requires an adapted domain-ontology which is not always available. Second, it assumes that there exist documents that are already annotated. Finally, the choice of a semantic similarity can be problematic, although we showed that the impact of this choice on USI is minimal.

It follows that semantic similarity-based approaches are more appropriate in some contexts than in others. The most conspicuous one is the biomedical field which is very rich in terms of ontologies. We participated in the BioASQ challenge that featured a real indexing use case. All year long, an NCBI team is dedicated to the semantic annotation of scientific papers. They use a semi-automatic tool to suggest annotations that they curate. This work is extremely time-consuming and any improvement of the quality of annotations that the semi-automatic method outputs is of huge importance for their work. The results of USI at this challenge are outstanding considering the fact that it is a generic approach (Fiorini et al., 2015b). It actually ranks in the top three systems of the challenge and provides real perspectives for improvements. Indeed, USI could be combined with a text-specific approach to get even better results and efficiently contribute to the

work of expert indexers.

The existence of such an indexing team also shows that even with powerful algorithms like the ones designed for the challenge, the need for the human expert remains. We investigated some solutions to help the experts in indexing documents in addition to proposing an annotation instead of replacing them. Including the user in other steps of the process—rather than at the end—happened to produce more relevant results compared to a fully automatic method. Here again, the use of semantic similarities allowed us to build interesting features such as a visual tool that displays the impact of imprecision when the expert has to make a decision, thus limiting the risk of mistakes (Fiorini et al., 2014b). We dedicated a lot of effort in the creation of ergonomic, flexible and easy-to-use interfaces that are typically ready for technology transfer, as the LGI2P laboratory works with several industrial partners.

4.2. On the genericity brought by semantics

As we claimed that USI is fast, generic and efficient, we wanted to extend it to another use case that involves deeper modifications—i.e., not only changing the corpus and the ontology. Clustering is another IR-related task that may be helped by SSMs. We followed with the same idea as with USI, that is, exploring the literature and trying to abstract from the previous methods.

We built a semantic similarity-based clustering approach that uses USI in order to cluster documents and automatically annotate the clusters (Fiorini et al., 2015a). Classical HAC (Hierarchical Agglomerative Clustering)

is based on a *pairwise* similarity matrix of documents. The pairwise similarities are used (averaged, compared, etc) in order to calculate the similarity of two groups. Consequently, by iteratively finding the most similar clusters, creating a new cluster and updating its similarity with the others according to an agglomeration of pairwise similarities, these approaches build a hierarchical representation of clusters. What we propose is to start from a pairwise semantic similarity matrix of documents and update it by using *groupwise* semantic similarities instead. We also label the clusters along with their creation. Indeed, when a new cluster is created, we can identify the semantics—the meaning, the reason—of this creation and use it to label the cluster. The result is a tree that is enriched by concepts associated to each node. Note that we process the resulting tree to keep only the most relevant clusters instead of the very deep binary tree that agglomerative clustering usually outputs. The strategy we detail is thus two-fold and both aspects are evaluated, i.e. (i) the quality of the clusters when using semantic similarities and (ii) the labels associated to each node.

Designing and evaluating this approach has been challenging as well, this time because of the lack of resources in this domain. However, since we are convinced that clustering semantically annotated documents is an important process, we created a benchmark for evaluating it. It is built upon annotated bookmarks extracted from `del.icio.us` used in previous studies. In order to create the benchmark, we developed an interface[1] that provides a clustering tool that experts can use to produce what will be our gold standards. This tool can be adapted to any ontology and any kind of document. It provides an ergonomic workspace where the user can create and manage clusters. Different views are proposed so that we can focus on specific as-

[1] http://clustering.nicolasfiorini.info

pects of the clusters (hierarchy or partitions). Besides, a visual signature of the documents is proposed, based on their semantic annotations. By using this signature, the user can easily compare the documents.

We compiled the expert trees gathered with the tool and created a benchmark[2] composed of eight datasets, one of which should be used for tuning the method. We showed with this benchmark that our method performs better than classical HAC. Besides, the semantic clustering combined with the post-process gives trees that are not much more different from expert trees than the expert trees with each other.

Regarding the cluster labels, the results are satisfying as our method outputs better labels than those of two alternatives we compared it to. However, we observe that our method is sensitive to outliers. These documents are often poorly annotated or have nothing to do with the other ones. While the experts usually group them together in a junk cluster, our method fails to identify them. An improvement of this detection would thus lead to less noisy clusters and better labels. Again the use of semantic similarities could help here, by identifying the documents in the dataset for which the semantic annotation is distant from every other annotation.

The study of hierarchical clustering led us to wonder about some further benefits of the hierarchical structure of the clusters. Although the ideas we detailed are part of a very early reflection, we think that it can be developed in-depth into a research project concerning the assessment of the complementarity of documents according to cluster organization. This kind of approach would certainly be problematic to evaluate but beneficial to a wide range of domains (human resources, study of literature, etc.) and tech-

[2] http://benchmark.nicolasfiorini.info

nologies (recommender systems, information retrieval, etc.). The idea behind it is that with semantic clustering, we are given (i) a structure of documents (ii) associated with semantics. This means that we can propose a hierarchy of documents, for example employees of a company clustered according to their skills. When the company wants to evaluate the relevance of a candidate to hire, an interesting study would be to integrate him/her in the cluster, provided the skills of his/her CV. Maybe the candidate will be branched in a dense cluster, meaning that his/her contribution to the company may be redundant; or maybe he/she will be branched at the intersection of two clusters, meaning that this person would be a key element for the *synergy* of the company.

