M1Info

UNIVERSITE PARIS 8 Vincennes - Saint-Denis

Université Paris 8 - Vincennes à Saint-Denis

Master Informatique des Systèmes Embarqués

Compréhension automatique du langage naturel et représentation des connaissances (IA)

Aghilas NAIT MESSAOUD

Amel ATEK

Date de soutenance : le 19/01/2018

Tuteur – Université : Anna PAPPA

Résumé

Nous présentons l'approche de modélisation, formelle et concrète, pour représenter et manipuler des connaissances d'un domaine. Le modèle des graphes conceptuels permet de modéliser des connaissances en termes de graphes, basés sur un support. Il possède une sémantique issue des logiques de descriptions.

Abstract

We present the modeling approach, formal and concrete, to represent and manipulate knowledge of a domain. The conceptual graphs model makes it possible to model knowledge in terms of graphs, based on a support. It has a semantics derived from the logic of descriptions.

Remerciements

Nous tenons à remercier Madame Anna PAPPA pour son encadrement, ses précieux conseils et sa disponibilité, ainsi que les juristes pour leurs disponibilités et pour le temps qui nous a été accorder.
Un grand merci également pour les différentes personnes qui nous ont orientés dans nos recherches.

Table des matières

Table des figures

plan :

- Introduction
- l'état de l'art.
  - Représentation des connaissances
- conception
  - Réalisation de graphe conceptuel
- Conclusion

Introduction

Dans le cadre du Master 1 Informatique, nous avons été amenés à réaliser un projet tuteuré sur un sujet proposé à l'aide d'un encadrant. parmi les projet proposé notre choix s'est porté sur «Compréhension automatique du langage naturel et représentation des connaissances (IA)», notre tuteur est Mme. Anna pappa. qui nous a accompagné au cours de la réalisation du projet dont vous lisez le rapport.

Il existe plusieurs formalismes pour la représentation de connaissances et l'un de ces formalismes c'est les graphes conceptuels.

la problématique a laquelle nous avons tenter de rependre comment réaliser un graphes conceptuel a partir d'un text quelconque trouver sur internet.

afin de rependre a ce problème :

— En premier lieu nous avons expliquer ce que c'est que la représentation des connaissances et les formalismes qui représentent ces connaissances ainsi que leur manières de fonctionné pour mieux comprendre ce qu'il faut faire a la suite.
— En deuxièmes lieux nous avons expliquer comment récupérer les textes de l'Internet, et réaliser un graphe conceptuel en utilisant le TF-IDF pour récupérer les mots clé et les plus répété.

et enfin nous avons conclue notre rapport avec un exemple réaliser sur un texte de lois.

Première partie

L'état de l'art

Cette partie vise à définir le modèle de représentation de connaissances étudié, le modèle des graphes conceptuels. nous donnerons les principales définitions du modèle ainsi que la méthode d'interrogation du modèle.

Chapitre 1

représentation des connaissances

La *représentation des connaissances* est une étape très importante elle répond d'une manière structuré a la façons dont les connaissances sont représentées et explique d'une manière précise comment les réseaux sémantique et les graphes conceptuels, ce qui vas idée a comprendre au mieux la façons dont le projet sera réaliser.

Il importe, d'entrée, d'établir la distinction entre les concepts d'information et de connaissance. Par « information », nous entendons toutes les données extérieures aux personnes, communiquées oralement par d'autres ou médiatisées dans des matériels sur divers soutiens numériques, imprimés ou analogiques.

Par « connaissance », nous entendons le résultat de toute construction mentale effectuée par un individu à partir d'informations ou d'autres stimuli.

L'apprentissage par un individu consiste à transformer des informations en connaissances.

1.1 définition des connaissances :

On définit la connaissance ou les connaissances comme ce qu'on a appris par l'étude ou par la pratique.

Dans le but de résoudre des problèmes complexes qui relèvent de l'intelligence artificielle, il faut un bon bagage de connaissances et des outils de manipulation de ces connaissances. Les connaissances concernent des faits, considérés vrais dans un certain monde. Pour représenter ces faits on a recours à un formalisme ou mode de représentation. Au niveau des faits, on traitera des objets et des relations qu'ils entretiennent.

Ces connaissances peuvent être modéliser soit par réseaux sémantique soit par graphe conceptuel.

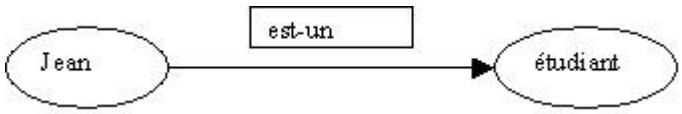
1.2 Réseaux sémantiques :

Un réseau sémantique se présentent comme des ensembles de points ou nœuds étiquetés, chaque nœud représente un concept, objet, une notion générale de logique, etc.. . . les nœuds sont relies par des arcs orientés figurant les relations sémantique qui existant ente les nœuds, les liaisons entre deux nœuds étiquetés par A et B ont la priorité et être en relation par R : R (A, B) qui est présenter sous le nom prédicat binaire. Les arcs du graphe représentent alors des relations entre ces concepts.

