ÉTABLIR LES CORRESPONDANCES ENTRE LES ARTÉFACTS D'UN SYSTÈME D'INFORMATION

THÈSE PRÉSENTÉE COMME EXIGENCE PARTIELLE

DU DOCTORAT EN INFORMATIQUE COGNITIVE - DIC

PAR

TAREK KHEI

SEPTEMBRE 2020

http://r-libre.teluq.ca/2092

TÉLUQ

ÉTABLIR LES CORRESPONDANCES ENTRE LES ARTÉFACTS D'UN SYSTÈME D'INFORMATION

THÈSE

PRÉSENTÉE

COMME EXIGENCE PARTIELLE

DOCTORAT EN INFORMATIQUE COGNITIVE - DIC

PAR

TAREK KHEI

SEPTEMBRE 2020

A mes parents Zbir El Kamla, Mohamed Khei, mes grands parents, mes sœurs, ma femme et mes enfants Youssef et Laya, à ma grande famille.

REMERCIEMENTS

À la fin de ce travail, je souhaite témoigner mes sentiments de gratitudes à l'égard de toutes les personnes qui ont aidé de près ou de loin pour la réalisation de cette thèse.

Mes remerciements s'adressent en premier et surtout à mes directeurs de thèse Daniel Lemire (Professeur titulaire en informatique dans le Département Science et technologie de la TELUQ) et Richard Hotte (Directeur du Centre de recherche en informatique cognitive et environnements fonctionnels et professeur titulaire en technologie de l'information dans le Département Science et technologie de la TELUQ), qui m'ont guidé du début jusqu'à la réalisation de ce travail, avec leur encadrement, leur conseil, leur support, leur disponibilité et surtout leur rétroaction efficace et leur grande patience. J'ai beaucoup appris de chacun de vous, l'efficacité, la gestion de temps, la profondeur des recherches et la focalisation sur les idées et les points importants.
Je remercie en particulier Daniel Lemire pour sa rétroaction rapide et efficace, le temps de réponse pour commenter les chapitres aussi efficacement est vraiment impressionnant. Le fait de recevoir une rétroaction pertinente, détaillée et précise à 7 h du matin sachant que j'ai envoyé le chapitre 3 heures avant est tout simplement motivant, encourageant et exemplaire. J'espère être à la hauteur de votre niveau d'encadrement et de support.

Je remercie Sarah Bernier pour sa révision linguistique en un temps record, je remercie Mélanie Samson, Djamila Abbas, Claude Breault pour leur disponibilité, leur aide et support, je remercie tout le personnel de la Teluq et de l'UQAM.

Mes remerciements s'adressent également aux professeurs que j'ai eu le plaisir de rencontrer tout au long de mon parcours universitaire. Je remercie également les professeurs qui ont participé à l'évaluation de ce travail.

A mes parents El Kamla Zbir et Mohamed Khei, merci de m'avoir permis d'être et de devenir ce que je suis, merci pour l'amour inconditionel et la grande confiance. Que ce travail soit une source de joie et de fiereté, aucun mot ne pourrait exprimer la gratitude et l'amour que je vous porte. À mes sœurs Hind et Houda, à ma femme Ouissal, à mes enfants Youssef et Laya, que ce travail soit un exemple et une source de motivation pour vous. À ma grande famille et à mes amis, pour leur soutien continu.

.

TABLE DES MATIÈRES

LISTE DES FIGURES

LISTE DES TABLEAUX

LISTE DES ABRÉVIATIONS, DES SIGLES ET DES ACRONYMES

ALD	Allocation latente de Dirichlet
ANSI	American National Standards Institute
API	Application Programming Interface
BPMN	Business Process Model and Notation
CASE	Computer-aided Software Engineering
COO	Conception orientée objet
CRM	Customer Relationship Management
HATS	High Assurance Transformation Systems
HDLTex	Hierarchical Deep Learning for Text Classification

IEC	International Electrotechnical Commission
IEEE	Institute of Electrical and Electronics Engineers
IIBA	International Institute of Business Analysis
IMDB	Internet Movie Database
IR	Information Retrieval
ISL	Indexation sémantique latente
ISO	International Standard Organism
LNP	Local Number Porting
MVS	Machine à vecteurs de support
NASA	National American Space Agency
NLTK	Natural Language Toolkit
RTCM	Restricted Test Case Modeling
TAL	Traitement automatique du langage
TF-IDF	Term Frequency Inversed Document Frequency
UCI	University of California Irvine

UMass	University of Massachusetts
UML	Unified Modeling Language
VCD	Virtual Company Dossier
WMD	Word Mover's Distance

RÉSUMÉ

Dans le cadre des travaux relatifs au cycle de développement logiciel, les analystes des systèmes gèrent un grand nombre d'informations sous forme d'artéfacts produits au cours des phases de développement. Les chercheurs du domaine de l'extraction de l'information ont déterminé les correspondances entre ces artéfacts en proposant des approches qui permettent de rétablir les traces entre eux.

L'objectif de notre travail consiste à proposer une nouvelle approche qui établit les correspondances entre les artéfacts à partir des exigences textuelles. Nous appliquons un modèle non supervisé pour explorer les thèmes dans les exigences en s'appuyant sur la rétroaction des spécialistes du domaine afin d'évaluer les résultats obtenus. Nous avons évalué dix documents à partir de huit sources de données distinctes. Les résultats obtenus ont montré que notre approche permet d'établir les correspondances sans aucune connaissance préalable du domaine et du contenu des exigences. De plus, elle peut s'appliquer à des exigences relatives à différents domaines d'affaires.

Notre approche comprend deux étapes principales : 1) exploration du contenu des exigences et extraction des artéfacts qui appartiennent aux thèmes (exigences, cas d'utilisation, processus d'affaires, entités conceptuelles et cas d'essai) et 2) identification des correspondances entre les artéfacts en calculant leur similarité sémantique. La rétroaction humaine a permis d'évaluer la performance du modèle en calculant la précision et le rappel des résultats. Finalement, nous proposons des pistes de recherche qui permettront d'enrichir les connaissances dans le domaine de l'extraction de l'information pour établir les correspondances à partir des exigences entre les éléments des systèmes.

Mots clés : Traçabilité des exigences, extraction de l'information, système d'information, allocation latente de Dirichlet, similarité sémantique, rétroaction.

ABSTRACT

As part of the software development cycle, information systems analysts manage a large amount of information in the form of artifacts produced through the development phases. Researchers—in the field of the information extraction—have targeted the connections between these artifacts by proposing approaches that help to restore the traces between these elements.

The goal of our work is to propose a new approach that establishes traces between artifacts from textual requirements. Our method uses an unsupervised model to explore the themes in the requirements using subject matter expert feedback to evaluate the results obtained. We evaluated ten documents from eight different data sources. The results showed that our approach makes it possible to establish the correspondences without any prior knowledge of the domain and the content of the requirements.

Our approach is based on two main steps: (1) Exploration of the requirements content and extraction of the artifacts that belong to the topic (requirements, use cases, business processes, conceptual entities and test cases). (2) Identification of the correspondences between the artifacts by calculating the semantic similarity between the elements. Human feedback was used to evaluate the model's performance by calculating the accuracy and recall of the results. Our approach can be applied to requirements related to different business areas. Finally, we propose research subjects by suggesting topics that will help advance the research of the extraction of information to establish the traceability from the requirements.

Keywords: Requirements traceability, information extraction, information systems, Latent Dirichlet Allocation, semantic similarity, feedback.

CHAPITRE I

ÉTABLIR LES CORRESPONDANCES ENTRE LES ARTÉFACTS D'UN SYSTÈME D'INFORMATION

1.1 Introduction

Le cycle de développement logiciel représente un processus qui regroupe plusieurs éléments que les analystes et concepteurs informatiques créent et produisent dans chaque phase du cycle. Le développement des systèmes d'information consiste à produire et à modifier un ensemble d'éléments qui sert à concevoir le logiciel ou le produit fini. Le succès de l'évolution des systèmes dépend d'un accès rapide et bref à l'information générée tout au long du cycle.

Les ingénieurs informatiques créent et mettent à jour, surtout pour les grands projets, une grande quantité d'information formelle et informelle (Cleland-Huang, Settimi, Duan, & Zou, 2005). Cette grande quantité d'information dessine l'écosystème du cycle logiciel. Elle inclut le code source, les exigences, les spécifications requises pour les différents niveaux d'abstraction relatifs aux phases de développement, les descriptions des essais, les rapports des anomalies, etc. Cet écosystème se structure sous forme d'artéfacts enregistrés dans des produits livrables dans des systèmes de gestion des documents, des bases de données, des exigences ou un référentiel du code. Ces biens livrables servent à gérer manuellement les traces entre les différents artéfacts du système (Gotel & Finkelstein, 1994).

La maintenance manuelle des traces entre les artéfacts ainsi que la dynamique du développement logiciel impliquent un travail fastidieux et sont sources d'erreurs (Hayes, Dekhtyar, & Sundaram, 2006). Les ingénieurs bénéficieront de la traçabilité pour rechercher les informations pertinentes et pour naviguer efficacement dans l'écosystème du développement logiciel. En effet, l'accès aux correspondances entre les artéfacts du système facilite l'exécution de tâches telles que l'analyse d'impact, l'identification des artéfacts réutilisables et la validation des exigences (Winkler & Pilgrim, 2010). Dans le même sens, les recherches ont confirmé que l'absence de traces et de correspondances entre les artéfacts des systèmes provoque l'échec des projets logiciels (Gotel & Finkelstein, 1994).

1.2 Contexte de la recherche

Établir les correspondances entre les éléments d'un système d'information fait partie des tâches qui permettent la traçabilité des systèmes. Cela consiste à retrouver les liens entre les composantes conçues et produites dans un processus de développement logiciel. Cette traçabilité, ou correspondance entre les éléments ou artéfacts, apparait grâce à deux opérations (ISO/IEC/IEEE 24764, 2010) :

- établir la relation entre deux ou plusieurs produits du processus de développement qui partagent un lien de type prédécesseur et successeur;
- établir la source de chaque élément dans un système d'information.

Les ingénieurs logiciels créent et modifient les artéfacts tout au long du processus de développement, afin de réaliser un produit/logiciel final (Winkler & Pilgrim, 2010).

Les correspondances entre les artéfacts sont retrouvées lorsque nous avons la capacité de décrire et de garder trace des liens dans les deux directions (Gotel & Finkelstein, 1994) :

- vers l'avant : c'est la direction qui permet de tracer les liens d'une exigence à partir de ses origines. Elle consiste à lister les relations en passant par son développement et ses spécifications jusqu'à son déploiement et son utilisation dans un processus d'ajustement et d'itérations dans ou durant toutes les phases du cycle de vie logiciel;
- vers l'arrière : c'est la direction qui permet de découvrir les liens d'une exigence à partir d'une composante transformée : code, modèle ou tout autre artéfact, jusqu'à ses origines.

Établir les correspondances entre les artéfacts à partir des exigences permettra de découvrir et d'explorer leurs liens. Pour cela, il s'agira d'explorer le contenu textuel des exigences. Ces dernières se définissent en respectant les trois conditions suivantes (ANSI/IEEE 830, 1993) :

1. une condition ou capacité requise par un utilisateur pour résoudre un problème ou atteindre un objectif;
2. une condition ou une capacité qu'un système doit obligatoirement assurer pour respecter un contrat, une norme, une spécification ou tout autre document formellement imposé;
3. une représentation documentée d'une condition ou d'une capacité, comme décrite dans les points 1 et 2.

Les correspondances ou traces se distinguent également par la phase à partir de laquelle les analystes les capturent. Gotel & Finkelstein (1994) ont décrit deux types de traces :

- avant spécification : cette méthode permet d'établir les liens durant les phases préliminaires, notamment les phases d'élicitation, de discussions et de validation des exigences. Nous nous intéressons aux liens, avant la transformation vers les artéfacts, que les analystes d'affaires consignent dans le document final des exigences. Les concepteurs utilisent ce document dans les phases subséquentes de conception, de développement, de tests et de déploiement;
- après spécification : cette méthode permet d'établir les liens généraux durant l'implémentation pas à pas d'une exigence. Elle cible les artéfacts produits tout au long des phases de conception, de réalisation, de tests et de déploiement. Elle couvre les correspondances entre les artéfacts générés à toutes les étapes de transformations, manuelles ou automatiques, pour la réalisation du système.

Selon la perspective d'utilisation, établir les correspondances est une étape importante dans un cycle de vie logiciel. Ramesh (1998) a identifié deux types de traçabilité :

- niveau technique : un ingénieur qui a trouvé une erreur en vérifiant ou en testant une composante peut s'appuyer sur les correspondances pour remonter vers les artéfacts qui appartiennent aux phases précédentes jusqu'aux exigences initiales. Il pourra alors expliquer la raison de l'erreur en identifiant les dépendances dans les phases du cycle de vie logiciel;
- niveau affaires : un gestionnaire pourra, en réponse à une demande de changement, extraire les composantes reliées à cette évolution afin de faciliter l'analyse d'impact et la propagation du changement en question.

Nous nous intéressons aux liens post-spécifications, plus précisément à l'extraction des liens entre les exigences et les artéfacts, qui représentent les cas d'utilisation, les processus d'affaires, les entités conceptuelles et les cas d'essai. Pinheiro (2004) a identifié ce genre de relations comme des correspondances extra-exigences.