With a more general scope, we stated two objectives (or raised two questions) in the introduction. The first one concerned the possibility of creating generic methods, the second one dealt with the impact of using semantic similarities. In fact, the latter implies the former, among other things. It is definitely possible to create generic methods, especially by relying on semantic similarities. They bring a lot of consistence in the results that overcomes the lack of features compared to type-specific approaches. On the other hand, using semantic similarities also impacts the development of the approaches. Our experience reveals that since the dawn of KRs, the semantic similarities and other semantic techniques still remain cutting-edge technologies as we face the lack of libraries, of supported languages and most importantly, of benchmarks. There is thus a lot of work to be done on this front, as we are certain that semantic similarity-based systems help in creating generic approaches that can be integrated in larger pipelines at a low cost while providing a huge gain, especially due to the inferences made possible by the use of KRs.

4.3. Perspectives

Some more in-depth studies should be conducted regarding the user experience in some of our applications. For example, we can question how difficult it is to accurately point on a semantic map the location of a document to be annotated. Indeed, we showed that it helps in improving the quality of annotations, however we did not assess how hard it is in a real use case. More generally, we think that working with ergonomists could enhance all the interfaces developed in these projects. Nevertheless, the feedbacks we could gather from users who experienced our tools were mostly positive, which convinces us that these tools are ready for technological transfer with industrial partners such as promoted by the laboratory.

Our work focuses on the use of semantic similarities and shows that great results can be achieved on their sole basis. This experience, certainly motivated by the huge effort in the NLP domain compared to that of SSs in indexing and clustering, reveals many interesting conclusions. SS-based systems can be fast, accurate and generic. Still, we would like to explore the domain of NLP, particularly combined with that of SSs. We think both approaches benefit from each other as they both present a set of advantages that can be exploited in order to build even better indexing and clustering techniques. After all, the whole point of building generic approaches is to be easily plugged in and adapted in more specific workflows.

This PhD thesis represents a good summary of my curriculum, initially in biology and now in computer science. Certainly my motivation has been and continues to be to help biologists and provide them with novel methods as proposed in this thesis and other studies (Fiorini et al., 2014a). Most of all, I think that the creation of a new approach is at least as much im-

portant as its implementation and sharing. This explains why, systematically, our approaches are coupled with demonstrations, interfaces, datasets, results, etc. My future post-doctoral work at the NCBI will hopefully fulfill all these objectives as I hope to be able to have a direct impact on the biomedical community by working on biomedical text-mining and, if applicable, combining such approaches with SS-based ones developed in this thesis.

As an epilogue for this thesis, I would like to cite an inspiring quote of Hal Elrod.

Give up being perfect for being authentic.

A

Appendix

A.1. List of abbreviations

By alphabetical order:

AI . . . . . Artificial Intelligence.

BioASQ . . . . . A challenge on large-scale biomedical semantic indexing and question answering.

BioASQ5000 . . . . . An indexing dataset created by Mao et al. (2014), from the BioASQ 2a task.

bioUSI . . . . . An interface for annotating biomedical papers with USI.

BMA . . . . . Best Match Average, a groupwise semantic similarity measure from Schlicker et al. (2006).

CHC . . . . . Conceptual Hierarchical Clustering, from Spanakis et al. (2011).

DAG . . . . . Directed Acyclic Graph.

GO . . . . . Gene Ontology.

HAC . . . . . Hierarchical Agglomerative Clustering.

HSC . . . . . Heavy Semantic Clustering, a HAC method proposed in chapter 3.

IC . . . . . Information Content, the amount of information a concept conveys.

IR . . . . . Information Retrieval.

IRS . . . . . Information Retrieval System.

k-NN . . . . . k-Nearest Neighbors.

KR . . . . . Knowledge Representation.

L1000 . . . . . An indexing dataset created by Huang et al. (2011).

L2R . . . . . Learning-to-rank, an ML method for ordering a set of items.

LCA-F . . . . . LCA stands for Lowest Common Ancestor. The LCA-F is a hierarchical F-measure.

LGI2P . . . . . *Laboratoire de Génie Informatique et d'Ingénierie de Production*, the laboratory that hosts this research.

LSA . . . . . Latent Semantic Analysis, also called LSI depending on the context.

LSC . . . . . Light Semantic Clustering, a HAC method proposed in chapter 3.

LSI . . . . . Latent Semantic Indexing, also called LSA depending on the context.

MAP . . . . . Mean Average Precision, a metric used to evaluate ordered results.

MDS . . . . . Multi-Dimensional Scaling, a projection method.

MeSH . . . . . Medical Subject Headings, a biomedical thesaurus.

MICA . . . . . Most Informative Common Ancestor, an information theory-based LCA (see LCA-F).

ML . . . . . Machine Learning.

moviesUSI . . . . . An interface for annotating movies with USI.

MTI . . . . . Medical Text Indexer, an indexing approach proposed by Aronson et al. (2004).

MTIFL . . . . . MTI First Line, a system submitted to BioASQ based on MTI.

NCBI . . . . . National Center for Biotechnology Information

NCBO . . . . . National Center for Biomedical Ontology

NLM . . . . . National Library of Medicine.

NLM2007 . . . . . An indexing dataset proposed in Aronson et al. (2004).

NLP . . . . . Natural Language Processing.

PMRA . . . . . PubMed Related Articles, an IRS from Lin and Wilbur (2007).

PMRA* . . . . . A modified version of PMRA that solely considers the textual similarity of two documents to compare.

RI . . . . . Random Indexing, for which a description is available in chapter 2 and Sahlgren (2005).

RRI . . . . . Reflective Random Indexing, an iterative version of RI.

SC . . . . . Semantic Clustering.

SM . . . . . Similarity Measure.

SML . . . . . Semantic Measures Library (Harispe et al., 2014a).

SS . . . . . Semantic Similarity.

SSM . . . . . Semantic Similarity Measure.

SVD . . . . . Singular Value Decomposition, an important process in LSA/LSI for factorizing a matrix.

SVM . . . . . Support Vector Machine, an ML method for binary classification.