Un concept est relié au réseau ou à la famille à laquelle il appartient par le lien : ISA et AKO. Le ISA est une relation qui décrit le fait qu'un concept est considéré comme une « instance » d'une famille d'objets. Correspond à l'appartenance en théorie des ensembles. L'OKA est une classe qui décrit le fait que le réseau considéré est un sous-réseau où encore qu'il fasse partie de la famille d'un réseau donner. Correspond à l'inclusion en théorie des ensembles.

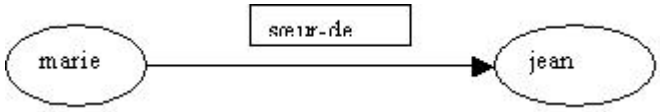
1.2.1 Structure générale des réseaux sémantiques :

Tout d'abord, les réseaux sémantiques peuvent représenter des objets individuels, des catégories d'objets, et des relations entre objets ou catégories. Les objets sont représentés dans des boîtes tandis que les relations étiquettent les nœuds qui relient ces objets :



Ceci correspond à l'assertion en logique à l'application d'un prédicat sur un terme : Étudiant(jean) ou bien, en théorie des ensembles : jean étudiant.

De la même manière, l'instance de la relation (sœur – de(X,Y),sœur – de(marie, jean)), s'écrit en réseaux sémantiques par :



On peut alors connecter toutes sortes d'objets par les différentes relations à disposition. Cependant, les réseaux doivent être conçus de façon cohérente. Par exemple, nous savons que les humains ont des hommes (= personne-masculin) comme pères. Il n'est pas possible d'établir un lien direct entre personne et personne-masculin parce que les catégories n'ont pas de pères. Pour ce faire, il faut introduire une notation complémentaire, la double boite, qui permet de coder des énoncés du type :

$$\forall X, X \in personne \implies [\forall Y, a - pere(X, Y) \implies Y \in personne - masculin$$

Cet énoncé s'insère graphiquement dans un réseau sémantique comme suit :

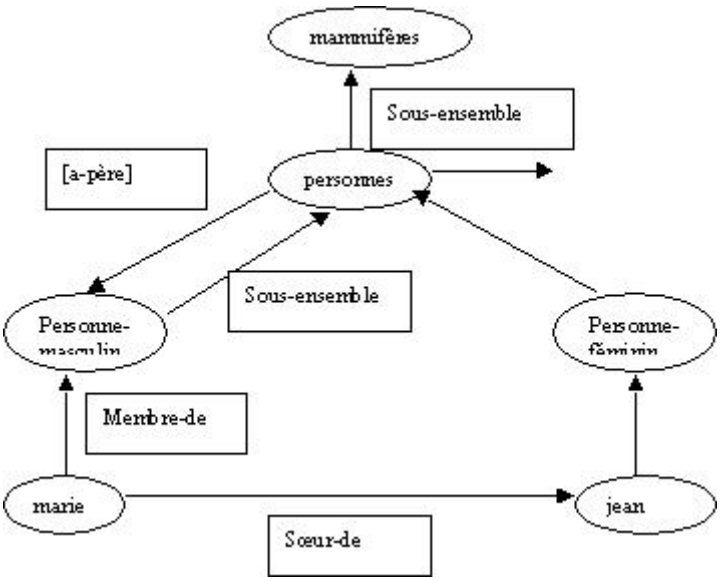


FIGURE 1.1 – Réseaux sémantique

1.2.2 La structure d'héritage :

La structure d'héritage dans les réseaux sémantiques est à la fois simple et riche : par le fait qu'elle est une personne, Marie hérite de toutes les propriétés attachées au nœud personne. L'algorithme d'héritage, à notre connaissance unique, suit les liens membre-de et sous-ensemble de (bien que ce dernier ne soit pas aussi simple qu'il ne le paraisse quant à l'héritage). La simplicité de cet algorithme et sa 'visibilité' graphique a été l'un des atouts majeurs des réseaux sémantiques : toute propriété des objets plus génériques est a priori héritée par l'objet plus spécifique, sans exception.

L'héritage de propriétés, on le sait, devient plus complexe lorsqu'un objet appartient à plus d'une catégorie, et ceci est fréquent dans des situations 'naturelles', on observe aussi des phénomènes dits de blocage lorsqu'une valeur à un niveau inférieur contredit une propriété qui serait héritable. Dans de nombreux environnements de l'époque des réseaux sémantiques, l'héritage multiple était tout simplement banni, dans le meilleur des cas au profit d'une notion de prototypicalité, laquelle imposait ses propriétés aux dépends des autres nœuds pères.

Une autre forme d'inférence intéressante est l'usage des liens inverses. Revenant à l'exemple ci-dessus, a–comme–sœur(Y,X) est la relation inverse de sœur-de(X,Y). Pour que ce lien inverse soit applicable, il faut qu'il corresponde à une réalité conceptuelle et dans les données.