Une correspondance se définit comme un lien de causalité, de type ou de contenu (Winkler & Pilgrim, 2010). Un lien est une forme d'information plus concrète qui décrit certains éléments du cycle de vie logiciel. Spanoudakis & Zisman (2005) ont décrit des catégories pour distinguer les correspondances entre les artéfacts. Nous nous intéressons plus spécifiquement à la relation de type dépendance. Celle-ci consiste en une relation d'existence entre deux artéfacts a1 et a2, lorsque chaque artéfact dépend de l'existence de l'autre.

Les chercheurs se sont de plus en plus intéressés à l'extraction des correspondances pour l'ingénierie des exigences. Une correspondance représente tout lien entre deux artéfacts (Aizenbud-Reshef, Nolan, Rubin, & Shaham-Gafni, 2006). Des recherches ont permis d'explorer la découverte automatique des correspondances. Cependant, l'industrie n'a pas encore adopté les solutions proposées (Zhang & al., 2014). Elle a constaté l'importance et la nécessité d'ajouter une intervention humaine pour confirmer les liens découverts, spécialement pour les exigences, en raison de la richesse de la sémantique du texte écrit en langage naturel.

Les correspondances des exigences jouent un rôle important dans l'analyse de la propagation des changements (Zhang & al., 2014). Zhang & al. (2014) ont proposé des modèles ciblant différents niveaux d'abstraction et différents types de catégorisation. Ils ont aussi classifié les types de dépendances par rapport à leur utilisation dans les projets. L'analyse d'impact des changements, par exemple, concerne les dépendances de similarité, de référence ou à toute relation générée par un enchainement logique/séquentiel de transformation entre les artéfacts au cours des phases de développement du système.

1.3 Contexte de la problématique

La tâche de découvrir les correspondances à partir des exigences est une nécessité dans le développement logiciel. C'est une tâche simple qui consiste à suivre les liens dans un système d'information. Établir les correspondances entre les exigences constitue un prérequis en raison de la grande masse d'information produite (Pinheiro, 2004). Lors du développement de grands systèmes complexes, se souvenir de tous les liens utilisés entre les informations est difficile en raison du grand nombre d'artéfacts et de composantes produites et transformées.

L'étude de correspondances est peu avancée : les recherches et les contributions sur le sujet s'appliquent uniquement dans les laboratoires (Winklere & Pilgrim, 2010). C'est un domaine qui requiert des recherches plus approfondies. Le défi ne réside pas uniquement dans la difficulté des questions de recherche, mais aussi, surtout, dans l'engagement ou l'intérêt de communautés diverses (ingénierie des exigences, modélisation…) qui communiquent rarement entre elles. Cette diversité représente un enjeu en ce qui concerne la compréhension commune et la réinvention de la roue par les communautés (Winklere & Pilgrim, 2010).

Les dépendances entre les exigences influencent considérablement les activités de l'ingénierie logicielle. Citons la planification des projets, l'architecture, la conception et l'analyse d'impact des changements (Zhang & al., 2014). L'évolution des systèmes pour les adapter aux changements dans les besoins des utilisateurs, des environnements d'affaires et technologiques requiert de contrôler le risque et de pouvoir identifier les artéfacts affectés par cette évolution. Les chercheurs se sont concentrés sur l'évolution des systèmes en ce qui concerne le code, la réingénierie et la conception. Zhang & al. (2014) ont suggéré d'analyser la propagation des changements dans le domaine des affaires en incluant également les exigences.

Ces chercheurs ont proposé des modèles de dépendances pour refléter la relation complexe entre les exigences et les niveaux sémantiques et structuraux des systèmes.

Zhang & al. (2014) ont adopté ces dépendances ayant différents niveaux d'abstraction pour différents aspects de gestion de projets informatiques. Cependant, les auteurs constatent un manque d'évaluation dans l'applicabilité et l'efficacité de l'analyse des changements pour des projets réels.

L'ingénierie des systèmes d'information permet de suivre un processus structuré. Ce processus comporte des tâches manuelles, idéalement exécutées par des analystes ou des ingénieurs expérimentés (Landhäußer, Körner, & Tichy, 2014). Cela inclut la création des spécifications textuelles et des modèles qui contiennent les artéfacts du système. Lors de l'exécution de ce processus, les changements des exigences et les décisions d'implémentation ont une incidence sur la maintenance et la synchronisation des composantes, des spécifications et des modèles. Afin de livrer le produit final tel qu'il est requis, une synchronisation des composantes s'avère nécessaire. Cependant, comme le cout de la consistance, dans la pratique, est élevé, les analystes doivent identifier les correspondances pour faciliter le recouvrement des liens entre les éléments.

Dans la figure 1.1, nous montrons les liens qui représentent les correspondances entre les artéfacts que nous ciblons dans notre thèse.

Les utilisateurs rédigent les exigences. Les architectes et concepteurs analysent ces exigences afin de produire des artéfacts tels que les cas d'utilisations, les processus d'affaires et les entités conceptuelles. Les programmeurs écrivent du code pour réaliser

le produit tel que dicté par les spécifications. Les concepteurs de tests conçoivent des cas d'essai afin de vérifier et de valider les fonctionnalités du système.

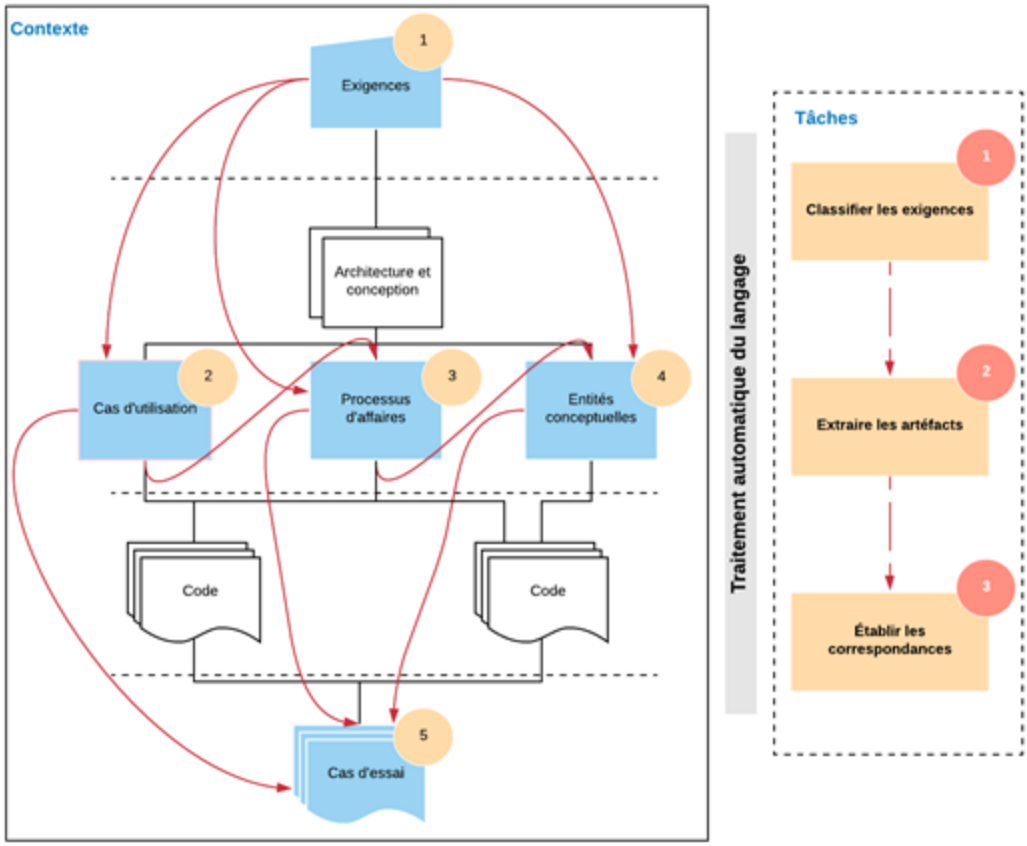


Figure 1.1 Contexte d'établissement des correspondances entre les artéfacts

Nous considérons les classes de documents suivantes :

- exigences : les analystes d'affaires rédigent les exigences en utilisant soit un langage naturel soit des modèles abstraits (Bakar, Kasirun, & Salleh, 2015). Les exigences textuelles se retrouvent sous forme de listes, d'objectifs et de fonctionnalités, de descriptions de produit, de manuels d'utilisateur ou de scénarios. Les modèles se retrouvent dans des diagrammes UML comme les diagrammes de flux de données, les diagrammes de classes et les diagrammes de comportementaux. Les modèles sont pour la plupart accompagnés de

descriptions textuelles des caractéristiques et des fonctionnalités (Nicolás & Toval, 2009);

- cas d'utilisation : les cas d'utilisation font partie des diagrammes UML employés pour décrire le comportement du système. Les concepteurs produisent des modèles dynamiques à partir des exigences textuelles pour décrire les actions conséquentes exécutées par divers acteurs qui participent aux processus de travail du système. Ils décrivent les relations entre les composantes et les acteurs du système en prenant en compte des perspectives différentes (Ilieva, 2007);
- processus d'affaires : les processus d'affaires supportent les phases de conception et documentent les processus qu'un système d'information couvre (Leopold, Eid-Sabbagh, Mendling, Azevedo, & Baiao, 2013). Les processus d'affaires représentent l'agrégation de toutes les tâches et activités exécutées par les organisations afin de créer des services et de les fournir aux clients (Aysolmaz, Leopold, Reijers, & Demirörs, 2018);
- entités conceptuelles : la modélisation conceptuelle consiste à créer et concevoir les entités conceptuelles à partir des exigences. Elle décrit les informations telles que les entités, les attributs et leur relation. C'est une tâche abstraite qui joue un rôle important dans la rédaction des spécifications, l'analyse et le développement des systèmes (Btoush & Hammad, 2015);
- cas d'essai : les spécifications des cas d'essai sont un moyen commun pour documenter les essais à exécuter pour valider un système. Les concepteurs utilisent ces spécifications pour tester et valider les fonctionnalités, manuellement ou automatiquement. Comme le veut la pratique courante, les concepteurs de tests utilisent, le langage naturel pour rédiger les spécifications (Yue, Ali, & Zhang, 2015).

Pour analyser les exigences textuelles, nous appliquons les techniques de traitement automatique du langage (TAL). Le TAL permet de traiter à la fois l'informatique et la linguistique. C'est un champ de recherche lié à l'intelligence artificielle et à la linguistique computationnelle (Btoush & Hammad, 2015). Les chercheurs et professionnels explorent le TAL en continu pour améliorer les modèles et représentations (Btoush & Hammad, 2015).

Le TAL consiste à effectuer un traitement computationnel ayant deux objectifs (Kamarudin, Sani, & Atan, 2015) :

- créer un modèle de représentation computationnel (base de connaissances par exemple);
- exploiter le modèle pour comprendre et générer le langage approprié pour un ensemble de tâches cognitives.

Le TAL permet de convertir automatiquement l'information recélée dans un langage naturel en un format compréhensible par la machine. Il consiste à extraire les connaissances à partir des données non structurées et hautement ambiguës ayant une grammaire complexe.

1.4 Tâches cognitives

Notre objectif consiste à explorer les exigences et spécifications textuelles pour établir les correspondances entre les artéfacts. Au préalable, cette exploration passe par la classification et l'identification des documents. Vlas & Robinson (2012) ont affirmé également que la classification fournit des taxonomies communes relatives aux types d'exigences. En partant du même principe, nous effectuons la classification des exigences et l'identification des artéfacts avant d'établir les liens entre ces éléments.

À partir des exigences et spécifications textuelles, nous commençons par identifier et classifier les documents textuels qui contiennent les artéfacts. Par la suite, nous procédons à une extraction de ces artéfacts avant d'établir les correspondances.

La figure 1.2 montre les tâches cognitives requises pour établir les correspondances.