USI . . . . . User-Oriented Semantic Indexing, a method detailed in chapter 2.

A.2. List of important mathematical notations

By order of appearance:

D	A set of documents.
θ	An ontology.
$\mathcal{C}$	The set of concepts in a KR.
$\mathcal{H}_C$	The taxonomy organizing $\mathcal{C}$.
$\mathcal{R}$	The set of non-taxonomic relationships of $\mathcal{C}$.
$\preceq$	The binary relation over $\mathcal{C}$ defining the taxonomy $\mathcal{H}_C$.
c	A concept such that $c \in \mathcal{C}$. Some other notations exist throughout the manuscript: c', u, v, w, c_i, etc.
θ_{Tax}	The θ ontology restricted to the taxonomic relationships.
$IC_{name}(c)$	The information content measure for a concept c, as proposed by name.
$desc(c)$	The set containing c and all of its descendants.
$sim_{name}(c, c')$	The semantic similarity measure of two concepts c, c', as proposed by name.
$MICA(c, c')$	The Most Informative Common Ancestor of c and c'.

$\mathrm{anc}(c)$	The set containing c and all of its ancestors.
A, B	Sets of concepts. Note that A is also used to represent the tested solutions of USI (see Chapter 2).
$\mathcal{P}(\mathcal{C})$	Set of partitions of $\mathcal{C}$.
K	The set of k nearest neighbors.
A_d	The annotation associated to document d.
$\mathfrak{A}_K$	The family of sets of annotations of all documents of K.
A_0	The set of concepts used as the search space of USI.
$f(A)$	The objective function of USI over the set of concepts A.
A^*	The optimal solution regarding the objective function $f(A)$.
$\mathrm{sim}_g(A, B)$	The groupwise semantic similarity of two sets of concepts A, B.
$\mathrm{sim}_p(a, b)$	The pairwise semantic similarity of two sets of concepts a, b.
μ	The parameter that controls concision in USI. It controls concision and abstraction in semantic clustering of chapter 3.

$\mathcal{O}$	Big O notation for detailing the complexity of an algorithm.
z	The size of A in USI, so $z = \lvert A \rvert$.
n	The size of A_0 in USI, so $z = \lvert A_0 \rvert$.
$S_{d_{max}}$	The maximal size of an annotation in $\mathfrak{A}_K$.
M_{ps}	The matrix of all pairwise similarities in USI.
$\mathsf{SumMaxCols}(M_{ps})$	The sum of all column maxima in the matrix M_{ps}. For better understanding, we also refer to its value as SumMaxCols.
$\mathsf{sumMaxRows}(M_{ps})$	The sum of all row maxima in the matrix M_{ps}. For better understanding, we also refer to its value as SumMaxCols.
SLINK	The single linkage criterion for HAC.
CLINK	The complete linkage criterion for HAC.
ALINK	The average linkage criterion for HAC.
Cl_i	A cluster.
L_i	The label of the cluster Cl_i.
L_0	The set of concepts used as the search space for the semantic clustering approach.
$g(L)$	The objective function of the semantic clustering algorithm over the set of concepts L.

L^* . . . . . The optimal solution regarding the objective function $g(L)$.

S_{max} . . . . . The maximal size of a cluster label.

Bibliography

Adryan, B. and Schuh, R. (2004). Gene-Ontology-based clustering of gene expression data. *Bioinformatics (Oxford, England)*, 20(16):2851–2.

Agirre, E. and Rigau, G. (1996). Word sense disambiguation using conceptual density. In *Proceedings of the 16th conference on Computational linguistics-Volume 1*, pages 16–22. Association for Computational Linguistics.

Agrawal, R., Gollapudi, S., Halverson, A., and Ieong, S. (2009). Diversifying search results. page 5, New York, New York, USA. ACM Press.

Andrews, P., Pane, J., and Zaihrayeu, I. (2011). Semantic disambiguation in folksonomy: A case study. In Bernardi, R., Chambers, S., Gottfried, B., Segond, F., and Zaihrayeu, I., editors, *Advanced Language Technologies for Digital Libraries*, volume 6699 of *Lecture Notes in Computer Science*, pages 114–134. Springer Berlin Heidelberg.

Aronson, A. R. (2001). Effective mapping of biomedical text to the UMLS Metathesaurus: the MetaMap program. pages 17–21.

Aronson, A. R., Mork, J. G., Gay, C. W., Humphrey, S. M., and Rogers, W. J. (2004). The NLM indexing initiative's medical text indexer. *Medinfo*, 11(Pt 1):268–72.

Baeza-Yates, R. and Ribeiro-Neto, B. (1999). *Modern information retrieval*, volume 9. ACM press, New York.

Baghel, R. and Dhir, D. R. (2010). A Frequent Concepts Based Document Clustering Algorithm. 4(5):6–12.

Balikas, G., Kosmopoulos, A., Krithara, A., and Kakadiaris, I. (2015). Results of the BioASQ tasks of the Question Answering Lab at CLEF 2015. In *Working Notes for the Conference and Labs of the Evaluation Forum (CLEF), Toulouse, France*.

Bauer, S., Grossmann, S., Vingron, M., and Robinson, P. N. (2008). Ontologizer 2.0–a multifunctional tool for GO term enrichment analysis and data exploration. *Bioinformatics (Oxford, England)*, 24(14):1650–1.

Beissbarth, T. and Speed, T. P. (2004). GOstat: find statistically overrepresented Gene Ontologies within a group of genes. *Bioinformatics (Oxford, England)*, 20(9):1464–5.

Bellman, R. E. (2003). *Dynamic Programming*. Dover Publications, Incorporated.

Ben Abacha, A. and Zweigenbaum, P. (2015). Means: A medical question-answering system combining nlp techniques and semantic web technologies. *Information Processing & Management*, 51(5):570–594.