Le principe d'héritage permet :

- de nombreuses déductions automatiques
- de définir la notion de distance sémantique entre 2 concepts = nombre de liens devant être traversés pour aller d'un concept à l'autre.

1.2.3 Quelque limitations :

Le lecteur attentif a pu remarquer une limitation importante des réseaux sémantiques : les relations sont uniquement binaires. Pour coder des relations, il faut passer par une notation événementielle qui relie les arguments :

$$aller(jean, paris, toulouse)$$

$$aller(e, jean) \wedge depart(e, paris) \wedge arrivee(e, toulouse).$$

L'opération de réification est aussi utilisée à ce niveau. Ceci affecte toutefois la simplicité et la lisibilité des réseaux sémantiques.

La réification de propositions permet de représenter toute formule du premier ordre instanciée et libre de fonctions dans la notation des réseaux sémantiques. La quantification universelle pose toutefois des problèmes et requiert des parcours assez indirects.

1.2.4 Réseaux sémantiques et formalismes informatiques

Les approches objet : On peut noter des ressemblances entre les réseaux sémantiques et les approches objet contemporaines de l'informatique. Un nœud est vu comme un objet, et les relations de sous-type et de sous-ensemble établissent des liens d'héritage et d'instance. Comme dans nombre d'approches objet, l'un des grands avantages des réseaux sémantiques est de permettre l'emploi de propriétés avec valeurs par défaut. Si ces défauts sont contredits au niveau d'objets sous-types, alors la valeur de plus bas niveau est préférée, la valeur par défaut étant 'bloquée'. Ce traitement des défauts et exceptions est simple à représenter dans les réseaux sémantiques et les inférences sont faciles à visualiser pour leurs concepteurs. Les graphes conceptuels, initialement développés par IBM, sont une reprise des réseaux sémantiques, auxquels ont été associées des opérations plus élaborées, telle que la jointure de graphes. Bien que modernisés, et largement utilisés en France, ces graphes gardent globalement la puissance des réseaux. Ils ont un habillage informatique plus poussé ainsi que la mise en place d'étiquettes plus standard, par exemple inspirées des rôles thématiques.

Les logiques de description : Récemment, les logiques de description sont apparues avec l'objectif essentiel de formaliser plus précisément les notions élaborées dans les réseaux sémantiques, en particulier en ce qui concerne la structure taxonomique, vue comme le principe organisateur principal, sinon unique. Les formes d'inférences principales sont alors la subsomption (vérifier qu'une catégorie est un sous-ensemble propre d'une autre en comparant leurs définitions), et la classification (contrôler si un objet appartient bien à une catégorie). Certaines approches incluent aussi des traitements de consistance (vérifier que les critères d'appartenance à une catégorie sont satisfiables).

Les applications : Si les réseaux sémantiques sont tombés un peu dans l'oubli, il n'en demeure pas moins que certaines applications informatiques industrielles en utilisent la base. Enfin, la notion d'héritage introduite dans les réseaux sémantiques se retrouve, avec quelques aménagements,

dans diverses approches formelles ou pratiques en linguistique : HPSG pour la syntaxe, DATR pour la morphologie, les systèmes à base de traits en général et, enfin, les structures lexicales hiérarchisées et les ontologies.

1.3 Graphe conceptuel

[?] Un graphe conceptuel est un formalisme de représentation de connaissances et de raisonnements. Ce formalisme a été introduit par John F. Sowa (en) en 1984. Depuis cette date, ce formalisme a été développé suivant trois directions principales : interface graphique de la logique du premier ordre, système diagrammatique pour la logique du premier ordre, formalisme de représentation de connaissances et de raisonnement basé sur les graphes.

Exemple :

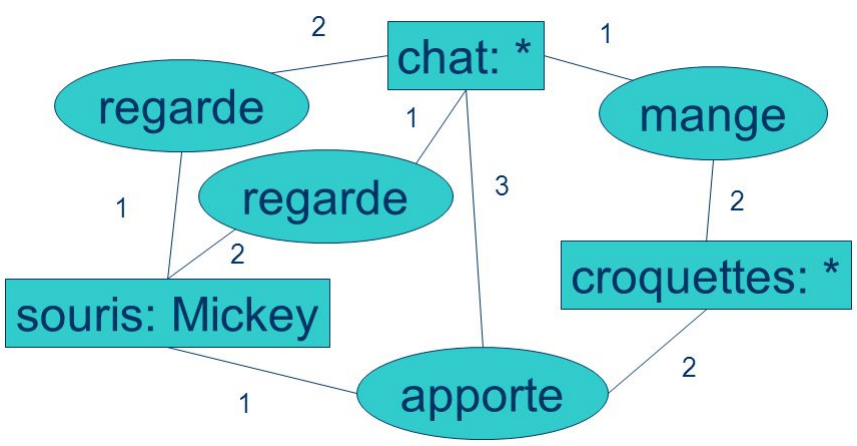


FIGURE 1.2 – Graphe conceptuel

1.3.1 Concepts et relations :

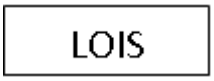
Les graphes conceptuels sont des graphes, c'est-à-dire un ensemble de nœuds liés entre eux. Chaque graphe va permettre de représenter une unité d'information cohérente et complète. Afin de définir le graphe, nous commençons par présenter les briques élémentaires, les nœuds de ces graphes.