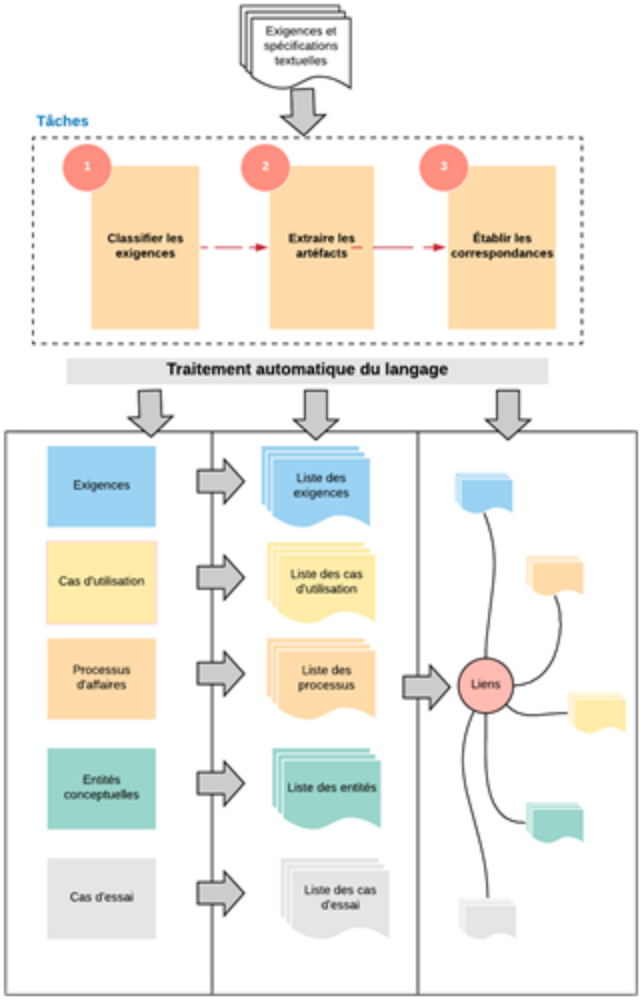


Figure 1.2 Tâches cognitives requises pour établir les correspondances entre les artéfacts

Nous considérons les trois tâches cognitives suivantes :

- classifier les exigences : cette étape consiste à distinguer les spécifications textuelles selon leur type (relatives aux exigences, aux cas d'utilisation, aux processus d'affaires, aux entités conceptuelles ou aux cas d'essai). Les spécifications peuvent faire partie d'un ou de plusieurs documents qui contiennent des paragraphes ou des sections relatifs aux classes, caractéristiques communes, suivantes :

  - exigences,
  - cas d'utilisation,
  - processus d'affaires,
  - entités conceptuelles,
  - cas d'essai;
- extraire les artéfacts : cette étape consiste à extraire la liste des artéfacts qui appartiennent à chaque classe. Nous identifions les éléments pertinents pour chaque catégorie. Puis, nous procédons à l'extraction des artéfacts par type de document classifié :
  - exigences : la liste des exigences que les utilisateurs ont rédigées dans la phase préliminaire,
  - cas d'utilisation : la liste des cas d'utilisation que les concepteurs et analystes produisent à partir des exigences des utilisateurs,
  - processus : la liste des processus d'affaires que les concepteurs ont identifiés et consignés dans les documents de processus,
  - entités conceptuelles : la liste des concepts ou classes que les concepteurs ont décrits dans les documents de spécifications,
  - cas d'essai : la liste des cas d'essai que les concepteurs de tests ont décrits dans les documents d'essai;
- établir les correspondances : cette étape consiste à extraire les correspondances entre les artéfacts identifiés dans la phase précédente. Nous visons à reconnaitre les liens sémantiques entre les artéfacts de la même manière qu'un concepteur ou un architecte établira, par exemple, les correspondances entre une exigence et un cas d'utilisation, un processus ou un cas d'essai.

1.5 Structure de la thèse

Nous structurons notre thèse en six chapitres, introduits par une exploration des recherches relatives à l'ingénierie des exigences et au TAL. Nous choisissons notre approche en comparant les performances des modèles et nous la décrivons en spécifiant les techniques à utiliser pour chaque tâche cognitive. Par la suite, nous procédons à l'évaluation de l'approche. Finalement, nous terminons par les contributions et les perspectives pour les travaux futurs.

Chapitre II : Ingénierie des exigences et TAL

Nous explorons la littérature pour étudier les techniques de TAL que les chercheurs ont exploitées pour analyser les exigences textuelles ainsi que les relations extraites entre les artéfacts qui appartiennent à un système d'information. Nous identifions les objectifs des recherches ainsi que leur couverture d'un ou de plusieurs artéfacts dans le cadre de l'ingénierie des exigences. Nous avons observé que, d'après les articles consultés, le TAL facilite l'extraction des éléments produits à partir des documents textuels.

Chapitre III : Évaluation des modèles

Nous parcourons les articles concernant l'évaluation des modèles de l'apprentissage profond dans le cadre d'expérimentations pour des tâches liées au TAL. Les recherches récentes ont utilisé ce type de modèle pour l'analyse des sentiments, la classification et d'autres tâches cognitives qui traitent le langage naturel. Ces modèles ont prouvé leur efficacité dans la représentation et le raisonnement cognitif (Gehrmann & al., 2018; Kowsari & al., 2017). Nous choisirons les modèles qui ont obtenu les meilleurs résultats en ce qui concerne l'analyse et l'extraction des connaissances pour le TAL.

Chapitre IV : Approche proposée

Nous présentons l'approche proposée pour la simulation des trois tâches ciblées par la thèse. Nous décrivons les étapes nécessaires pour analyser les exigences et spécifications textuelles en exploitant les techniques du TAL et l'apprentissage profond. Nous expliquerons les approches employées et adaptées pour la classification automatique des exigences en utilisant les caractéristiques communes de chaque catégorie, pour l'extraction des artéfacts et pour l'identification des correspondances à l'aide de la similarité sémantique.

Chapitre V : Évaluation de l'approche proposée

Nous évaluons les approches décrites dans le chapitre IV en appliquant des métriques, telles que la précision et le taux d'erreur. Nous identifions les mesures nécessaires pour évaluer les performances des modèles et les représentations appliquées.

Chapitre VI : Contributions et perspectives

Nous faisons ressortir les contributions de la thèse à l'ingénierie des exigences rédigées dans un langage naturel. Nous expliquons l'apport de notre approche pour faciliter l'établissement des correspondances dans un système d'information. Nous décrivons également les lignes directrices de questions de recherche pour les travaux futurs.

CHAPITRE II

INGÉNIERIE DES EXIGENCES ET TRAITEMENT AUTOMATIQUE DU LANGAGE

2.1 Introduction

Concevoir des systèmes d'information consiste à exécuter séquentiellement un ensemble de phases. Chaque phase prépare un contenu abstrait pour la suivante afin de livrer un produit ou logiciel final. Les parties prenantes et les utilisateurs amorcent le développement d'un système d'information en rédigeant des exigences sous forme de texte dans un langage naturel. Bozyi & Deniz (2016) ont décrit cette activité de rédaction comme l'étape qui initie les travaux de développement. La valeur de cette activité importante repose sur l'engagement des acteurs et utilisateurs, ayant différents profils, qui doivent viser un produit final qui répond aux besoins d'affaires.

Ensuite, les concepteurs analysent les exigences pour produire les artéfacts nécessaires au développement du système. Ils conçoivent des modèles abstraits et livrent des documents de spécification qui présentent ces modèles. Citons les artéfacts suivants :

- les cas d'utilisation,
- les processus d'affaires,
- les modèles conceptuels,
- les modèles logiques,
- les modèles physiques,

- les cas d'essai.

L'ingénierie du logiciel se caractérise par l'utilisation de modèles abstraits par les concepteurs (Casamayor, Godoy, & Campo, 2012b). Ces modèles aident à décrire les différents états du produit. Les spécifications couvrent les phases de conception, de développement, de test, de configuration et de maintenance. Chaque phase fournit un ensemble de biens livrables qui respectent des normes prédéfinies. Pour établir les correspondances entre les artéfacts, il y a une exploration documentaire sur la relation entre les exigences textuelles et les techniques de TAL.

Les exigences renvoient à une forme de rapport de nécessité reliée à des contraintes et à des conditions (ISO/IEC/IEEE 29148, 2011). L'analyse des exigences fait partie des activités les plus importantes dans le cycle de développement (Bozyi & Deniz, 2016). Son succès affecte directement les autres étapes du cycle. Des exigences bien documentées, claires, non ambigües, cohérentes et consistantes auront une incidence positive sur la qualité du produit final.

Le langage naturel constitue le langage parlé et écrit par les humains à des fins de communication et d'interaction (Liu, Li, & Thomas, 2017). Cette interaction comprend deux branches d'activités :

- la compréhension concerne tous les processus computationnels de transfert du langage naturel des humains vers un format interprétable par la machine;
- la génération permet de produire un texte dans un format compréhensible par les humains.

Le traitement automatique du langage (TAL) pour l'ingénierie des exigences vise à exploiter ces deux branches d'activités. Nous nous intéressons à la compréhension et à l'extraction automatiques des connaissances à partir de ce type de document. L'objectif de la thèse consiste à établir, à partir des exigences textuelles, les correspondances entre les artéfacts d'un système.

Dans ce chapitre, nous explorons les recherches qui ont étudié l'ingénierie des exigences supportée par les techniques de TAL. Pour cette exploration, nous nous intéressons aux différentes connaissances que nous pouvons extraire des documents. Nous distinguons deux types de connaissances :

- caractéristiques : elles représentent un ensemble de qualificatifs qui décrivent les exigences, par exemple :
    - les conflits et ambigüités : les exigences, par la nature du langage, peuvent présenter ce type de caractéristiques (exemple : un même mot peut avoir des sens différents selon le contexte et l'intention du rédacteur);
    - la classification : les exigences peuvent traiter les aspects fonctionnels ou non fonctionnels du système;
    - la traçabilité : les exigences peuvent contenir les liens entre les artéfacts du système;
- artéfacts : ce sont des éléments abstraits que les concepteurs produisent lors de l'exécution des différentes phases, notamment :
    - les cas d'utilisation et diagrammes d'activités;
    - les processus et modèles d'affaires;
    - les modèles conceptuels et diagrammes de classes;
    - les cas d'essai.

Dans ce chapitre, nous commençons par présenter une vue globale de la structure sur laquelle nous avons basé notre exploration des articles de recherche. Dans les sous-sections, nous passons en revue les travaux qui ont couvert les différentes caractéristiques et les artéfacts supportés par les techniques de TAL. Nous étudions les thèmes suivants :

- conflits et ambigüités;
- classification automatique;
- cas d'utilisations et diagrammes d'activités;

- modèles et processus d'affaires;
- modélisation conceptuelle et diagrammes de classes;
- cas d'essai;
- traçabilité.

Par la suite, nous explorons les recherches les plus récentes qui ont appliqué, de manière générale, les techniques de TAL. Nous donnons en exemple les réseaux de neurones et plus précisément l'apprentissage profond pour traiter le texte. À la fin de ce chapitre, nous évaluons la possibilité d'utiliser ces techniques dans notre thèse.

2.2 Vue globale

2.2.1 Structure

La figure 2.1 décrit la structure documentaire que nous avons explorée. Nous distinguons les connaissances en tant que caractéristiques ou en tant qu'artéfacts en prenant en considération leur type et les techniques appliquées.

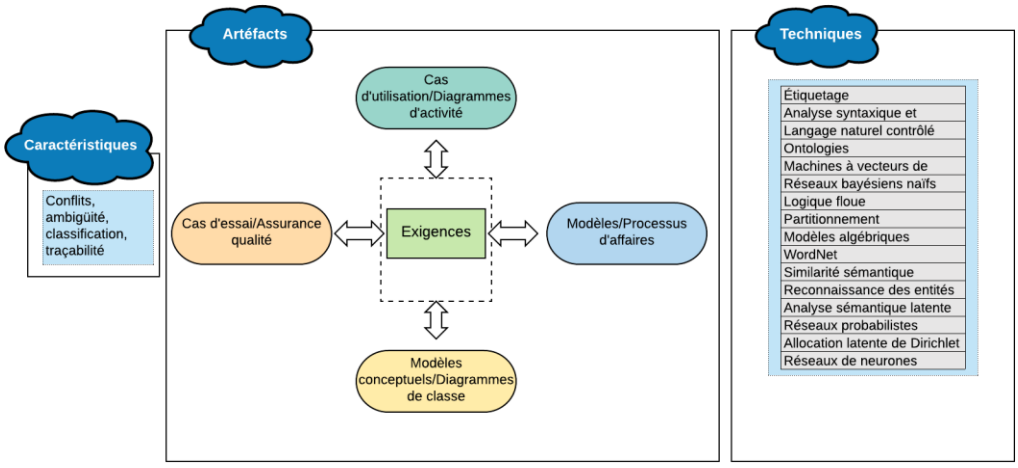


Figure 2.1 Structure de la thèse

2.2.2 Cartographie

Nous avons effectué notre revue de la littérature scientifique en ciblant un ensemble de thèmes. Nous avons traité chaque sujet en mettant l'accent sur les techniques utilisées. Le tableau 2.1 résume l'état de l'art sous forme d'une cartographie avec les caractéristiques suivantes :

- les lignes contiennent les techniques adoptées;
- les colonnes contiennent les sujets et les contextes d'utilisation considérés pour chaque recherche;
- les cellules indiquent l'existence de travaux liés à la combinaison technique/sujet.

Tableau 2.1 Cartographie documentaire

	Conflits et ambigüités	Classification automatique	Cas d'utilisation	Processus d'affaires	Modélisation conceptuelle	Cas d'essais	Traçabilité
Étiquetage morphosyntaxique	x	x	x	x	x	x	x
Analyse syntaxique et sémantique	x	x	x		x	x	x
Langage naturel contrôlé	x	x	x	x	x	x	
Ontologies	x	x	x		x		

	Conflits et ambigüités	Classification automatique	Cas d'utilisation	Processus d'affaires	Modélisation conceptuelle	Cas d'essais	Traçabilité
Machines à vecteurs de support	x	x					x
Réseaux bayésiens naïfs	x	x					
Logique floue	x						
Partionnement		x		x			
Modèles algébriques			x				

	Conflits et ambigüités	Classification automatique	Cas d'utilisation	Processus d'affaires	Modélisation conceptuelle	Cas d'essais	Traçabilité
WordNet				x	x		
Similarité sémantique				x			x
Reconnaissance des entités nommées					x		
Analyse sémantique latente							x

	Conflits et ambigüités	Classification automatique	Cas d'utilisation	Processus d'affaires	Modélisation conceptuelle	Cas d'essais	Traçabilité
Réseaux probabilistes							x
Allocation latente de Dirichlet							x
Réseaux de neurones	x	x					

Ce chapitre est structuré en fonction de l'objectif d'appliquer des techniques pour analyser les exigences et en extraire des modèles ou des artéfacts. Nous abordons les sujets d'intérêt suivants :

- conflits et ambigüités : détecter les conflits et les ambigüités;
- classification automatique : regrouper les exigences selon leur type;
- cas d'utilisation et diagrammes d'activités : générer les cas d'utilisation et diagrammes d'activités;
- processus d'affaires : extraire les processus d'affaires;
- modèles conceptuels et diagrammes de classes : générer les entités conceptuelles et les classes à partir du texte;
- cas d'essai et assurance qualité des logiciels : extraire les cas d'essai;
- traçabilité : établir la relation entre les artéfacts du système d'information.