Bernardini, A., Carpineto, C., and D'Amico, M. (2009). Full-Subtopic Retrieval with Keyphrase-Based Search Results Clustering. *2009*

IEEE/WIC/ACM International Joint Conference on Web Intelligence and Intelligent Agent Technology, pages 206–213.

Berners-Lee, T., Hendler, J., Lassila, O., et al. (2001). The semantic web. *Scientific american*, 284(5):28–37.

Bhagdev, R., Chapman, S., Ciravegna, F., Lanfranchi, V., and Petrelli, D. (2008). Hybrid Search: Effectively Combining Keywords and Semantic Searches. *The Semantic Web Research and Applications*, 5021(ii):554–568.

Bharathi, G. and Venkatesan, D. (2012). Study of ontology or thesaurus based document clustering and information retrieval. *Journal of Engineering and Applied Sciences*, 7(4):342–347.

Breaux, T. D. and Reed, J. W. (2005). Using ontology in hierarchical information clustering. In *System Sciences, 2005. HICSS'05. Proceedings of the 38th Annual Hawaii International Conference on*. IEEE.

Cao, Z., Qin, T., Liu, T.-Y., Tsai, M.-F., and Li, H. (2007). Learning to rank: from pairwise approach to listwise approach. In *Proceedings of the 24th international conference on Machine learning*, pages 129–136. ACM.

Carmel, D., Roitman, H., and Zwerdling, N. (2009). Enhancing cluster labeling using wikipedia. In *Proceedings of the 32nd international ACM SIGIR conference on Research and development in information retrieval - SIGIR '09*, page 139, New York, New York, USA. ACM Press.

Carmel, D., Yom-Tov, E., Darlow, A., and Pelleg, D. (2006). What makes a query difficult? In *Proceedings of the 29th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '06, pages 390–397, New York, NY, USA. ACM.

Carson, C., Thomas, M., Belongie, S., Hellerstein, J. M., and Malik, J. (1999). Blobworld: A system for region-based image indexing and retrieval. In *Visual Information and Information Systems*, pages 509–517. Springer.

Chebel, M., Latiri, C., and Gaussier, E. (2015). Multilingual documents clustering based on closed concepts mining. In Chen, Q., Hameurlain, A., Toumani, F., Wagner, R., and Decker, H., editors, *Database and Expert Systems Applications*, volume 9261 of *Lecture Notes in Computer Science*, pages 517–524. Springer International Publishing.

Clarke, C. L. A., Craswell, N., Soboroff, I., , and Ashkan, A. (2011). A Comparative Analysis of Cascade Measures for Novelty and Diversity. *Proceedings of the fourth ....*

Clerkin, P., Cunningham, P., and Hayes, C. (2001). Ontology discovery for the semantic web using hierarchical clustering. *Semantic Web Mining*, page 27.

Cohen, T., Schvaneveldt, R., and Widdows, D. (2010). Reflective random indexing and indirect inference: A scalable method for discovery of implicit connections. *Journal of biomedical informatics*, 43(2):240–256.

Cormen, T. H. (2009). *Introduction to algorithms*. MIT press.

Crampes, M. and Plantié, M. (2014). A unified community detection, visualization and analysis method. *Advances in complex systems*, 17(1):1450001.

Delbecque, T. and Zweigenbaum, P. (2010). Using Co-Authoring and Cross-Referencing Information for MEDLINE Indexing. *AMIA Annual Symposium Proceedings*, 2010:147.

Fiorini, N., Harispe, S., Ranwez, S., Montmain, J., and Ranwez, V. (2015a). Annotation sémantique de clusters. In *16e conférence Roadef*.

Fiorini, N., Lefort, V., Chevenet, F., Berry, V., and Chifolleau, A.-M. (2014a). CompPhy: a web-based collaborative platform for comparing phylogenies. *BMC Evolutionary Biology*, 14(1):253.

Fiorini, N., Ranwez, S., Harispe, S., Montmain, J., and Ranwez, V. (2015b). Usi at bioasq 2015: a semantic similarity-based approach for semantic indexing. In *Working Notes for the Conference and Labs of the Evaluation Forum (CLEF), Toulouse, France*.

Fiorini, N., Ranwez, S., Montmain, J., and Ranwez, V. (2014b). Coping with imprecision during a semi-automatic conceptual indexing process. In *Information Processing and Management of Uncertainty in Knowledge-Based Systems*, pages 11–20. Springer.

Fiorini, N., Ranwez, S., Montmain, J., and Ranwez, V. (2015c). USI: a fast and accurate approach for conceptual document annotation. *BMC Bioinformatics*, 16(1):1–10.

Fiorini, N., Ranwez, S., Ranwez, V., and Montmain, J. (2014c). Indexation conceptuelle par propagation. application à un corpus d'articles scientifiques liés au cancer. In *CORIA 2014, COnférence en Recherche d'Information et Applications*, page 187.

Fisher, D. (1987). Knowledge acquisition via incremental clustering. *Machine Learning*, 2(1980):139–182.

Fouss, F., Pirotte, A., Renders, J.-M., and Saerens, M. (2007). Random-walk computation of similarities between nodes of a graph with applica-

tion to collaborative recommendation. *Knowledge and data engineering, ieee transactions on*, 19(3):355–369.

Gay, C. W., Kayaalp, M., and Aronson, A. R. (2005). Semi-automatic indexing of full text biomedical articles. In *AMIA Annual Symposium Proceedings*, volume 2005, page 271. American Medical Informatics Association.

Gentleman, R. (2010). Visualizing and distances using go.

Geraci, F., Pellegrini, M., Maggini, M., and Sebastiani, F. (2006). Cluster Generation and Labeling for Web Snippets: A Fast, Accurate Hierarchical Solution.

Giunchiglia, F., Kharkevich, U., and Zaihrayeu, I. (2009). Concept search. In *The Semantic Web: Research and Applications*, pages 429–444. Springer.