Les nœuds constituant les graphes peuvent appartenir à l'un des deux genres suivants : les concepts et les relations. Comme leurs noms le signifient, les concepts représentent des objets, les relations qualifient les interactions qui peuvent exister entre ces objets dans le cadre d'un graphe.

Les concepts sont des « objets mentaux » qui ont une existence reconnue par le cerveau humain et peuvent être définis. Ils ont éventuellement un numéro, une marque qui les différencient les uns des autres. Nous pourrions utiliser comme

concept n'importe quel objet qui nous viendrait à l'esprit, car un concept est théoriquement capable de représenter n'importe quel objet dicible.

Dans le formalisme des graphes conceptuels, les concepts sont représentés par des carrés qui entourent l'expression scripturale du concept exprimé. Nous pouvons illustrons ceci par le concept JURIDIQUE et qui peut alors se représenter simplement de la manière suivante :



Les relations sont des interactions entre les concepts. Elles servent alors à qualifier la structure des propositions impliquant un ensemble de concepts. La plupart du temps, les relations ne sont pas déterminables : leur utilisation nous servira a priori à représenter dans un système ce qui paraît évident pour le cerveau humain. Cette représentation se rapprochera alors plutôt la composante assertionnelle des systèmes.

Dans le formalisme des graphes conceptuels, les relations sont de la même manière représentée par des cercles qui entourent l'expression scripturale de la relation exprimée. Nous illustrons ceci grâce à la relation COUR, qui exprime qu'un concept (le sujet) est l'agent d'un autre concept et qui peut alors se représenter de la manière suivante :

Les concepts et les relations sont deux représentations qui doivent être définis les uns vis-à-vis des autres au cœur d'un système. La stratégie d'un système basé sur les graphes conceptuels sera directement issue de la manière dont ces objets ont été répartis. En général, les relations sont d'un nombre très limité pour faciliter les manipulations et uniformiser la structure des graphes. Mais certains systèmes en viennent à former un nombre plus important de relations pour exprimer plus simplement et plus souplement des assertions complexes.

Le nombre de relations à utiliser peut être augmente. Cependant les concepts sont dans tous les cas les objets élémentaires, une relation peut toujours être définie comme un graphe comportant des relations génériques, qui met en jeu des concepts correspondant à la définition de la relation.

1.3.2 Les types :

Les concepts et les relations sont des représentations issues de l'esprit humain, les premiers correspondant à des termes, les seconds à des liaisons lors de la construction d'assertions à partir de ces termes. Cette dernière organise ces données a l'aide de type qui permettent de regrouper des concepts ou des relations qui ont des propriétés communes sous forme d'ensembles. Ces

types correspondent à une classification inspirée de l'esprit humain, à l'image des réseaux sémantiques, utilisables pour un raisonnement proche de celui de l'homme. Les types sont une marque apposée aux données afin qu'un traitement générique puisse être décliné par des codes différents selon le type de chaque information.

Pour la représentation des types, définit un ensemble de types **T** qui donne tous les types disponibles, ainsi qu'une fonction **t**ype() qui à chaque concept ou relation associe un type $t \in T$. Ces types sont donc souvent partagés entre types de concepts **T**c et types de relations **T**r. Nous définissons un type comme un label, car c'est finalement ce qui permet le mieux de décrire un concept ou une relation. Du point de vue terminologique, le label pourrait être associé à l'intention d'un type. Par ailleurs, il est possible d'obtenir l'ensemble des instances d'un type grâce à la dénotation définie pour n'importe quel type $t \in T$, l'ensemble des concepts qui appartiennent à ce type est noté σt, cette notation correspond alors à l'extension du type.

1.3.3 La quantification :

Les concepts et les relations sont définis existentiellement à partir du moment où un concept ou une relation est introduite dans un graphe, nous pouvons en déduire que cette donnée existe. Pour, le monde des graphes conceptuels est sémantiquement ouvert tout ce qui n'est pas défini peut exister. Tout ce qui est acquis à travers un graphe ou par un mécanisme d'inférence peut être défini existentiellement sous réserve qu'aucune donnée identique n'existe déjà.

Mais définir les concepts par leur type n'est pas suffisant. En effet, l'utilité d'un type est de formaliser des regroupements des individus, sous forme d'ensembles. Or pour en conserver tout le bénéfice, il faut également pouvoir différencier ces individus, afin de savoir si c'est le même individu qui revient dans plusieurs graphes, ou des individus différents d'un même type.

1.3.4 Les ensembles de concepts :

Il possible de décrire extensionnellement des ensembles de concepts. Ceux-ci sont alors constitués de plusieurs individus d'un même type. Les ensembles ne s'appliquent qu'aux concepts : les relations.