2.3 Conflits et ambigüités des exigences

En collaboration avec les analystes d'affaires et les concepteurs, les utilisateurs rédigent les exigences dans un langage naturel. Les contributions de ces utilisateurs détenant des profils et expériences diversifiés peuvent comporter des difficultés : conflits, ambigüités et inconsistance de l'information contenue dans le texte rédigé. L'application d'une technique de TAL doit prendre en considération l'ambigüité du langage afin de détecter les anomalies, les erreurs de traitement et de compréhension. L'ambigüité dans le texte survient lorsque des phrases comportent plus qu'un sens (Sinpang, Sulaiman, & Idris, 2017). Tjong, Hartley, & Berry (2007) ont listé les règles qui permettent d'identifier ce type d'anomalie.

Les conflits dans les exigences représentent une interaction contradictoire et inattendue entre les phrases dans le texte (Kim, Park, Sugumaran, & Yang, 2007). En fonction de la source, nous distinguons deux types de conflits :

- conflits des activités qui surviennent quand des exigences différentes concernent une même action;
- conflits de ressources qui surviennent quand des fonctions, qui visent différentes cibles, utilisent en parallèle des ressources limitées.

Le langage naturel est le moyen le plus utilisé par les utilisateurs pour rédiger les exigences (Sinpang, Sulaiman, & Idris, 2017). Il facilite la spécification des services attendus d'un système particulier. Des exigences ambigües peuvent amener les concepteurs, analystes et développeurs à interpréter les besoins différemment. Le système sera alors conçu selon la compréhension des concepteurs et non selon les attentes des propriétaires et des utilisateurs.

Nous considérons six types d'ambigüité (Sinpang, Sulaiman, & Idris, 2017) :

- lexicale : un mot ou une expression avec des sens différents;
- syntaxique : un mot peut apparaître de différentes manières dans la syntaxe d'une phrase avec des sens différents, peu importe le contexte;
- sémantique : une même phrase avec plus qu'une interprétation possible;
- vague : une phrase qui ne peut être interprétée de manière précise;
- incomplétude : pas assez de détails fournis pour transmettre le sens dans une phrase avec une grammaire correcte;
- référentiel : crée de la confusion chez le lecteur à cause de références qui apparaissent cependant dans un contexte grammatical correct.

Shah & Jinwala (2015) ont identifié d'autres types d'ambigüités liées à l'ingénierie des exigences, telles que celles liées au domaine d'application, que ce soit un système ou du développement. Ils ont utilisé les techniques de TAL afin d'éliminer les ambigüités dans les exigences. Plusieurs auteurs (Ilieva, 2007; Fliedl & al., 2007; Kiyavitskaya, Zeni, Mich, & Berry, 2008; Nikora, Hayes, & Holbrook, 2010; Leopold, Eid-Sabbagh, Mendling, Azevedo, & Baiao, 2013; Landhäußer, Körner, & Tichy, 2014; Shah & Jinwala, 2015; Sinpang, Sulaiman, & Idris, 2017; Rosadini & al., 2017) ont appliqué l'étiquetage morphosyntaxique. Certains ont aussi adopté l'analyse syntaxique et

sémantique (Kim, Park, Sugumaran, & Yang, 2007; Ilieva, 2007; Tjong, Hartley, & Berry, 2007; Carlson & Laplante, 2014; Shah & Jinwala, 2015). D'autres auteurs ont choisi les langages naturels contrôlés (Ferreira & Da Silva, 2008, 2012).

De plus, les ontologies représentent un support pour maitriser les concepts du domaine (Velasco, Valencia-García, Fernández-Breis, & Toval, 2009). Des auteurs ont conçu des outils d'étiquetage et d'analyse tels que QTAG (Kop, Fliedl, & Mayr, 2010) et l'analyseur de Stanford (Arora, Sabetzadeh, Briand, & Zimmer, 2015). D'autres recherches ont testé l'apprentissage machine, notamment les réseaux bayésiens naïfs et les machines à vecteurs de support (*Support Vector Machine*) (Nikora, Hayes, & Holbrook, 2010) et la logique floue (Sinpang, Sulaiman, & Idris, 2017). Sinpang, Sulaiman & Idris (2017) ont mis en place une approche qui permet d'identifier les ambigüités en implémentant ce genre de techniques.

Shah & Jinwala (2015) ont développé une méthode qui marque chaque mot dans une phrase à l'aide de l'étiquetage morphosyntaxique, selon sa catégorie grammaticale (nom, verbe, adjectif...). Les auteurs ont présenté trois types de raisonnement. Le premier applique un ensemble de règles prédéfinies. Le deuxième utilise des formules statistiques obtenues par un grand volume de données d'apprentissage. Le troisième se caractérise par un raisonnement hybride qui combine les deux premiers types de raisonnement. Nikora, Hayes & Holbrook (2010) ont appliqué l'étiquetage morphosyntaxique pour évaluer des classificateurs selon les critères suivants : probabilité de détection, probabilité des détections fausses et précision. Cette étude a démontré que les réseaux bayésiens naïfs et l'optimisation minimale séquentielle des machines à vecteurs de support constituent les meilleurs classificateurs. Kiyavitskaya, Zeni, Mich & Berry (2008) ont détecté, par une approche semi-automatique, les ambigüités dans le texte écrit en langage naturel en appliquant l'étiquetage morphosyntaxique. Cette approche permet aux utilisateurs de comprendre le raisonnement derrière chaque résultat.

Carlson & Laplante (2014) ont développé un outil d'analyse et d'identification des ambigüités et des incomplétudes dans les exigences pour la NASA. L'analyse consiste à effectuer des calculs statistiques basés sur l'occurrence des mots à différents niveaux structuraux des documents. Rosadini & al. (2017) ont appliqué l'étiquetage morphosyntaxique et l'analyse syntaxique et sémantique qui s'appuient sur un ensemble de règles afin de détecter les anomalies dans le texte. L'approche permet de détecter les types d'anomalies suivants : ambigüité anaphore, ambigüité de coordination, terme vague, adverbe modal, voix passive, longueur excessive, terme indéfini, condition absente, unité de mesure ou référence manquante.

Ferreira & Da Silva (2008) ont utilisé d'autres techniques. Ils ont conçu un moteur Wiki pour supporter les activités liées à l'ingénierie des exigences. Cet engin applique un langage naturel contrôlé. Ce langage facilite l'analyse syntaxique et sémantique du texte pour détecter les anomalies, telles que les ambigüités et les conflits. Les auteurs (Nowakowski, Smialek, Ambroziewicz, & Straszak, 2013) ont proposé un langage basé sur les modèles pour un traitement efficace des connaissances du domaine. Les utilisateurs s'expriment à l'aide des contraintes des phrases rédigées. Les phrases sont divisées par noms et verbes, adjectifs et prépositions. Ce langage permet la formulation du vocabulaire pour décrire les caractéristiques du comportement pour un problème spécifique du domaine. Ferreira & Da Silva (2008) ont conçu un cadre qui, à l'aide de la détection des anomalies, permet de capturer les spécifications logiques du système en les rendant disponibles pour la transformation automatique vers le code. L'approche considère les cas d'utilisation comme des unités liées aux fonctionnalités du système. Chaque cas se compose de scénarios qui contiennent un ensemble de phrases rédigées dans un langage naturel contrôlé. D'autres chercheurs ont présenté une approche qui utilise un outil de correspondance des modèles (*Stanford NLP model matcher*) (Arora, Sabetzadeh, Briand, & Zimmer, 2015). Elle permet de vérifier et de valider la conformité d'une spécification par rapport à un modèle de document normalisé.

En résumé, les conflits et les ambigüités font partie des plus grandes difficultés lors de l'analyse des exigences rédigées en langage naturel. Les chercheurs ont utilisé l'étiquetage morphosyntaxique, l'analyse syntaxique et sémantique, les langages naturels contrôlés, les réseaux bayésiens naïfs et les machines à vecteurs de support pour détecter les anomalies du langage dans les exigences textuelles. La détection des conflits et des ambigüités a aussi facilité la classification automatique, la génération des modèles d'affaires, des modèles conceptuels et des cas d'essai. Avant d'établir les correspondances entre les artéfacts produits lors des phases de développement des systèmes, nous nous intéresserons à la détection des ambigüités, qui représente un grand enjeu pour l'analyse des documents.

2.4 Classification automatique

Classifier les exigences consiste à les distinguer selon le type, l'origine, la raison d'être et les objectifs (Ko, Park, Seo, & Choi, 2007). Ko, Park, Seo & Choi (2007) ont affirmé qu'à l'échelle mondiale, l'ingénierie des exigences prend de plus en plus de place en raison des besoins d'affaires exprimés par des utilisateurs distribués et délocalisés. Les exigences sont classifiées selon les caractéristiques suivantes : identification, priorité, criticité, faisabilité, risque, source et type. Cette classification requiert des efforts en matière de ressources et de temps. Les exigences peuvent être regroupées en exigences fonctionnelles qui décrivent les fonctionnalités du système. Les exigences fonctionnelles représentent le centre d'intérêt des analystes (Fauzi, Hassan, & Shah, 2017). Les exigences peuvent aussi décrire les attentes non fonctionnelles relatives à la performance, à la robustesse et à la disponibilité du système.

De plus, les recherches ont visé à classifier les exigences en utilisant les techniques de TAL. Divers auteurs ont appliqué l'étiquetage morphosyntaxique (Hussain, Kosseim, & Ormandjieva, 2008; MacDonell, Min, & Connor, 2014; Vlas & Robinson, 2012; Arellano, Carney, & Austin, 2015). Certains ont adopté un langage naturel contrôlé (Hussain, Kosseim, & Ormandjieva, 2008). D'autres ont construit des ontologies

(Velasco, Valencia-García, Fernández-Breis, & Toval, 2009; Vlas & Robinson, 2012; Arellano, Carney, & Austin, 2015). D'autres encore ont préconisé l'exploration et le partitionnement (*Clustering*) du texte (Hussain, Ormandjieva, & Kosseim, 2009). Certains ont utilisé les machines à vecteurs de support et les réseaux bayésiens (Casamayor, Godoy, & Campo, 2010; Ko, Park, Seo, & Choi, 2007). Ces derniers auteurs ont classifié les exigences en prenant en compte toutes les phrases du texte. Ils ont regroupé les phrases en catégories, ce qui permet une meilleure compréhension par les concepteurs et les développeurs. Chaque catégorie regroupe un ensemble de mots-clés utilisés par le classificateur. L'approche est basée sur une méthode de démarrage (*Bootstrap*) qui effectue des comparaisons de similarité à partir de l'échantillon des données disponibles relatives aux thèmes. Elle consiste à identifier une phrase barycentre qui sert à déterminer la meilleure représentation d'un thème. Les auteurs mesurent la similarité entre les mots et les phrases. La comparaison de similarité est effectuée entre la phrase à classifier et les phrases représentatives rédigées en anglais et en coréen.

D'ailleurs, Hussain, Kosseim & Ormandjieva (2008) ont automatisé le processus de détection des phrases pour les exigences non fonctionnelles. La classification applique un étiquetage morphosyntaxique du texte. Elle consiste à rechercher le radical morphologique en appliquant cinq fonctionnalités syntaxiques : nombre d'adjectifs, nombre d'adverbes, nombre d'adverbes qui modifient les verbes, cardinaux, nombre de degrés adjectif/adverbe. Le processus de classification des phrases qui correspondent aux exigences non fonctionnelles s'appuie sur un langage naturel contrôlé. Casamayor, Godoy & Campo (2010) ont détecté les exigences de type non fonctionnel en utilisant les techniques d'apprentissage semi-supervisées. L'approche consiste à effectuer un apprentissage à l'aide d'un nombre réduit d'exigences catégorisées manuellement par les analystes. Elle exploite une technique de rétroaction des utilisateurs pour améliorer les performances. L'approche s'appuie sur un

classificateur bayésien et un prétraitement avec des machines à vecteurs de support pour obtenir des représentations efficaces des documents.

Une autre recherche a consisté à développer une table de travail qui facilite l'automatisation du processus de mesure de la taille des exigences écrites dans un langage naturel libre (Hussain, Ormandjieva, & Kosseim, 2009). L'approche utilise la méthode du COSMIC (*Common Software Measurement International Consortium).* L'évaluation repose sur l'identification des processus et de leurs évènements déclencheurs. L'unité de mesure dans COSMIC est calulée à l'aide du mouvement de la donnée représentée par une composante de base fonctionnelle. En déplaçant les attributs des données qui appartiennent à un groupe spécifique, l'approche offre une perspective qui utilise l'exploration et le partitionnement du texte. L'exploration fait partie des techniques d'extraction qui classifient les informations textuelles en catégories basées sur des modèles statistiques appris. Le partitionnement aide à découper le document en catégories avec des propriétés et des modèles en commun. COSMIC permet d'extraire les processus fonctionnels et de classifier les mouvements des données en identifiant leurs groupes.