Gollapudi, S. and Sharma, A. (2009). An axiomatic approach for result diversification. In *Proceedings of the 18th international conference on World wide web*, pages 381–390. ACM.

Haav, H.-M. and Lubi, T.-L. (2001). A survey of concept-based information retrieval tools on the web.

Harispe, S. (2014). *Knowledge-based Semantic Measures: From Theory to Applications*. PhD thesis in computer science, Université de Montpellier.

Harispe, S., Imoussaten, A., Trousset, F., and Montmain, J. (2015a). On the consideration of a bring-to-mind model for computing the information content of concepts defined into ontologies. In *Proceedings of FUZZ-IEEE 2015*.

Harispe, S., Ranwez, S., Janaqi, S., and Montmain, J. (2014a). The seman-

tic measures library and toolkit: fast computation of semantic similarity and relatedness using biomedical ontologies. *Bioinformatics*, 30(5):740–2.

Harispe, S., Ranwez, S., Janaqi, S., and Montmain, J. (2015b). *Semantic Similarity from Natural Language and Ontology Analysis*, volume 8. Morgan & Claypool Publishers.

Harispe, S., Sánchez, D., Ranwez, S., Janaqi, S., and Montmain, J. (2014b). A framework for unifying ontology-based semantic similarity measures: A study in the biomedical domain. *Journal of Biomedical Informatics*, 48:38–53.

Hecht-Nielsen, R. (1994). Context vectors: general purpose approximate meaning representations self-organized from raw data. *Computational intelligence: Imitating life*, pages 43–56.

Holyoak, K. J. and Koh, K. (1987). Surface and structural similarity in analogical transfer. *Memory & Cognition*, 15(4):332–340.

Hotho, A., Maedche, A., and Staab, S. (2001). Text clustering based on good aggregations. In *Data Mining, 2001. ICDM 2001, Proceedings IEEE International Conference on*, pages 607–608. IEEE.

Hotho, A., Maedche, A., and Staab, S. (2002). Ontology-based text document clustering. *KI*, pages 1–13.

Hotho, A., Staab, S., and Stumme, G. (2003). Wordnet improves Text Document Clustering. *Data Mining, 2003. ICDM 2003. Third IEEE International Conference on*, 03:541–544.

Huang, M., Névéol, A., and Lu, Z. (2011). Recommending MeSH terms

for annotating biomedical articles. *Journal of the American Medical Informatics Association*, 18(5):660–667.

Jimeno-Yepes, A., Mork, J. G., Demner-Fushman, D., and Aronson, A. R. (2012). A One-Size-Fits-All Indexing Method Does Not Exist: Automatic Selection Based on Meta-Learning. *Journal of Computing Science and Engineering*, 6(2):151–160.

Jimeno Yepes, A., Mork, J. G., Wilkowski, B., Demner Fushman, D., and Aronson, A. R. (2012). Medline mesh indexing: lessons learned from machine learning and future directions. In *Proceedings of the 2nd ACM SIGHIT International Health Informatics Symposium*, pages 737–742. ACM.

Jonquet, C., Shah, N. H., and Musen, M. A. (2009). The open biomedical annotator. *Summit on translational bioinformatics*, 2009:56.

Kosmopoulos, A., Partalas, I., Gaussier, E., Paliouras, G., and Androutsopoulos, I. (2013). Evaluation measures for hierarchical classification: a unified view and novel approaches. *Data Mining and Knowledge Discovery*, 29(3):820–865.

Kosmopoulos, A., Partalas, I., Gaussier, E., Paliouras, G., and Androutsopoulos, I. (2015). Evaluation measures for hierarchical classification: a unified view and novel approaches. *Data Mining and Knowledge Discovery*, 29(3):820–865.

Kuhn, A., Ducasse, S., and Gîrba, T. (2007). Semantic clustering: Identifying topics in source code. *Information and Software Technology*, 49:230–243.

Lancaster, F. W. and Gallup, E. (1973). Information retrieval on-line. Technical report.

Lee, H. K., Braynen, W., Keshav, K., and Pavlidis, P. (2005). ErmineJ: tool for functional analysis of gene expression data sets. *BMC bioinformatics*, 6:269.

Li, Y., Bandar, Z., McLean, D., et al. (2003). An approach for measuring semantic similarity between words using multiple information sources. *Knowledge and Data Engineering, IEEE Transactions on*, 15(4):871–882.

Lin, D. (1998). An information-theoretic definition of similarity. In *ICML*, volume 98, pages 296–304.

Lin, J. and Wilbur, W. J. (2007). PubMed related articles: a probabilistic topic-based model for content similarity. *BMC bioinformatics*, 8(1):423.

Liu, K., Wu, J., Peng, S., Zhai, C., and Zhu, S. (2014). The fudan-uiuc participation in the bioasq challenge task 2a: The antinomyra system. *Risk*, 129816:100.

Liu, X., Croft, W. B., Oh, P., and Hart, D. (2004). Automatic recognition of reading levels from user queries. In *Proceedings of the 27th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 548–549. ACM.

Lloyd, S. P. (1982). Least squares quantization in pcm. *IEEE Transactions on Information Theory*, 28(2):129–137.

Lula, P. and Paliwoda-Pękosz, G. (2008). An ontology-based cluster analysis framework. In *Proceedings of the first international workshop on Ontology-supported business intelligence*, page 7. ACM.

Maedche, A. and Staab, S. (2001). Ontology learning for the Semantic Web. *IEEE Intelligent Systems*, 16(2):72–79.

Maedche, A. and Zacharias, V. (2002). Clustering Ontology-Based Metadata in the Semantic Web. In *Principles of Data Mining and Knowledge Discovery*, volume 2431, pages 383–408.