Pour utiliser les ensembles, nous devons préciser comment le concept de l'ensemble se factorise en l'ensemble des concepts sous-jacents. À cet effet, définit plusieurs types d'ensembles :

collectifs : on ne considère que la somme des individus.

disjonctifs : on considère l'un des individus sans savoir lequel.

distributifs : on peut considérer chaque individu à la place de l'ensemble.

respectifs : on considère chaque élément vis-à-vis des éléments d'un autre ensemble.

1.3.5 Les arcs :

Les arcs forment la structure du graphe conceptuel par liaison des concepts aux relations. Ces arcs donnent tout le sens du graphe en tant qu'une assertion. Ils sont, par leur nature, des éléments relationnels du graphe et nous les associerons beaucoup plus facilement aux relations qu'aux concepts. En effet, il est possible de voir un concept exister sans arcs, mais une relation sans arcs n'aurait pas de sens par elle-même. Les arcs forment en quelque sorte l'instanciation de chaque relation, en précisant quels concepts la relation met en jeu.

1.3.6 Les contextes :

Permettent de faire d'une proposition (le graphe) un objet mental (le concept). Cela nous donne un moyen de raisonner avec un plus haut niveau d'abstraction : grâce aux contextes, nous pouvons former des relations entre des graphes. Du point de vue de l'interprétation, ces contextes correspondent à des propositions subordonnées, relatives, infinitives et du point de vue de la logique, nous pourrions les qualifier comme prédicats de « deuxième ordre », un raisonnement sur les prédicats du premier ordre.

Nous pouvons également le rapprocher de la logique floue, qui effectue des assertions de vérité sur d'autres assertions.
Pour la représentation, il suffit de placer un graphe à l'intérieur d'un carré.

1.3.7 Forces et faiblesses des Graphes Conceptuels :

Forces :

- C'est un très puissant formalisme de représentation de la connaissance :
  - doté d !une représentation graphique.
  - disposant d !une sémantique formelle (équivalence avec la logique).
- doté d'une représentation graphique.
- disposant d !une sémantique formelle (équivalence avec la logique).
- Nombreux travaux en cours (sur les opérateurs, introduction de notions de vraisemblance, de précision, ...).
- Très efficace en analyse du langage naturel.
- Présence de nombreux outils : CoGiTant, Notio (API Java), CharGer (éditeur GC).

Faiblesses :

- Problèmes de disjonction
- Problèmes de négation
- Problème d !imbrication de quantificateurs

Deuxième partie

Conception

Cette partie est consacrée a la réalisation de graphe conceptuel,dans cette partie nous allons donnés les déférentes étapes a suivre ainsi que les méthodes qu'ils faut appliquées.

Chapitre 2

Réalisation de graphe conceptuel

Pour réaliser un graphe conceptuel de n'importe quel texte il faut en passer par plusieurs étapes qui sont essentiel pour arriver au résultat final ce chapitre explique les déférentes étapes et ce que nous allons faire dans notre projet qui consiste a créé un graphe conceptuel pour des textes de lois pour cela il faudrait récupérer à partir d'un texte de lois les mot clés et les mots les plus répéter dans les texte.

2.1 Le web scraping

Avant de parler de corpus de texte il faut d'abord récupérer des textes des texte sur internet en appel ça le web scraping

2.1.1 Quesque c'est que le web scraping :

Web Scraping (également screen Scraping, Web Data Extraction, Web Harvesting, etc.) est une technique utilisée pour extraire de grandes quantités de données de sites Web où les données sont extraites et enregistrées dans un fichier local de votre ordinateur ou dans une base de données.

2.1.2 Pour quoi le web scraping :

Les données affichées par la plupart des sites Web peuvent uniquement être consultées à l'aide d'un navigateur Web. Ils n'offrent pas la fonctionnalité pour enregistrer une copie de ces données pour un usage personnel. La seule option consiste alors à copier et coller manuellement les données - un travail très fastidieux qui peut prendre plusieurs heures ou parfois plusieurs jours. Web Scraping est la technique d'automatisation de ce processus, de sorte que, au lieu de copier manuellement les données à partir de sites Web, le logiciel Web Scraping effectuera la même tâche en une fraction du temps.

2.1.3 Outils de web scraping :

Il y'a plusieurs outils sur internet qui permettent de faire le web Scraping nous allons parler de quelques un

Import.io :

Import.io scrape les données à partir d'une page web particulière et il est possible d'exporter les données au format CSV [1].

Scrapy :

Scrapy est un framework collaboratif and open-source pour extraire des données. Il est rapide, et facilement extensible, mais il s'adresse aux développeurs ayant des connaissances en python et connaissant XPath [2]

2.2 Corpus de texte

Un corpus est une collection de textes ou de discours regroupé selon un certain nombre de critères prédéterminés le but de corpus est de Tester des hypothèses sur la langue naturelle, Extraction statistique et des information linguistiques.

On peut utiliser des corpus dans plusieurs domaines : études littéraires, linguistiques, scientifiques, philosophie.