Certains chercheurs ont également opté pour les ontologies pour permettre de classifier les exigences. Une ontologie est une vue commune, réutilisable et partageable, qui capture les connaissances d'une communauté (Graber, 1995). Elle décrit les exigences en fournissant un moyen d'extraction des relations sémantiques. Afin de faciliter l'extraction des connaissances à partir des exigences de sécurité, Velasco, Valencia-García, Fernández-Breis, & Toval (2009) ont combiné les ontologies d'analyse du risque et des normes de sécurité. D'autres auteurs ont présenté la conception et la validation d'un classificateur des exigences relatif aux logiciels libres (Vlas & Robinson, 2012). L'outil permet d'identifier et de classifier les documents en les découpant en différents niveaux hiérarchiques. L'approche utilise un étiquetage morphosyntaxique avec une ontologie propre au domaine. Une autre recherche a consisté à interpréter et à organiser la gestion des exigences avec des ontologies

spécifiques aux domaines (Arellano, Carney, & Austin, 2015). Une telle approche utilise la bibliothèque logicielle de Python[1] pour le traitement automatique des langues NLTK (*Natural Language Toolkit*[2]). L'approche consiste à appliquer un étiquetage morphosyntaxique avant de reconnaitre les entités, les relations et les tuples.

MacDonell, Min & Connor (2014) ont conçu un prototype qui permet d'assister les analystes pour sélectionner et vérifier, à partir des exigences textuelles, les termes importants d'un projet. Le prototype comporte trois modules :

- un module qui applique l'étiquetage morphosyntaxique des phrases;
- un module qui effectue l'analyse syntaxique de chaque phrase à l'aide d'une charte technique;
- un module manuel qui aide l'utilisateur à sélectionner les termes importants en alimentant la base de connaissances.

Les auteurs ont proposé une approche basée sur les mots-clés pour extraire les exigences non fonctionnelles à partir des sources normalisées (Fauzi, Hassan, & Shah, 2017).

Nous considérons la classification des exigences comme une tâche importante pour traiter automatiquement des documents ou des paragraphes. Cette classification permet de valider les exigences selon une norme d'écriture prédéfinie. Établir les correspondances entre les artéfacts nécessitera un prétraitement ayant pour but l'identification des documents et leur classification en catégories. Nous nous intéressons aux catégories liées aux cas d'utilisation, aux processus d'affaires, aux entités conceptuelles ainsi qu'aux essai.

---

[1] https://www.python.org/

[2] https://www.nltk.org/

2.5 Cas d'utilisation et diagramme d'activité

La conception est un processus important dans le développement des systèmes d'information (Fauzi, Hassan, & Shah, 2017), car elle soutient le travail des développeurs. En effet, cette phase fournit une abstraction de la réalité prête à être intégrée dans les étapes suivantes. Les diagrammes UML représentent des composantes principales dans ce processus. Ils décrivent les fonctions d'affaires, les processus et les flux, les acteurs et les classes du système. Prenons par exemple les cas d'utilisation et les diagrammes d'activité. Ces diagrammes décrivent le comportement du système et font partie des modèles dynamiques de la modélisation UML. Des outils de conception et de modélisation, tels que IBM Rational Rose[3], Lucidchart[4] et Sparx Enterprise Architect[5], facilitent la conception des artéfacts du système et la rédaction des livrables.

Nous explorons ci-dessous les articles qui ont étudié les cas d'utilisation et les diagrammes d'activités dans le cadre de l'analyse des exigences à l'aide des techniques de TAL. La génération des modèles de comportement fait partie des travaux qui ont bénéficié de ces techniques. Plusieurs auteurs ont appliqué l'étiquetage morphosyntaxique et l'analyse syntaxique et sémantique (Ilieva, 2007; Sinha, Paradkar, Kumanan, & Boguraev, 2008; Zeng, 2008; Kamarudin, Sani, & Atan, 2015). D'autres ont combiné le langage naturel contrôlé avec les modèles algébriques (Cabral & Sampaio, 2008). Certains ont produit une représentation

[3] http://www-01.ibm.com/software/rational/uml/products/

[4] http://www.lucidchart.com

[5] http://www.sparxsystems.com

intermédiaire des exigences avec un modèle d'objet récursif (Zeng, 2008; Chen & Zeng, 2009).

Ilieva (2007) a généré les cas d'utilisation en appliquant l'étiquetage morphosyntaxique. La méthode s'appuie sur un ensemble de règles syntaxiques et sémantiques pour découper les phrases en sous-phrases et identifier les actions et les acteurs. Ces derniers étant les composantes principales, ils serviront à produire les cas d'utilisation. Cabral & Sampaio (2008) ont généré les modèles formels à des fins de vérification. Ces modèles aident à simplifier l'analyse et la conception, donc à réduire les couts. L'approche propose un langage naturel contrôlé, un modèle de spécification des cas et un outil qui supporte le processus de génération des modèles formels algébriques. Chaque modèle est structuré pour stocker les informations relatives à la traçabilité des exigences, une brève description ainsi que les interactions de l'acteur avec le système. Ensuite, à l'aide d'un processus séquentiel de communication CSP (*Communicating Sequential Processes*), Cabral & Sampaio (2008) ont traduit les cas d'utilisation en une spécification textuelle. CSP est un langage formel de la famille des théories mathématiques de concurrence qui décrit les modèles dans les systèmes concurrents. Pour de futurs travaux, Sinha, Paradkar, Kumanan, & Boguraev (2008) ont proposé d'adopter la même approche pour générer les cas d'essai et les cas d'utilisation. Ils ont appliqué leur méthode à quatre-vingts descriptions de cas d'utilisation universitaire et industrielle. Kamarudin, Sani, & Atan (2015) ont opté pour l'étiquetage morphosyntaxique contextuel et une analyse morphosyntaxique des phrases. Ils se sont aussi intéressés aux cas d'utilisation et aux diagrammes d'activités. Ces diagrammes symbolisent le résultat de la transformation des exigences. L'approche consiste à appliquer l'analyseur de Stanford pour l'étiquetage morphosyntaxique. L'acteur et le cas d'utilisation sont générés en les liant à l'aide de mots-clés. Les diagrammes d'activités sont produits en connectant chaque composante dans la bibliothèque des activités avec celles détectées dans les cas d'utilisation.

L'extraction des modèles à partir d'un langage naturel non contrôlé peut passer par une étape intermédiaire. Cette étape facilitera la représentation du texte pour la génération finale. De même, Zeng (2008) a proposé un nouveau langage graphique, décrit comme un modèle d'objet récursif (ROM, de *Recursive Object Model*), utilisé dans l'ingénierie des systèmes. Ce langage est doté de cinq symboles de base pour représenter un objet, un ensemble d'objets, une relation de contrainte, une relation de prédicat et une relation de connexion. Chaque objet dans l'univers s'attache à d'autres objets par des relations. Nous pouvons prendre en considération le mot, la phrase, les verbes et les adjectifs dans des modèles prédéfinis. Chaque modèle correspond à une représentation ROM. Cette représentation aide à capturer la sémantique d'un problème de conception et à générer les scénarios, les cas d'utilisation et les diagrammes de classes. D'autres travaux ont aussi utilisé la représentation ROM. Chen & Zeng (2009) ont transformé les exigences en un ensemble de diagrammes comportementaux. La transformation s'effectue en deux étapes. La première étape permet de convertir le langage naturel en un langage graphique intermédiaire (ROM). La deuxième étape permet de traduire les représentations ROM en modèles UML.

Les diagrammes UML contiennent les composantes de conception du système. Des outils de modélisation permettent de produire les diagrammes avec des schémas et des objets visuels accompagnés de descriptions textuelles. Les diagrammes sont intégrés, par la suite, dans des documents de spécification. Ces descriptions facilitent l'identification des fonctions principales et des acteurs du système. Pour établir les correspondances à partir des exigences, nous nous interrogeons sur la manière d'extraire ce type d'artéfact. En effet, le contenu des cas d'utilisation et des diagrammes d'activités permettra l'identification des fonctions principales du système et la découverte des liens entre les autres éléments.

2.6 Processus d'affaires

Aysolmaz, Leopold, Reijers & Demirörs (2018) ont souligné le rôle important des modèles de processus dans le contexte de la spécification des exigences relatives aux systèmes d'information. Ces modèles font partie des éléments nécessaires dans n'importe quel cycle de développement. Ces auteurs notent que les recherches relatives au traitement automatique des processus ont porté sur les sujets suivants :

- extraire les règles d'affaires à partir des modèles;
- assurer la validation à l'aide de la génération des langages naturels;
- générer les exigences fonctionnelles;
- établir les correspondances avec l'analyse des exigences.

La modélisation des processus d'affaires permet de documenter et de concevoir les opérations des entreprises ou organisations (Leopold, Eid-Sabbagh, Mendling, Azevedo, & Baiao, 2013). Ces modèles sont requis pour la conception et le développement des systèmes d'information à base de processus. Leur utilisation comporte une difficulté : leur validation est faite par des acteurs ayant des compétences diverses (Leopold, Mendling, & Polyvyanyy, 2014).

Nous énumérons ici les travaux qui ont adopté les techniques TAL pour générer ou extraire les modèles d'affaires à partir des exigences. De même, certains auteurs ont appliqué l'étiquetage morphosyntaxique (Leopold, Eid-Sabbagh, Mendling, Azevedo, & Baiao, 2013; Leopold, Mendling, & Polyvyanyy, 2014). D'autres ont traité les exigences à l'aide de la base WordNet[6] (Thayasivam, Verma, Kass, & Vasquez, 2011; Friedrich, Mendling, & Puhlmann, 2011; Leopold, Mendling, & Polyvyanyy, 2014). En outre, des travaux se sont appuyés sur les graphes et la similarité sémantique

[6] https://wordnet.princeton.edu/

(Thayasivam, Verma, Kass, & Vasquez, 2011). D'autres chercheurs ont supporté l'approche avec des modèles prédéfinis (Aysolmaz, Leopold, Reijers, & Demirörs, 2018). Hussain, Ormandjieva, & Kosseim (2009) ont identifié les processus fonctionnels ainsi que leurs évènements déclencheurs en utilisant l'exploration et le partitionnement du texte.

De plus, Leopold, Eid-Sabbagh, Mendling, Azevedo & Baiao (2013) se sont intéressés au problème de vérification linguistique relatif aux conventions de noms des éléments dans les modèles d'affaires. L'approche applique l'étiquetage morphosyntaxique. La vérification des noms et la détection des ambigüités s'effectuent en comparant les noms avec un corpus qui contient des modèles linguistiques valides.

D'autres auteurs ont généré les exigences textuelles à partir des modèles d'affaires (Leopold, Mendling, & Polyvyanyy, 2014). En comparant les modèles générés avec les descriptions de processus créées manuellement, les auteurs ont évalué, avec succès, la complétude de structure et la complexité linguistique du texte généré. L'évaluation du texte a démontré que l'approche facilite l'inférence de la sémantique à travers les modèles. L'approche s'appuie sur l'extraction des informations linguistiques, WordNet et l'étiquetage morphosyntaxique de Stanford. Friedrich, Mendling, & Puhlmann (2011) ont traité les processus d'affaires comme un outil important pour gérer les changements organisationnels et pour capturer les exigences logicielles. Ils ont généré automatiquement ces modèles à partir du langage naturel. L'approche combine l'analyseur de Stanford, WordNet et un mécanisme de résolution de l'anaphore pour analyser sémantiquement les phrases et les mots.

En outre, des chercheurs se sont intéressés à la problématique de l'inconsistance entre les modèles de processus et les exigences (Aysolmaz, Leopold, Reijers, & Demirörs, 2018). L'approche consiste à générer les exigences à partir des modèles de processus en utilisant des modèles prédéfinis. Elle permet d'extraire les phrases en distinguant trois types de connaissances :

- les phrases qui décrivent les rôles et les responsabilités dans l'exécution des activités;
- les phrases qui décrivent la manipulation des données des entités;
- les phrases qui touchent la spécification des contraintes.

Les modèles d'affaires représentent les activités relatives au processus du système. Établir les correspondances entre les processus d'affaires et les autres artéfacts du système fait partie des objectifs de la thèse.

2.7 Modélisation conceptuelle et diagramme de classes

La conception orientée objet (COO) s'appuie sur des caractéristiques et des comportements spécifiques (Bozyi & Deniz, 2016). L'application de la COO produit un ensemble d'artéfacts avec des significations et des objectifs précis. Cette conception est composée des classes et de leurs relations. Chaque classe comprend un ensemble d'attributs qui représentent les propriétés de l'objet. Les méthodes de classe représentent les types de comportements possibles sur l'objet et ses attributs. Les attributs et les relations fournissent l'information à propos de la sémantique (composition, agrégation, héritage, association, instanciation…).

D'ailleurs, des travaux ont été centrés sur l'extraction des modèles conceptuels à partir des exigences. Des auteurs ont appliqué l'étiquetage morphosyntaxique et l'analyse syntaxique et sémantique (Fliedl & al., 2007; Meziane, Athanasakis, & Ananiadou, 2008; Montes, Pacheco, Estrada, & Pastor, 2008; Zeng, 2008; Al-Safadi, 2009; Cabral & Sampaio, 2008; Hussain, Kosseim, & Ormandjieva, 2008; Kof & Penzenstadler, 2011; Herchi & Abdessalem, 2012; Casamayor, Godoy, & Campo, 2012a; More & Phalnikar, 2012; Arora, Sabetzadeh, Briand, & Zimmer, 2016). D'autres ont opté pour un langage naturel contrôlé (Meziane, Athanasakis, & Ananiadou, 2008; Ferreira & Da Silva, 2012). Certains chercheurs ont adopté le ROM (Zeng, 2008; Chen & Zeng, 2009). D'autres encore ont construit des ontologies (Ibrahim & Ahmad, 2010; Ghosh & al.,

2016). Des auteurs ont analysé les exigences à l'aide de la base lexicale WordNet (Hussain, Kosseim, & Ormandjieva, 2008; More & Phalnikar, 2012; Ghosh & al., 2016). D'autres chercheurs (Tichy & Koerner, 2010) ont choisi les rôles thématiques de Fillmore. Des auteurs ont conçu un vocabulaire sémantique des affaires (Afreena & Bajwa, 2011; Zhong, Zhang, Xie, & Mei, 2011). D'autres ont appliqué l'analyseur d'étiquetage morphosyntaxique de Stanford (Afreena & Bajwa, 2011; Ghosh & al., 2016; Bozyi & Deniz, 2016).