Maere, S., Heymans, K., and Kuiper, M. (2005). BiNGO: a Cytoscape plugin to assess overrepresentation of gene ontology categories in biological networks. *Bioinformatics (Oxford, England)*, 21(16):3448–9.

Manning, C. D., Raghavan, P., Schütze, H., et al. (2008). *Introduction to information retrieval*, volume 1. Cambridge university press.

Mao, Y. and Lu, Z. (2013). NCBI at the 2013 BioASQ challenge task: Learning to rank for automatic MeSH indexing. *Technical report*.

Mao, Y., Wei, C.-h., and Lu, Z. (2014). NCBI at the 2014 BioASQ challenge task: large-scale biomedical semantic indexing and question answering. *CLEF 2014 Working Notes Proceedings*, pages 1319–1327.

Maron, M. E. and Kuhns, J. L. (1960). On relevance, probabilistic indexing and information retrieval. *Journal of the ACM (JACM)*, 7(3):216–244.

Mi, H., Dong, Q., Muruganujan, A., Gaudet, P., Lewis, S., and Thomas, P. D. (2010). PANTHER version 7: improved phylogenetic trees, orthologs and collaboration with the Gene Ontology Consortium. *Nucleic acids research*, 38(Database issue):D204–10.

Mistry, M. and Pavlidis, P. (2008). Gene ontology term overlap as a measure of gene functional similarity. *BMC bioinformatics*, 9(1):327.

Névéol, A. and Shooshan, S. (2009). A recent advance in the automatic indexing of the biomedical literature. *Journal of Biomedical Informatics*, 42(5):814–823.

Névéol, A., Zeng, K., and Bodenreider, O. (2006). Besides precision & recall: Exploring alternative approaches to evaluating an automatic indexing tool for medline. In *AMIA Annual Symposium Proceedings*, volume 2006, page 589. American Medical Informatics Association.

Neves, M. and Leser, U. (2014). A survey on annotation tools for the biomedical literature. *Briefings in bioinformatics*, 15(2):327–40.

Ovaska, K., Laakso, M., and Hautaniemi, S. (2008). Fast gene ontology based clustering for microarray experiments. *BioData mining*, 1(1):11.

Papanikolaou, Y., Dimitriadis, D., Tsoumakas, G., Laliotis, M., Markantonatos, N., and Vlahavas, I. (2014). Ensemble approaches for large-scale multi-label classification and question answering in biomedicine. In *2nd BioASQ Workshop: A challenge on large-scale biomedical semantic indexing and question answering*.

Paquette, J. and Tokuyasu, T. (2010). EGAN: exploratory gene association networks. *Bioinformatics (Oxford, England)*, 26(2):285–6.

Pattengale, N. D., Gottlieb, E. J., and Moret, B. M. (2007). Efficiently computing the robinson-foulds metric. *Journal of Computational Biology*, 14(6):724–735.

Pedersen, T., Pakhomov, S. V., Patwardhan, S., and Chute, C. G. (2007). Measures of semantic similarity and relatedness in the biomedical domain. *Journal of biomedical informatics*, 40(3):288–299.

Pesquita, C., Faria, D., Bastos, H., Falcão, A., and Couto, F. (2007). Evaluating go-based semantic similarity measures. In *Proc. 10th Annual Bio-Ontologies Meeting*, volume 37, page 38.

Pich, C. (2009). Mdsj: Java library for multidimensional scaling (version 0.2). 2009.

Rada, R., Mili, H., Bicknell, E., and Blettner, M. (1989). Development and application of a metric on semantic nets. *Systems, Man and Cybernetics, IEEE Transactions on*, 19(1):17–30.

Ranwez, S., Duthil, B., Sy, M. F., Montmain, J., Augereau, P., and Ranwez, V. (2013). How ontology based information retrieval systems may benefit from lexical text analysis. In *New Trends of Research in Ontologies and Lexical Resources*, pages 209–231. Springer.

Ranwez, V. and Gascuel, O. (2002). Improvement of distance-based phylogenetic methods by a local maximum likelihood approach using triplets. *Molecular Biology and Evolution*, 19(11):1952–1963.

Reimand, J., Kull, M., Peterson, H., Hansen, J., and Vilo, J. (2007). g:Profiler–a web-based toolset for functional profiling of gene lists from large-scale experiments. *Nucleic acids research*, 35(Web Server issue):W193–200.

Resnik, P. (1995). Using information content to evaluate semantic similarity in a taxonomy. In *Proceedings of IJCAI-95*, pages 448–453.

Resnik, P. (1999). Semantic Similarity in a Taxonomy: An Information-Based Measure and its Application to Problems of Ambiguity in Natural Language. *Journal of Artificial Intelligence Research*, 11.

Robertson, S. E., Walker, S., Jones, S., Hancock-Beaulieu, M. M., Gatford, M., et al. (1995). Okapi at trec-3. *NIST SPECIAL PUBLICATION SP*, pages 109–109.

Robinson, D. and Foulds, L. R. (1981). Comparison of phylogenetic trees. *Mathematical Biosciences*, 53(1):131–147.

Role, F. and Nadif, M. (2014). Beyond cluster labeling: Semantic interpretation of clusters' contents using a graph representation. *Knowledge-Based Systems*, 56:141–155.

Rossi, R. G., Marcacini, R. M., and Rezende, S. O. (2013). Benchmarking text collections for classification and clustering tasks. *Institute of Mathematics and Computer Sciences, University of Sao Paulo*.

Russell, S. and Norvig, P. (1995). *Artificial intelligence: a modern approach*. Prentice Hall.

Sahlgren, M. (2005). An introduction to random indexing. In *Methods and applications of semantic indexing workshop at the 7th international conference on terminology and knowledge engineering, TKE*, volume 5.

Saitou, N. and Nei, M. (1987). The neighbor-joining method: a new method for reconstructing phylogenetic trees. *Molecular biology and evolution*, 4(4):406–425.