2.2.1 Corpus bien formé :

Plusieurs caractéristiques sont à prendre en compte pour dire qu'un corpus est bien formé :

1. csv est un format informatique ouvert représentant des données tabulaires sous forme de valeurs séparées par des virgules
2. XPath est un langage (non XML) pour localiser une portion d'un document XML.

taille :

Le corpus doit évidemment atteindre une taille critique pour permettre des traitements statistiques fiables. Il est impossible d'extraire des informations fiables à partir d'un corpus trop petit.

Langage de corpus :

Un corpus bien formé doit nécessairement couvrir un seul langage, et une seule déclinaison de ce langage. Il existe par exemple de subtiles différences entre le français de France et le français parlé en Belgique. Il ne sera donc pas possible de tirer des conclusions fiables à partir d'un corpus franco-belge sur le français de France, ni sur le français de Belgique.

Temps couvert par les texte de corpus :

Le temps joue un rôle important dans l'évolution du langage : le français parlé aujourd'hui ne ressemble pas au français parlé il y a 200 ans ni, de façon plus subtile, au français parlé il y a 10 ans, à cause notamment des néologismes. C'est un phénomène à prendre en compte pour toutes les langues vivantes. Un corpus ne doit donc pas contenir de textes rédigés à des intervalles de temps trop larges, où il doit les dater (pour un usage par les historiens de la langue ou des concepts).

Registre de langage :

Il ne faut pas non plus mélanger des registres différents et le scientifique ne peut s'autoriser à extraire des informations d'un corpus destiné à un certain registre en les appliquant à un autre. Un corpus construit à partir de textes scientifiques ne peut être utilisé pour extraire des informations sur les textes vulgarisés, et un corpus mélangeant des textes scientifiques et vulgarisés ne permettra de tirer aucune conclusion sur ces deux registres.

2.2.2 Méthodologie :

Il serait maladroit d'un point de vue méthodologique d'appliquer des traitements statistiques sur le corpus qui a permis de faire ressortir un classement ou une modélisation du langage.

Lorsque l'on travaille avec des corpus, il convient donc de séparer un corpus initial en deux sous-corpus :

- Le corpus d'apprentissage, qui sert à retirer un modèle ou un classement à partir d'un nombre suffisant d'information.
- Le corpus de test, qui sert à vérifier la qualité de l'apprentissage à partir du corpus d'apprentissage.

Le calibrage des volumes des corpus se discute en fonction du problème, mais il est fréquent d'utiliser les 2/3 du corpus initial pour l'apprentissage et le tiers restant pour effectuer les tests.

Lorsque le volume du corpus initial n'est pas suffisant, il est possible de croiser les corpus de tests et d'apprentissage sur plusieurs expérimentations.

La mesure de qualité des résultats est alors plus précise, mais en aucun cas les corpus d'apprentissage et de tests n'ont été mélangés.

2.3 Extraction du vocabulaire spécifique à partir d'un corpus web sélectionné

2.3.1 Tf-idf :

Cet algorithme est utile lorsque on a un ensemble de documents, particulièrement un grand, qui doit être catégorisé. Il est particulièrement intéressant parce que nous n'avons pas besoin de former un modèle à l'avance et il va automatiquement tenir compte des différences dans la longueur des documents.

Que signifie TF-IDF :

TF-IDF est acronyme de term frequency-inverse document frequency, le poids TF-IDF est un poids souvent utilisé dans la recherche d'information et l'exploration de texte. Ce poids est une mesure statistique utilisée pour évaluer l'importance d'un mot pour un document dans une collection ou un corpus. L'importance augmente proportionnellement au nombre de fois qu'un mot apparaît dans le document mais est compensé par la fréquence du mot dans le corpus. Les variations du schéma de pondération de TF-IDF sont souvent utilisées par les moteurs de recherche comme un outil central de notation et de classement de la pertinence d'un document en fonction d'une requête de l'utilisateur.
Une des fonctions de classement les plus simples est calculée en sommant le TF-IDF pour chaque terme de requête ; de nombreuses fonctions de classement plus sophistiquées sont des variantes de ce modèle simple.
TF-IDF peut être utilisé avec succès pour le filtrage des mots d'arrêt dans divers domaines, y compris la synthèse et la classification des textes.

TF :

mesure la fréquence d'apparition d'un terme dans un document. Comme chaque document est de longueur différente, il est possible qu'un terme apparaisse plusieurs fois dans les documents longs que dans les documents plus courts. Ainsi, le TF est souvent divisé par la longueur du document.

IDF :

mesure l'importance d'un terme. Lors du calcul de TF, tous les termes sont considérés comme importants. Cependant, il est connu que certains termes, tels que "est", "de", et "cela", peuvent apparaître plusieurs fois mais ont peu d'importance. Ainsi, nous devons peser les termes fréquents tout en augmentant les moins fréquents.