Casamayor, Godoy, & Campo (2012a) ont sélectionné le partitionnement du texte. Bozyi & Deniz (2016) ont utilisé l'enracinement de Lancaster. D'autres chercheurs ont présenté une approche qui consiste à reconnaitre les entités nommées pour extraire les modèles (Arora, Sabetzadeh, Briand, & Zimmer, 2016).

D'autres auteurs ont généré les modèles conceptuels à partir des exigences textuelles (Fliedl & al., 2007). Leur approche utilise l'étiquetage morphosyntaxique avec une analyse basée sur un gabarit préconçu, intermédiaire entre le texte et les modèles. Elle utilise un ensemble de règles afin d'extraire les concepts statiques et dynamiques. Montes, Pacheco, Estrada & Pastor (2008) ont identifié les éléments linguistiques qui correspondent aux éléments de base des modèles conceptuels orientés objet. Les éléments en question sont les classes, les méthodes, les relations et les propriétés. L'analyse syntaxique consiste à affecter des étiquettes au texte, à partir d'un ensemble de scénarios, et de le découper en phrases plus simples. L'analyse sémantique, elle, permet d'éliminer les ambigüités découvertes dans le texte, telles que la décomposition des noms et les périphrases verbales.

Al-Safadi (2009) a conçu un modèle semi-automatique pour la conception des bases de données. L'auteur s'est intéressé à l'étape d'analyse des exigences. L'approche consiste à appliquer un étiquetage morphosyntaxique combiné à une analyse syntaxique et sémantique. Elle établit une correspondance entre les noms et les entités, les verbes et les relations, les déterminants et les cardinalités, les adverbes avec l'unicité.

De plus, Bajwa, Samad, & Mumtaz (2009) ont construit un système pour convertir les exigences rédigées en anglais simple sous forme de paragraphes en modèles orientés objet. Les modèles sont ensuite convertis en code écrit dans un langage de programmation. Le texte est analysé sémantiquement pour extraire les classes et les objets ainsi que leurs attributs, leurs méthodes et leurs associations. Les diagrammes servent à générer le code automatiquement. L'approche utilise un engin de règles pour extraire les éléments de modélisation. Les auteurs ont appliqué l'étiquetage morphosyntaxique avec une analyse morphologique pour générer les diagrammes et le code Java[7] et VB.net[8].

Générer les classes à partir des exigences écrites en langage naturel représente une difficulté considérable (Ibrahim & Ahmad, 2010). Ibrahim & Ahmad (2010) ont développé un outil pour faciliter ce processus supporté par le TAL et l'ontologie du domaine. L'outil d'extraction des diagrammes de classes et d'analyse des exigences RACE (*Requirements Analysis and Class Diagram Extraction*) permet aux acteurs humains d'analyser les exigences textuelles, d'identifier les concepts/relations et d'extraire le diagramme de classes. L'approche utilise l'étiquetage morphosyntaxique de *OpenNLP*[9]. D'ailleurs, Kof & Penzenstadler (2011) ont effectué un apprentissage en temps réel. Leur méthode utilise l'étiquetage morphosyntaxique avec une traduction semi-automatique. L'utilisateur marque les séquences des mots et sélectionne les éléments du modèle (entité/relation). L'outil génère la trace entre les séquences et les modèles et prend en compte les décisions des utilisateurs pour les prochains traitements. De même, Herchi & Abdessalem (2012) ont appliqué l'étiquetage

---

[7] https://www.java.com/

[8] https://docs.microsoft.com/en-us/dotnet/visual-basic/

[9] https://opennlp.apache.org/

morphosyntaxique, une analyse morphologique et une première génération sous forme XML à l'aide de l'ontologie du domaine. Cette représentation XML facilite la visualisation du diagramme de classes. More & Phalnikar (2012) ont développé également une approche pour générer les diagrammes UML à partir des exigences. Ils ont adopté un outil d'analyse des exigences pour produire des diagrammes instantanés RAPID (*Requirement Analysis to Provide Instant Diagrams*) avec une ontologie spécifique au domaine. L'outil permet d'assister les analystes et les ingénieurs en facilitant l'extraction des concepts clés et de leurs relations. La méthode utilise une analyse lexicale et syntaxique ainsi que la base WordNet pour valider la sémantique des phrases générées lors de l'analyse syntaxique.

L'analyse des exigences fait partie des étapes nécessaires pour concevoir l'architecture d'un système d'information. Le problème majeur lié à cette tâche se trouve dans l'écart conceptuel entre les exigences initiales et les exigences du système décomposées en fonctionnalités. Casamayor, Godoy, & Campo (2012a) ont introduit une approche pour l'exploration et le partitionnement des fonctionnalités à partir des descriptions textuelles des exigences. Elle est composée des tâches suivantes :

- classifier les exigences fonctionnelles et non fonctionnelles;
- détecter les responsabilités d'architecture, les composantes et les liens entre les concepts.

Cette approche consiste à appliquer un étiquetage morphosyntaxique pour déterminer les actions et les activités, et un ensemble de règles pour déterminer l'existence d'une responsabilité candidate. Un apprentissage non supervisé a facilité le regroupement des responsabilités pour inférer leurs composantes respectives. Un analyste supervise le processus et y apporte les améliorations jugées nécessaires. L'approche permet d'obtenir un découpage fonctionnel des responsabilités du système qui permet la détection des éléments d'architecture.

Arora, Sabetzadeh, Briand, & Zimmer (2016) ont développé un extracteur de modèles du domaine en utilisant des règles du TAL combinées avec des règles d'extraction de l'information pour une meilleure adaptation et exploitation des analyseurs. L'approche utilise l'analyse syntaxique, l'étiquetage morphosyntaxique, la reconnaissance des entités nommées, un analyseur des dépendances, et un gestionnaire des coréférences. La génération s'applique aux concepts du domaine, association et généralisation ainsi qu'aux attributs et cardinalités. Landhäußer, Körner, & Tichy (2014) ont utilisé les spécifications du langage pour soutenir le processus de production logiciel à travers la génération automatique des modèles. L'objectif consiste à synchroniser les modèles, les spécifications et les analyses d'impacts. La génération concerne les diagrammes de classes, les diagrammes des états et les diagrammes d'activités. Landhäußer, Körner, & Tichy (2014) ont adopté une approche d'automatisation complète des exigences RECAA (*Requirements Engineering Complete Automation Approach*). Le processus intègre les changements dans les exigences des utilisateurs et s'appuie sur des outils qui offrent les options suivantes :

- détecter les anomalies linguistiques et permettre aux analystes de choisir la correction la plus appropriée;
- encoder la sémantique à l'aide des annotations;
- synchroniser les modèles avec les exigences textuelles.

Chen & Zeng (2009) ont proposé une nouvelle approche pour transformer les exigences textuelles en cas d'utilisation et en diagramme de classes. La transformation a lieu en deux étapes :

1. convertir le langage naturel en un langage graphique intermédiaire (ROM);
2. convertir le ROM en diagrammes UML.

Les exigences représentent les descriptions formelles du comportement souhaité et attendu d'un système complexe. Les utilisateurs les définissent selon leur perspective et leur rôle (clients, opérateurs, concepteurs et ingénieurs). Pour des fins de conception, de test et de vérification des systèmes critiques, nous pouvons analyser et transformer

automatiquement les exigences en modèles formels. Ghosh & al. (2016) ont développé l'outil d'extraction automatique ARSENAL (*Automatic Requirements Specification Extraction from Natural Language*) . Cet outil aide à transformer systématiquement les exigences, écrites en langage naturel, en modèles formels et en spécifications logiques. ARSENAL utilise l'étiquetage morphosyntaxique et l'analyseur des dépendances de Stanford, la base lexicale WordNet et une ontologie spécifique au domaine. L'outil produit un graphe de dépendance typé qui inclut l'étiquetage morphosyntaxique de tous les mots de toutes les phrases. Tichy & Koerner (2010) ont généré les modèles UML, le code et les cas d'essai à partir des exigences textuelles. Leur approche permet d'améliorer les spécifications textuelles approuvées par les clients et de construire un graphe sémantique basé sur les rôles thématiques de Fillmore. Elle produit les cas d'essai à partir des API du code.

L'extraction des connaissances peut aider à générer les modèles conceptuels. Des chercheurs ont développé AutoClass, qui fait partie de la famille des outils CASE (Bozyi & Deniz, 2016). Il permet d'extraire les diagrammes de classes et génère du code source C#[10] à partir des exigences textuelles. L'outil s'appuie sur un étiquetage morphosyntaxique de Stanford et de Lancatser combiné à des modèles à base de règles.

L'utilisation d'un vocabulaire commun comme moyen de communication et de conversion permet également d'extraire les diagrammes de classes. Afreena & Bajwa (2011) ont élaboré un vocabulaire associé à des règles sémantiques SBVR (*Semantic Business Vocabulary and Rules*) et un outil de conversion SBVR2UML. Ce vocabulaire adopte une logique d'ordre supérieur, ce qui rend facile son interprétation par une machine et sa compréhensibilité par les humains. L'utilisateur saisit les spécifications des exigences du système, le SBVR applique une analyse syntaxique et

[10] https://visualstudio.microsoft.com/vs/features/net-development/

sémantique pour extraire les informations orientées objet. L'outil SBVR2UML permet d'établir les correspondances entre les connaissances orientées objet obtenues et les modèles de classe. Ce vocabulaire se compose de types d'objets, de noms individuels, de concepts de verbe ainsi que des caractéristiques et des types de faits. L'approche utilise un prétraitement lexical et l'étiquetage morphosyntaxique de Stanford pour identifier les étiquettes de base. Le vocabulaire est extrait par la suite pour faciliter l'analyse et la conception du modèle de classes. L'évaluation de la méthode a montré de meilleurs résultats pour SVRB2UML.

De plus, Meziane, Athanasakis, & Ananiadou (2008) ont généré les spécifications en langage naturel à partir des diagrammes de classes UML. L'architecture du système repose sur le prétraitement et l'étiquetage morphosyntaxique. Les phrases sont associées à des étiquettes liées aux mots qui les composent. La génération se base sur les entités, les attributs et les opérations ainsi que sur les relations entre les objets (associations, agrégation et généralisation).

D'autres auteurs se sont orientés vers un langage naturel contrôlé. Ils ont décrit une approche sociotechnique qui adopte un type de langage pour résoudre les problèmes d'ambigüités (Ferreira & Da Silva, 2012). Elle consiste à intégrer l'ingénierie des exigences avec les processus des modèles (*model-driven*). Les exigences sont analysées à l'aide des modèles définis dans le langage (Zhong, Zhang, Xie, & Mei, 2011). Ils ont inféré les spécifications pour les ressources à partir de la documentation API en utilisant les modèles de Markov cachés.

Nous exposons les limitations de l'extraction des modèles à l'aide des techniques de TAL dans les trois points suivants (Arora, Sabetzadeh, Briand, & Zimmer, 2016) :

- les évidences et les preuves sont limitées par rapport à l'utilité des règles d'extraction au moment de leur application dans l'industrie;

- les règles d'extraction existantes n'exploitent pas adéquatement les dépendances du langage naturel détectées par les technologies de TAL modernes;
- une importante classe de règles développée par la communauté de l'extraction de l'information demeure inutilisée pour la construction des modèles.

Dans le cadre du développement des systèmes d'information, les modèles conceptuels et les diagrammes de classes représentent l'abstraction du monde réel. Ces modèles encapsulent la sémantique telle que dictée par les exigences et comprise par les concepteurs. La modélisation conceptuelle se définit comme la phase qui permet la conversion des exigences en artéfacts. Nous nous intéressons aux modèles conceptuels afin d'établir leurs correspondances avec les autres artéfacts du système.

2.8 Cas d'essai

Les cas d'essai représentent les essais réalisés pour chaque livraison du produit final. Chaque version livrée du produit doit passer par l'exécution de ces cas qui couvrent les fonctionnalités livrées (Yue, Ali, & Zhang, 2015). Yue, Ali, & Zhang (2015) ont utilisé les spécifications des cas d'essai comme moyen commun pour les documenter. Le concepteur d'essais doit comprendre ces spécifications pour assurer une bonne couverture des fonctionnalités. L'objectif consiste à évaluer la qualité d'une livraison en prenant en compte toutes les possibilités prévues par les exigences du système.

Des chercheurs ont combiné les langages naturels contrôlés avec l'étiquetage morphosyntaxique, l'analyse syntaxique et l'analyse sémantique (Carvalho & al., 2014; De Santiago & Vijaykumar, 2012; Yue, Ali, & Zhang, 2015). D'autres ont appliqué les rôles thématiques de Fillmore (Tichy & Koerner, 2010).
Le langage de spécification contrôlé est composé de modèles préconçus, d'un ensemble de règles de restriction. L'approche est mise en œuvre à l'aide d'un outil de génération et de production des cas d'essai basée sur une modélisation restreinte RTCM (*Restricted Test Case Modeling*). D'autres auteurs ont développé NAT2TEST pour générer les cas d'essai à partir des exigences textuelles (Carvalho & al., 2014). Cette approche utilise également un langage naturel contrôlé SysReq-*CNL* (*System Requirements Controled Natural Language*). Carvalho & al. (2014) ont appliqué une analyse morphologique, une analyse syntaxique, une correspondance sémantique, une analyse de discours et une analyse pragmatique. CNL transmet le vocabulaire du domaine d'application et classifie les entrées dans des catégories lexicales à l'aide de l'étiquetage morphosyntaxique. Par la suite, les exigences sont analysées afin d'obtenir un arbre syntaxique qui facilite l'extraction des cas d'essai.