Salton, G., Wong, A., and Yang, C.-S. (1975). A vector space model for automatic indexing. *Communications of the ACM*, 18(11):613–620.

Sánchez, D. and Batet, M. (2011). Semantic similarity estimation in the biomedical domain: An ontology-based information-theoretic perspective. *Journal of Biomedical Informatics*, 44(5):749–759.

Sayers, E. W., Barrett, T., Benson, D. A., Bolton, E., Bryant, S. H., Canese, K., Chetvernin, V., Church, D. M., DiCuccio, M., Federhen, S.,

et al. (2011). Database resources of the national center for biotechnology information. *Nucleic acids research*, 39(suppl 1):D38–D51.

Schlicker, A., Domingues, F. S., Rahnenführer, J., and Lengauer, T. (2006). A new measure for functional similarity of gene products based on Gene Ontology. *BMC bioinformatics*, 7(1):302.

Seco, N., Veale, T., and Hayes, J. (2004). An intrinsic information content metric for semantic similarity in WordNet. *ECAI*, 16:1089.

Sedding, J. and Kazakov, D. (2004). Wordnet-based text document clustering. In *proceedings of the 3rd workshop on robust methods in analysis of natural language data*, pages 104–113. Association for Computational Linguistics.

Shah, N. H. and Fedoroff, N. V. (2004). CLENCH: a program for calculating Cluster ENriCHment using the Gene Ontology. *Bioinformatics (Oxford, England)*, 20(7):1196–7.

Shannon, C. E. (1948). A mathematical theory of communication. *Bell System Technical Journal*, 27(3):379–423.

Shehata, S., Karray, F., and Kamel, M. (2006). Enhancing Text Clustering Using Concept-based Mining Model. *Sixth International Conference on Data Mining (ICDM'06)*, pages 1043–1048.

Skoutas, D., Minack, E., and Nejdl, W. (2010). Increasing Diversity in Web Search Results. pages 1–5.

Smith, B., Ashburner, M., Rosse, C., Bard, J., Bug, W., Ceusters, W., Goldberg, L. J., Eilbeck, K., Ireland, A., Mungall, C. J., et al. (2007). The obo foundry: coordinated evolution of ontologies to support biomedical data integration. *Nature biotechnology*, 25(11):1251–1255.

Song, W., Li, C. H., and Park, S. C. (2009). Genetic algorithm for text clustering using ontology and evaluating the validity of various semantic similarity measures. *Expert Systems with Applications*, 36(5):9095–9104.

Spanakis, G., Siolas, G., and Stafylopatis, a. (2011). Exploiting Wikipedia Knowledge for Conceptual Hierarchical Clustering of Documents. *The Computer Journal*, 55(3):299–312.

Sparrow, B., Liu, J., and Wegner, D. M. (2011). Google effects on memory: Cognitive consequences of having information at our fingertips. *Science*, 333(6043):776–778.

Spellman, P. T., Sherlock, G., Zhang, M. Q., Iyer, V. R., Anders, K., Eisen, M. B., Brown, P. O., Botstein, D., and Futcher, B. (1998). Comprehensive identification of cell cycle–regulated genes of the yeast saccharomyces cerevisiae by microarray hybridization. *Molecular biology of the cell*, 9(12):3273–3297.

Staab, S. and Studer, R. (2013). *Handbook on ontologies*. Springer Science & Business Media.

Stokoe, C., Oakes, M. P., and Tait, J. (2003). Word sense disambiguation in information retrieval revisited. In *Proceedings of the 26th annual international ACM SIGIR conference on Research and development in informaion retrieval*, pages 159–166. ACM.

Sy, M.-F., Ranwez, S., Montmain, J., Regnault, A., Crampes, M., and Ranwez, V. (2012). User centered and ontology based information retrieval system for life sciences. *BMC bioinformatics*, 13(Suppl 1):S4.

Trieschnigg, D., Pezik, P., Lee, V., de Jong, F., Kraaij, W., and Rebholz-

Schuhmann, D. (2009). MeSH Up: effective MeSH text classification for improved document retrieval. *Bioinformatics*, 25(11):1412–1418.

Tseng, V. S., Su, J.-H., Huang, J.-H., and Chen, C.-J. (2008). Integrated mining of visual features, speech features, and frequent patterns for semantic video annotation. *IEEE Transactions on Multimedia*, 10(2):260–267.

Turnbull, D. and Barrington, L. (2008). Semantic annotation and retrieval of music and sound effects. *Semantic annotation and retrieval of music and sound effects. Audio, Speech, and Language Processing*, 16(2):467–476.

Tversky, A. (1977). Features of similarity. *Psychological review*, 84(4):327–352.

Vasuki, V. and Cohen, T. (2010). Reflective random indexing for semi-automatic indexing of the biomedical literature. *Journal of biomedical informatics*, 43(5):694–700.

Wetzker, R., Zimmermann, C., and Bauckhage, C. (2008). Analyzing social bookmarking systems: A del. icio. us cookbook. In *Proceedings of the ECAI 2008 Mining Social Data Workshop*, pages 26–30.

Yang, Y. (1999). An evaluation of Statistical Approaches to Text Categorization. *Information retrieval*, 1(1-2):69–90.

Yoo, I. and Hu, X. (2006). Scale-Free Network based Clustering using Knowledge-enriched Graph Representation of Biomedical Documents. *Knowledge Creation Diffusion Utilization*.

Zhang, S., Tian, Q., Hua, G., Huang, Q., and Gao, W. (2014). ObjectPatchNet: Towards scalable and semantic image annotation and retrieval. *Computer Vision and Image Understanding*, 118:16–29.

Zhou, X., Zhang, X., and Hu, X. (2006a). Maxmatcher: Biological concept extraction using approximate dictionary lookup. In *PRICAI 2006: Trends in Artificial Intelligence*, pages 1145–1149. Springer.