Formules mathématiques TF-IDF

$$W_{x,y} = tf_{x,y} * log(\frac{n}{df})$$

Au fil des années, la formule de poids TF*IDF a été perfectionnée, de nombreuses variantes ont alors été inventés et testées. Plus récemment, l'une de celles qui a fourni les meilleurs résultats dans un moteur de recherche est connue sous le nom "Okapi BM25" dont voici la formule :

$$BM25 = \sum_{i=1}^{W} \frac{TF(i)(1+k)}{TF(i)+k(1-b+b\frac{DL}{avgDL})} IDF(i)$$

$$IDF(i) = \frac{\log(\frac{N-n+1}{n})}{\log(N)}$$

Calculer TF-IDF :

L'importance augmente proportionnellement au nombre de fois qu'un mot apparaît dans le document individuel lui-même - c'est ce qu'on appelle la fréquence des termes. Cependant, si plusieurs documents contiennent plusieurs fois le même mot, vous rencontrez un problème. C'est pourquoi TF-IDF compense également cette valeur par la fréquence du terme dans l'ensemble du document, une valeur appelée fréquence de document inverse.

$TF(t) = (Nombre de fois que le terme t apparaît dans un document) / (Nombre total de termes dans le document)$

$IDF(t) = log_e(Nombre total de documents / Nombre de documents contenant le terme t). Valeur = TF * IDF$

TF-IDF est calculé pour chaque terme dans chaque document.

2.3.2 Exemple graphe conceptuel sur un texte juridique :

un texte juridique a été exploité afin d'extraire les différentes l'information du corpus. Le terme le plus utiliser et répéter va être mis en avant, avec des entités nommées.

Nous avons utilisé l'algorithme tf-idf pour sélectionner les mot qui paraissent comme les plus pertinents par rapport à une requête précise, l'algorithme permet de sélectionner uniquement les mots qui caractérisent le mieux un ensemble de documents générés par une requête donnée par rapport à un corpus de référence.

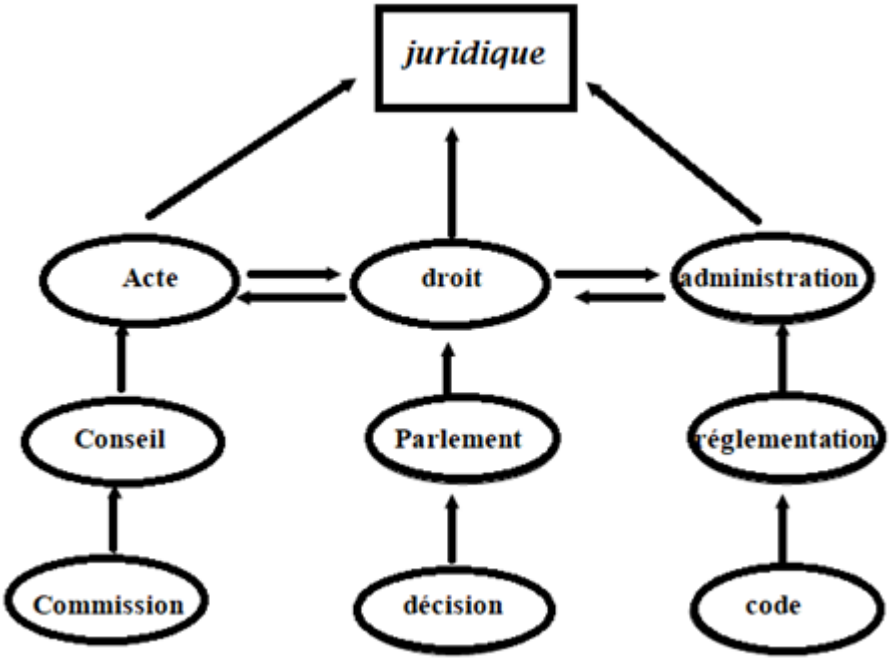


FIGURE 2.1 – Graphe conceptuel d'un texte de juridique

2.3.3 Exemple de programme TF-IDF

TF-IDF en JAVA :

```
1  //https://gist.github.com/guenodz/d5add59b31114a3a3c66
2  import java.util.Arrays;
3  import java.util.List;
4
5
6  public class TFIDFCalculator {
7      /*Term Frequency(t,d)*/
8      public double tf(List<String> doc, String term) {
9          double result = 0;
10         for (String word : doc) {
11             if (term.equalsIgnoreCase(word))
12                 result++;
13         }
14         return result / doc.size();
15     }
16
17     /*Inverse Term Frequency(t,D)*/
18     public double idf(List<List<String>> docs, String term) {
19         double n = 0;
20         for (List<String> doc : docs) {
21             for (String word : doc) {
```

```
                if (term.equalsIgnoreCase(word)) {
                    n++;
                    break;
                }
            }
        }
        return Math.log(docs.size() / n);
    }


    public double tfIdf(List<String> doc, List<List<String>> docs, String
         term) {
        return tf(doc, term) * idf(docs, term);

    }

    public static void main(String[] args) {

        List<String> doc1 = Arrays.asList("Lorem", "ipsum", "dolor", "ipsum",
             "sit", "ipsum");
        List<String> doc2 = Arrays.asList("Vituperata", "incorrupte", "at",
             "ipsum", "pro", "quo");
        List<String> doc3 = Arrays.asList("Has", "persius", "disputationi", "id",
             "simul");
        List<List<String>> documents = Arrays.asList(doc1, doc2, doc3);

        TFIDFCalculator calculator = new TFIDFCalculator();
        double tfidf = calculator.tfIdf(doc1, documents, "ipsum");
        System.out.println("TF-IDF (ipsum) = " + tfidf);
```