Des chercheurs ont généré les cas d'essai à partir des exigences (De Santiago & Vijaykumar, 2012). Ils ont conçu un outil qui les traduit en modèles de transition d'états. L'outil utilise la conception combinatoire pour identifier les scénarios du

système et les tests d'acceptation. Cette conception requiert la définition d'un dictionnaire du domaine par un concepteur de test. L'approche applique l'étiquetage morphosyntaxique de Stanford, qui identifie les catégories lexicales de chaque phrase des exigences. Tichy & Koerner (2010) ont présenté un outil qui améliore les spécifications textuelles avec approbation des utilisateurs. Cet outil génère un graphe interne ainsi que les API du code en appliquant la sémantique définie par les rôles thématiques de Fillmore.

De plus, les langages contrôlés permettent de générer plus facilement les cas d'essai. Cependant, de tels langages requièrent un grand effort d'adaptation de la part des concepteurs. Ces derniers peuvent aussi utiliser les modèles de documents pour faciliter la rédaction des spécifications, ce qui fournit plus d'indices et facilitent la génération automatique.

Les utilisateurs évaluent et exécutent des tests d'acceptation des systèmes à l'aide des cas d'essai. Établir les liens avec les autres étapes du cycle de développement facilitera l'analyse d'impacts, à travers les phases, dans le cas de changements des spécifications. Ces correspondances permettront une meilleure visibilité pour les différents acteurs et aideront à explorer les liens entre les artéfacts dans les deux sens :

- explorer les exigences pour arriver au code final;
- explorer le code pour remonter aux exigences.

Nous considérons que retracer le lien entre les exigences et les cas d'essai est un critère de succès pour le système. Une exigence qui ne correspond à aucun cas d'essai peut indiquer son absence ou sa non-prise en compte par le cycle de développement.

2.9 Traçabilité

La traçabilité dans les systèmes d'information correspond à la possibilité de relier les artéfacts qui appartiennent aux phases du cycle de développement. Ainsi, des auteurs ont appliqué l'étiquetage morphosyntaxique et l'analyse syntaxique et sémantique (Arora, Sabetzadeh, Goknil, Briand, & Zimmer, 2015; Falessi, Cantone, & Canfora, 2013; Landhäußer, Körner, & Tichy, 2014). D'autres recherches se sont appuyées sur WordNet pour analyser les exigences (Thayasivam, Verma, Kass, & Vasquez, 2011). D'autres auteurs (Falessi, Cantone, & Canfora, 2013) ont adopté les machines à vecteurs de support. Des auteurs ont utilisé la similarité sémantique (Falessi, Cantone, & Canfora, 2013; Arora, Sabetzadeh, Goknil, Briand, & Zimmer, 2015; Hayes, Dekhtyar, & Sundaram, 2006; Thayasivam, Verma, Kass, & Vasquez, 2011). D'autres ont appliqué l'analyse sémantique latente (Lin & al., 2006; Falessi, Cantone, & Canfora, 2013). Parmi ces derniers, Falessi, Cantone & Canfora (2013) ont utilisé une base de thésaurus. Des auteurs ont développé une approche automatisée de l'ingénierie des exigences (RECCA) (Landhäußer, Körner, & Tichy, 2014). D'autres chercheurs ont conçu ARSENAL (Ghosh & al., 2016). D'autres travaux ont assuré la traçabilité à l'aide des mots-clés (Tsumaki & Morisawa, 2000).

La traçabilité a été un enjeu considérable depuis les années soixante-dix (Casamayor, Godoy, & Campo, 2012b). C'est pourquoi, Pierce (1978) a conçu un outil de traçabilité pour développer et maintenir une base de données des exigences. Son objectif consistait à faciliter l'analyse, la validation et la vérification du système. Cet outil utilise un seuil de correspondance des mots-clés, ce qui aide à extraire le texte et à affecter les mots-clés. L'approche permet d'assigner des mots spécifiques aux liens existants. La traçabilité, en utilisant la modélisation UML, fait partie des sujets abordés dans les recherches (Tsumaki & Morisawa, 2000). Tsumaki & Morisawa (2000) se sont concentrés sur des artéfacts tels que les cas d'utilisation, les diagrammes de classes et les diagrammes de séquence en prenant en considération les modèles d'affaires et

l'analyse conceptuelle. Afin de réduire l'effort requis par les tâches manuelles et la complexité de celles-ci, les concepteurs ont adopté des approches automatiques et semi-automatiques de l'acquisition intelligente de la traçabilité. D'autres chercheurs ont généré la traçabilité de l'information en produisant des dépendances entre les différents modèles et artéfacts du système (Egyed & Grünbacher, 2005). La génération repose sur les traces spécifiées précédemment par le concepteur. Elle facilite la création des dépendances difficilement identifiables manuellement.

L'identification des liens entre les artéfacts fait partie des problèmes classiques liés à l'extraction de l'information. Hayes, Dekhtyar, & Osborne, 2003; Hayes, Dekhtyar, & Sundaram (2005) ont généré les liens de traçabilité en calculant un score de similarité basé sur la fréquence et la distribution des termes dans un document textuel. Des outils tels que Poirot (Lin & al., 2006) et RETRO (Hayes, Dekhtyar, & Sundaram, 2006) ont intégré les modèles de raisonnement, tels que les machines à vecteurs de support, les réseaux probabilistes et l'indexation sémantique latente. D'autres chercheurs ont appliqué le modèle probabiliste et l'espace vectoriel dans deux cas d'utilisation pour établir les correspondances entre les exigences fonctionnelles et le code source écrits en C++ [11] et en Java (Antoniol, Canfora, Casazza, De Lucia, & Merlo, 2002). Marcus, Maletic, & Sergeyev (2005) ont ajouté l'indexation sémantique latente comme une extension du modèle vectoriel pour rétablir les liens entre la documentation et le code source. Des chercheurs ont utilisé l'allocation latente de Dirichlet pour un apprentissage automatique des thèmes sémantiques à partir des artéfacts (H.Asuncion, A.Asuncion, & Taylor, 2010).

Le problème d'extraction de la traçabilité réside dans la capacité à faire face à l'extraction des liens candidats tout en assurant de meilleurs précisions et rappels.

[11] http://www.cplusplus.com

L'amélioration de la performance d'extraction est effectuée à l'aide de trois stratégies différentes (Cleland-Huang, Settimi, Duan, & Zou, 2005) :

- modèle d'extraction probabiliste;
- modélisation hiérarchique;
- partitionnement logique des artéfacts.

Chen, Hosking, & Grundy (2011) ont amélioré la performance pour le rétablissement de la traçabilité entre le code source et les documents. D'autres ont analysé l'efficacité des machines à vecteurs de support à l'aide d'un thésaurus pour établir les correspondances entre les exigences, le code et les modèles UML (Settimi & al., 2004). Des chercheurs ont exploré l'utilisation du partitionnement pour améliorer la performance des traces à travers l'augmentation de la compréhension et la réduction des efforts humains pour évaluer l'ensemble des relations candidates (Duan & Cleland-Huang, 2007).

Établir les correspondances manuellement entre les artéfacts du même type ou de types différents est une tâche extrêmement difficile. Falessi, Cantone, & Canfora (2013) ont fourni des principes d'évaluation des techniques de TAL afin de soutenir les analystes lors de l'identification des liens. Dans leurs études, ils ont adopté ces techniques pour établir les traces entre les artéfacts à différents niveaux d'abstraction. Les techniques sont comparées à l'aide de critères tels que les objets à extraire, la sémantique des liens, les types de résultats et les approches.

Falessi, Cantone, & Canfora (2013) se sont intéressés aux techniques suivantes :

- l'application du modèle algébrique, tel que les bases de thésaurus, le modèle vectoriel et la sémantique latente;
- l'extraction des termes simples avec l'étiquetage morphosyntaxique;
- la pondération des termes et les métriques de similarité.

La traçabilité facilite également la tâche d'analyse d'impact dans les systèmes d'information. Arora, Sabetzadeh, Goknil, Briand, & Zimmer (2015) ont développé un outil qui permet d'effectuer cette tâche à travers les exigences textuelles. L'outil calcule les scores quantitatifs qui représentent le degré d'impact de chaque document. L'approche a pour objectif d'identifier les étiquettes des phrases dans les exigences et d'éliminer les déterminants, les pronoms et les prépositions. La similarité est calculée pour chaque paire de phrases ainsi que leur étiquette. L'approche permet d'utiliser la similarité syntaxique de Levenshtein et la similarité sémantique, qui capturent la relation sémantique définie dans un dictionnaire. Les scores les plus élevés représentent les relations fortes.

Lors de changements dans les spécifications, maintenir manuellement une consistance et une synchronisation entre les exigences et les modèles devient une tâche complexe. Landhäußer, Körner, & Tichy (2014) ont proposé une approche d'ingénierie de rétroaction (*Requirements engineering feedback*) qui automatise ce processus de maintien de la consistance entre le texte et les modèles.

Thayasivam, Verma, Kass, & Vasquez (2011) ont étudié la traçabilité entre des processus. Ils ont conçu un outil qui combine les techniques de TAL, l'extraction de l'information et le raisonnement sémantique pour établir automatiquement les correspondances avec les modèles de processus. Le modèle d'analyse des écarts ProcGap (*Process Model Requirements Gap Analyzer*) utilise un algorithme basé sur les éléments suivants :

- la similarité entre verbe, objet et objet prépositionnel (triplet VOP) : elle sert à produire un score relatif aux similarités sémantiques entre les VOP;
- la similarité cosinus modélise la phrase comme un sac de mots. Cette métrique ne dépend pas de la structure de la phrase et s'avère utile pour traiter les phrases complexes ou mal formées;
- le graphe sémantique et WordNet définissent les termes spécifiques du domaine.

Établir la traçabilité entre les artéfacts produits par les phases de développement d'un système est utile pour identifier les liens existants, ce qui facilite l'analyse d'impact et fournit aussi une meilleure compréhension des liens entre les composantes du système.

2.10 Réseaux de neurones et TAL

Des études récentes ont réalisé l'extraction des connaissances à partir du texte en utilisant les réseaux de neurones (RN) et l'apprentissage profond. Les auteurs ont utilisé ce type d'apprentissage, supervisé, non supervisé ou semi-supervisé, pour la classification des documents. Le traitement des images et la reconnaissance des formes ont été les premiers à bénéficier de ce type d'apprentissage. Dans d'autres études récentes, ces technologies ont été appliquées à des domaines différents, tels que l'exploration des données et du texte.

L'architecture de base d'un RN est complètement connectée et organisée en couches. La première couche représente l'entrée du réseau et la dernière couche représente la sortie ou le résultat. Toutes les autres couches au milieu sont cachées.

En raison de leur efficacité dans le traitement des images et la reconnaissance des formes, Gehrmann & al. (2018) se sont intéressés à leur application dans le domaine médical. Ils ont affirmé que les dernières avancées dans l'apprentissage profond et les techniques de TAL ont permis de produire des modèles pour apprendre une représentation riche du langage. Les RN à convolution (RNC) pour la classification du texte ont aidé à améliorer les techniques existantes. L'approche bénéficie de la représentation du langage pour apprendre la relation entre la phrase dans le texte et une condition médicale spécifique. Les auteurs ont comparé les performances des RNC avec d'autres méthodes classiques pour l'extraction des concepts. L'évaluation a démontré que les RNC dépassent les autres méthodes pour ce qui est des tâches exécutées, de la précision et de l'amélioration du F1-score. Ces deux mesures facilitent

l'évaluation de la performance des RN. La performance des classificateurs des documents textuels diminue avec l'augmentation du nombre de documents (Kowsari & al., 2017). Kowsari & al. (2017) ont appliqué la classification hiérarchique des documents, traitée comme un problème de classes multiples, en utilisant l'apprentissage profond hiérarchique HDLTex (*Hierarchical Deep Learning for Text Classification*). Cette méthode emploie un ensemble de modèles pour fournir une compréhension spécialisée pour chaque niveau de la hiérarchie.

Les techniques de TAL classiques ne sont pas adaptées pour les programmes qui contiennent des informations riches, structurées, explicites et complexes (Mou, Li, Zhang, Wang, & Jin, 2016). Mou, Li, Zhang, Wang, & Jin (2016) ont proposé l'application des RNC avec l'approche TBCNN (*Tree Based Convolutional Neural Network*). L'approche repose sur une architecture générique pour traiter les langages de programmation. La méthode a démontré son efficacité pour la classification des programmes par fonctionnalité ainsi que pour la détection des codes réutilisables.

Nous pouvons comprendre les classificateurs de texte en utilisant une approche visuelle basée sur les RNC et une représentation word2vec (Winkler & Vogelsang, 2017). Cette dernière permet de convertir les mots dans des vecteurs afin de faciliter l'application des formules mathématiques, au moment de l'apprentissage à l'aide d'un corpus.

Knight, Nenkova, & Rambow (2016) ont développé un réseau d'attention hiérarchique pour la classification des documents. Ce modèle se distingue par deux caractéristiques :

- une structure hiérarchique qui schématise la structure;
- deux mécanismes pour traiter les mots et les phrases.