Zhou, X., Zhang, X., and Hu, X. (2006b). Using concept-based indexing to improve language modeling approach to genomic IR. In *Advances in Information Retrieval*, pages 444–455. Springer.

Zhou, Z., Wang, Y., and Gu, J. (2008). A new model of information content for semantic similarity in wordnet. In *Future Generation Communication and Networking Symposia, 2008. FGCNS'08. Second International Conference on*, volume 3, pages 85–89. IEEE.

Pour exploiter efficacement une masse toujours croissante de documents électroniques, une branche de l'Intelligence Artificielle s'est focalisée sur la création et l'utilisation de **systèmes à base de connaissance**. Ces approches ont prouvé leur efficacité, notamment en recherche d'information. Cependant elles imposent une indexation sémantique des ressources exploitées, i.e. que soit associé à chaque ressource un ensemble de termes qui caractérise son contenu. Ces termes peuvent être remplacés par des concepts issus d'une ontologie de domaine, on parle alors d'indexation conceptuelle. Ceci permet non seulement de s'affranchir de toute ambiguïté liée au langage naturel, mais également d'exploiter les liens existants entre ces concepts. Le plus souvent cette indexation est réalisée en procédant à l'extraction des concepts du contenu même des documents. On note, dans ce cas, une forte dépendance des méthodes d'indexation au type de document considéré. Pourtant une des forces des **approches conceptuelles** réside dans leur généricité. En effet, par l'exploitation d'indexation sémantique, ces approches permettent de traiter de la même manière un ensemble d'images, de gènes, de textes ou de personnes, pour peu que ceux-ci aient été correctement indexés. Les travaux de cette thèse proposent des solutions génériques pour indexer sémantiquement des documents ou des groupes de documents.

Deux axes de recherche sont suivis dans cette thèse. Le premier est celui de **l'indexation sémantique**. L'approche proposée exploite l'indexation de documents proches en contenu pour annoter un document cible. Grâce à l'utilisation de similarités sémantiques entre les annotations des documents proches et d'une heuristique efficace, notre approche, USI (User-oriented Semantic Indexer), permet d'annoter des documents plus rapidement que les méthodes existantes tout en assurant une qualité d'indexation comparable. Une attention particulière a été portée dans ces travaux à l'interaction homme-machine et une **approche interactive** prenant en compte l'impact d'une imprécision humaine a également été proposée. Le second axe de cette thèse concerne la **classification de documents** en fonction de leurs contenus. Là encore, la méthode est indépendante du type des documents considérés puisqu'ils sont regroupés sur la base de leurs annotations sémantiques. Un autre avantage de cette approche est que les groupes formés sont automatiquement annotés sémantiquement par notre algorithme.

L'ensemble des développements de cette thèse ont fait l'objet d'un soin particulier concernant leur **optimisation algorithmique** afin de permettre un passage à l'échelle, leur validation sur des benchmarks existants ou construits spécifiquement et leur mise à disposition pour des développeurs (via des librairies java) et des utilisateurs finaux (via des serveurs Web). Nos travaux ont montré que l'utilisation d'ontologies permet d'abstraire plusieurs processus et ainsi de proposer des approches génériques sans dégrader les performances. Cette généricité n'empêche en aucun cas d'être couplée à des approches plus spécifiques, mais constitue en soi une simplicité de mise en place dès lors que l'on dispose de documents annotés sémantiquement.

In order to improve the search and use of documents, Artificial Intelligence has dedicated a lot of effort to the creation and use of knowledge bases such as ontologies. They are graphs in which nodes represent a meaning unit–a concept– and edges are their relationships. For example, this allows to represent the concept “dog” as a subclass of the concept “mammal”. Indexing documents is a useful process for further processing and consists in associating them with sets of terms that describe them. These terms can be concepts from an ontology, in which case the annotation is said to be **semantic**. Such annotations benefit from the inherent properties of ontologies: the absence of synonymy and polysemy. Most approaches designed to annotate documents have to read them and extract concepts from this reading. This means that the approach is dependent from the type of documents, as a text would not be processed the same way a picture or a gene would be. Approaches that solely rely on semantic annotations can ignore the document type, leading to **generic processes**. This has been proved in Information Retrieval where researchers experienced approaches called semantic information retrieval that can fit any type of document.

This thesis capitalizes on genericity accessible through **semantic annotations** to build novel systems and compare them to state-of-the-art approaches. To this end, we rely on semantic annotations coupled with semantic similarity measures. Of course, such generic approach can then be enriched with type-specific ones, which would increase the quality of the results. This work explores the relevance of this paradigm for indexing documents. The idea is to rely on already annotated close documents to annotate a target document. We defined a heuristic algorithm for this purpose that uses the semantic annotations of these close documents and semantic similarities to provide a generic indexing method. This resulted in USI (User-oriented Semantic Indexer) that we showed to perform as well as best current systems while being faster. This idea has been extended to another task, **clustering**. Clustering is a very common process that is useful for finding documents or understanding a set of documents. We propose a hierarchical clustering algorithm that reuses the same components of classical methods to provide a novel one applicable to any kind of documents. Another benefit of this approach is that when documents are grouped together, the group is annotated by using our indexing algorithm. Therefore, the result is not only a hierarchy of clusters containing documents as clusters are actually described by concepts as well. This helps a lot to better understand the result of the clustering. A particular attention has been devoted in this work to **algorithmic optimization** and user-friendliness, with **interactive human-machine interfaces**, that take into account imprecision of human actions.

This thesis shows that apart from improving the results of classical approaches, building conceptual approaches allows us to abstract them and provide a generic framework. Yet, while bringing **easy to setup methods**–as long as documents are semantically annotated–, genericity does not prevent us from mixing these methods with type-specific ones, in other words creating hybrid methods.