Le résultat est : TF-IDF (ipsum) = 0.2027325540540822

TF-IDF en PYTHON :

```
#https://gist.github.com/guenodz/d5add59b31114a3a3c66

import re
import nltk
from nltk.tokenize import RegexpTokenizer
from nltk import bigrams, trigrams
import math


stopwords = nltk.corpus.stopwords.words('portuguese')



def freq(word, doc):
    return doc.count(word)


def word_count(doc):
    return len(doc)
```

```
20
21
22  def tf(word, doc):
23      return (freq(word, doc) / float(word_count(doc)))
24
25
26  def num_docs_containing(word, list_of_docs):
27      count = 0
28      for document in list_of_docs:
29          if freq(word, document) > 0:
30              count += 1
31      return 1 + count
32
33
34  def idf(word, list_of_docs):
35      return math.log(len(list_of_docs) /
36              float(num_docs_containing(word, list_of_docs)))
37
38
39  def tf_idf(word, doc, list_of_docs):
40      return (tf(word, doc) * idf(word, list_of_docs))
41
42  #Compute the frequency for each term.
43  vocabulary = []
44  docs = {}
45  all_tips = []
46  for tip in (['documment 1', 'documment 2']):
47      tokens = tokenizer.tokenize(tip.text)
48
49      bi_tokens = bigrams(tokens)
50      tri_tokens = trigrams(tokens)
51      tokens = [token.lower() for token in tokens if len(token) > 2]
52      tokens = [token for token in tokens if token not in stopwords]
53
54      bi_tokens = [' '.join(token).lower() for token in bi_tokens]
55      bi_tokens = [token for token in bi_tokens if token not in
            stopwords]
56
57      tri_tokens = [' '.join(token).lower() for token in tri_tokens]
58      tri_tokens = [token for token in tri_tokens if token not in
            stopwords]
59
60      final_tokens = []
61      final_tokens.extend(tokens)
62      final_tokens.extend(bi_tokens)
63      final_tokens.extend(tri_tokens)
64      docs[tip] = {'freq': {}, 'tf': {}, 'idf': {},
65                       'tf-idf': {}, 'tokens': []}
66
67      for token in final_tokens:
```

```
68          #The frequency computed for each tip
69          docs[tip]['freq'][token] = freq(token, final_tokens)
70          #The term-frequency (Normalized Frequency)
71          docs[tip]['tf'][token] = tf(token, final_tokens)
72          docs[tip]['tokens'] = final_tokens
73
74      vocabulary.append(final_tokens)
75
76  for doc in docs:
77      for token in docs[doc]['tf']:
78          #The Inverse-Document-Frequency
79          docs[doc]['idf'][token] = idf(token, vocabulary)
80          #The tf-idf
81          docs[doc]['tf-idf'][token] = tf_idf(token,
                docs[doc]['tokens'], vocabulary)
82
83  #Now let's find out the most relevant words by tf-idf.
84  words = {}
85  for doc in docs:
86      for token in docs[doc]['tf-idf']:
87          if token not in words:
88              words[token] = docs[doc]['tf-idf'][token]
89          else:
90              if docs[doc]['tf-idf'][token] > words[token]:
91                  words[token] = docs[doc]['tf-idf'][token]
92
93      print doc
94      for token in docs[doc]['tf-idf']:
95          print token, docs[doc]['tf-idf'][token]
96
97  for item in sorted(words.items(), key=lambda x: x[1],
        reverse=True):
98      print "%f <= %s" % (item[1], item[0])
```

La Webographie

Graphe conceptuel https://fr.wikipedia.org/wiki/Graphe_conceptuel

Web scraping https://www.webharvy.com/articles/what-is-web-scraping.htmlwebscraping

Outils de scraping http://1001startups.fr/startup-5-outils-pour-scraper-des-donnees-en-ligne

TF-IDF http://www.tfidf.com/

Conclusion et Perspectives

Dans ce rapport, nous avons fait un état de l'art de la des connaissances ensuite nous avons présenté les qu'il faut suivre. Cela nous a permit de découvrir les déférentes étapes pour réaliser un graphes conceptuel trouver sur internet. Cette étude nous a nous a éclairée sur la manière par la quel nous procèderons afin de développement de notre projet.

notre seul regret c'est de ne pas avoir eu plus de temps pour mener a bien nos recherches.

Au prochain semestre, nous allons mettre en pratique les connaissance théoriques acquises tout au cours de de cette périodes ou on a réaliser ce rapport.