Ororbia, Giles, & Reitter (2015) ont conçu une nouvelle architecture hybride pour la classification semi-supervisée du texte. Ils ont identifié une procédure ascendante et descendante pour l'apprentissage du réseau expert de Boltzmann.

Hughes, Li, Kotoulas, & Suzumura (2017) ont classifié automatiquement les documents cliniques en prenant en compte les phrases. La méthode utilise les RNC pour représenter les fonctionnalités complexes. Les auteurs ont démontré que nous pouvons utiliser les RNC pour représenter la sémantique des documents cliniques, ce qui facilite la classification sémantique de la phrase. Le RN multicouche permet de générer plus de fonctionnalités optimales pendant l'apprentissage pour représenter la sémantique des phrases analysées. Les modèles facilitent l'exécution des tâches alternatives comme la comparaison des textes et l'extraction des tâches. Les auteurs ont généralisé cette approche pour les paragraphes et les documents.

Lenc & Král (2016) ont traité la classification des documents rédigés en langue tchèque. Le prétraitement utilisé par les approches actuelles peut avoir des impacts négatifs, comme la perte d'information. La méthode évite ce prétraitement en utilisant les RN profonds, qui peuvent apprendre à partir de fonctionnalités simples. Les auteurs comparent deux réseaux différents pour la tâche de classification : un réseau perceptron multicouche et un réseau à convolution. Les expérimentations ont démontré que les deux réseaux performent mieux que les autres approches, avec une meilleure performance des RNC.

L'apprentissage des représentations se classe parmi les problèmes fondamentaux dans l'intelligence artificielle et plus particulièrement dans les techniques de TAL. La classification du texte fait partie des tâches les plus communes et dépend beaucoup des représentations apprises. C'est une technique utilisée dans l'analyse des sentiments, la classification et l'inférence du langage. Zhang & al. (2018) ont proposé un apprentissage renforcé pour construire les représentations des phrases. Cet apprentissage permet d'identifier les structures pertinentes sans aucune annotation explicite. Les auteurs ont formulé l'exploration des structures en considérant le sujet comme un problème de décision séquentielle. Les décisions précédentes affectent celles du futur.

Zhang & al. (2018) ont décrit deux modèles de représentation structurée de la mémoire longue à court terme LSTM (*Long Short Term Memory*) :

- l'information distillée permet de sélectionner les mots pertinents qui composent une phrase;
- la hiérarchie structurée permet de découvrir les structures de phrases et de construire la représentation de chaque phrase avec deux niveaux de mémoire à court terme.

Joulin, Grave, Bojanowski, & Mikolov (2016) ont exploré les algorithmes de classification des textes en comparant les classificateurs « Fastext », les RNC et les RNC très profonds. Fastext est un classificateur qui permet un apprentissage efficace pour la représentation des mots et la classification des phrases à partir des documents textes. En effectuant une évaluation pour les tâches de prédiction des étiquettes et l'analyse des sentiments, Fastext peut atteindre des millions de mots en moins de dix minutes avec un processeur multi-cœur.

La représentation du texte est considérée comme une tâche clé pour les applications de TAL (Li & al., 2018). L'objectif de l'article consiste à représenter numériquement les documents sous forme de texte non structuré afin de pouvoir les traiter d'une manière computationnelle et mathématique. La plupart des méthodes existantes tirent profit de l'apprentissage profond en produisant une représentation du langage. Cependant, ces modèles ne prennent pas en considération la problématique des ambigüités. Li & al. (2018) ont considéré la nécessité de consulter une base de connaissances externe pour une meilleure compréhension du texte. Ils ont proposé un nouveau cadre de vecteur de concept TCV (*Text Concept Vector*). Ce cadre utilise les RN et la base de connaissances pour produire une représentation de meilleure qualité.

Li & al., (2018) ont affirmé qu'un texte brut est représenté à l'aide d'un ensemble de concepts à travers une large taxonomie dans une base de connaissances. Les RN transforment le texte conceptualisé sous forme de vecteur qui encode l'information

sémantique et celle liée au concept à partir du texte original. Li & al. (2018) ont évalué que le raisonnement prend en considération la phrase et le document avec succès.

2.11 Travaux connexes

L'analyse et la manipulation des exigences textuelles permettent d'extraire des connaissances utiles selon des besoins spécifiques. L'objectif de la thèse consiste à extraire les correspondances entre les différents artéfacts/éléments du cycle de vie du développement des systèmes. Le modèle d'objet récursif ROM (*Recursive Object Model*) a été proposé comme langage intermédiaire (Zeng, 2008; Chen & Zeng, 2009). C'est un modèle qui permet de structurer les connaissances dans un format compréhensible par la machine. Nous pouvons prendre en considération le mot, la phrase, le verbe et l'adjectif dans des modèles prédéfinis et liés à une structure ROM. Ibrahim & Ahmad (2010) ont développé un outil d'analyse RACE (*Requirements Analysis and Class Diagram Extraction*) supporté par une ontologie du domaine. D'autres auteurs (More & Phalnikar, 2012) ont généré les diagrammes UML à partir des exigences. Ils ont utilisé une ontologie du domaine avec une analyse lexicale et syntaxique avec WordNet.

Landhäußer, Körner, & Tichy (2014) ont synchronisé les modèles, les spécifications et les analyses d'impacts. Cette synchronisation a pour objectif de supporter le processus de production logiciel à travers la génération automatique des diagrammes de classes, des diagrammes d'états et des diagrammes d'activité. Ghosh & al. (2016) ont développé le cadre ARSENAL, qui permet de transformer systématiquement les exigences textuelles en modèles formels et spécifications logiques. Thayasivam, Verma, Kass, & Vasquez (2011) ont combiné les techniques TAL, l'extraction de l'information et le raisonnement sémantique pour établir automatiquement les correspondances entre les exigences textuelles et les modèles des processus. Ils ont appliqué la similarité cosinus et les graphes sémantiques. Li & al. (2018) ont proposé un nouveau cadre, testé avec succès, qui utilise les RN et une base de connaissances

pour produire une représentation du texte de meilleure qualité. Ils ont conceptualisé le texte dans une base de connaissances à travers une large taxonomie. Gehrmann & al. (2018) ont utilisé les RNC pour la classification du texte. Ils ont évalué la pertinence des phrases liée à une condition médicale spécifique et ils ont démontré que les RNC performent mieux que les autres méthodes en termes de tâches exécutées, de précision et d'amélioration du score F1. Des chercheurs ont calculé les scores quantitatifs qui représentent le degré d'impact de chaque document (Arora, Sabetzadeh, Goknil, Briand, & Zimmer, 2015). Ils ont capturé la relation sémantique définie dans un dictionnaire en utilisant la similarité syntaxique de Levenshtein et la similarité sémantique. Ferrari, Spagnolo, & Genesis (2017) ont construit une base de documents d'apprentissage relative aux exigences textuelles.

2.12 Conclusion

L'ingénierie des exigences produit un ensemble d'artéfacts, de textes qui spécifient et décrivent les connaissances du système. Les analystes ont pour objectif de convertir les exigences exprimées par les propriétaires du système dans des documents de spécifications textuelles. Ces spécifications touchent les aspects d'architecture, de conception, de développement et des essais. Ces documents contiennent des descriptions et des caractéristiques relatives aux éléments pertinents pour chaque étape. Notre objectif consiste à analyser les spécifications afin de les classifier selon leur contenu. Par la suite, nous identifierons la liste des artéfacts par catégorie et nous établirons les correspondances entre ces artéfacts.

Dans ce chapitre, nous avons exploré différentes techniques pour analyser les exigences et en extraire les connaissances. Nous avons observé que la classification est une tâche nécessaire pour organiser les documents en vue de les préparer pour l'annotation et l'extraction. Kowsari & al. (2017) ont étudié l'utilisation de

l'exploration des données et l'apprentissage machine pour analyser le langage. Ils se sont intéressés aux machines à vecteur de support, qui sont plus difficiles à interpréter. Ils ont également étudié les arbres de décision et les réseaux bayésiens naïfs, qui sont plus faciles à comprendre et peuvent gérer la reformulation des requêtes. L'efficacité des RN dans la reconnaissance des images a encouragé les chercheurs à les utiliser pour analyser le langage naturel. Malgré le fait que ces techniques sont populaires auprès de la communauté du TAL, l'ingénierie des exigences n'en a pas encore bénéficié. De plus, nous explorons l'apprentissage profond en raison de son efficacité (Gehrmann & al., 2018; Kowsari & al., 2017). Le prochain chapitre complètera l'état de l'art en listant les avantages et les inconvénients des techniques d'apprentissage profond pour le TAL; cela nous amènera à choisir l'approche la plus adaptée pour notre thèse.

CHAPITRE III

ÉVALUATION DES MODÈLES DE L'APPRENTISSAGE PROFOND POUR LE TRAITEMENT AUTOMATIQUE DU LANGAGE

3.1 Introduction

Dans le chapitre précédent, nous avons étudié l'évolution des techniques de TAL pour l'ingénierie des exigences. En raison de son efficacité dans la reconnaissance des formes, nous nous intéressons à l'apprentissage profond en explorant les modèles utilisés pour des tâches cognitives relatives au texte. Nous donnons en exemple l'analyse des sentiments et la comparaison des mots. Nous nous appuyons sur les évaluations et les expérimentations des chercheurs cités dans le chapitre précédent pour choisir l'approche la plus adaptée pour la thèse.

Au fil des années, l'apprentissage profond a impacté positivement les techniques de TAL. Les auteurs ont proposé des pistes d'amélioration relatives aux représentations du texte pour faciliter le raisonnement pour des tâches spécifiques au langage.

Nous nous inspirons de ces méthodes afin de les appliquer aux tâches cognitives ciblées dans notre travail. À la fin de ce chapitre, nous pourrons présenter les lignes directrices de l'approche à utiliser dans notre thèse; nous pourrons ainsi accomplir les trois tâches suivantes :

- classifier les exigences en quatre catégories de documents : architecture, conception, processus et essais;
- extraire les artéfacts et éléments pertinents par catégorie;
- établir les correspondances entre les éléments extraits.

Ces trois tâches font partie du domaine d'extraction des associations et de la sémantique à partir du langage. Kapetanios, Alshahrani, Angelopoulou, & Baldwin (2018) ont décrit les algorithmes qui traitent ce type de tâches. Ces algorithmes convertissent les mots en vecteurs qui représentent leurs occurrences dans un contexte spécifique. Kapetanios, Alshahrani, Angelopoulou, & Baldwin (2018) se sont intéressés aux deux tâches suivantes :

- extraire la sémantique des mots à l'aide d'un modèle d'espace vectoriel de haute dimension;
- traiter les associations des mots en utilisant leur distance sémantique.

Kapetanios, Alshahrani, Angelopoulou & Baldwin (2018) ont proposé des méthodes qui améliorent la représentation du texte pour entrainer des modèles afin d'extraire les associations des mots et de calculer la similarité sémantique. Ils ont illustré les contextes et listé les limitations de ces algorithmes. Ils ont également fourni un aperçu des approches les plus adoptées par les techniques de TAL. Ces algorithmes ont en commun deux principes (Kapetanios, Alshahrani, Angelopoulou, & Baldwin, 2018) :

- le contexte détermine le sens lexical des mots;
- la transformation des mots en représentation numérique facilite le calcul de la similarité et l'application des opérations mathématiques.

Nous structurons ce chapitre en trois sections : la première fournit une vue sur les algorithmes et les approches; la deuxième décrit l'évolution de l'apprentissage profond pour le TAL; et la troisième présente les résultats des évaluations de ces approches.

3.2 Approches

Les chercheurs ont revisité le flux principal des approches en évaluant la nature des associations des mots qui appliquent des algorithmes de l'apprentissage machine. En effet, Kapetanios, Alshahrani, Angelopoulou, & Baldwin (2018) ont affirmé que le TAL se caractérise par l'extraction des relations entre les mots en employant des méthodes statistiques telles que les mesures de cooccurrence. Ils ont également présenté des modèles typiques relatifs à l'extraction des associations des mots et au calcul des similarités sémantiques. Ils se sont penchés sur ces approches pour deux raisons principales :

- l'existence de plusieurs travaux et recherches qui ont mis en place des algorithmes qui facilitent leur adaptation pour des cas réels;
- l'existence de liens entre l'analyse des données et l'apprentissage profond.

Nous passerons maintenant en revue les approches basées sur la mémoire, la sémantique distributionnelle, l'analyse sémantique latente, l'allocation latente de Dirichlet, word2vec, doc2vec et les vecteurs globaux.

3.2.1 Approches basées sur la mémoire

Les approches probabilistes consistent à analyser les mots, réputés similaires, qui ont une grande probabilité de substitution (Kapetanios, Alshahrani, Angelopoulou, & Baldwin, 2018). Afin d'interpréter une phrase, le traitement implique la comparaison de l'extraction de fragments avec la phrase à interpréter. À l'aide d'une version bayésienne de la théorie de concaténation, le calcul de la probabilité de substitution permet de prendre chaque fragment qui appartient à une classe et de le comparer avec tous les fragments de la même classe. C'est une opération qui peut générer des traitements supplémentaires pour les corpus de grande taille. Ces derniers sont découpés en classes afin de réduire leur taille et de minimiser le cout de comparaison. Les mots les plus fréquents constituent les frontières d'un fragment. Ces frontières

représentent une séquence de mots entourés par d'autres mots (Kapetanios, Alshahrani, Angelopoulou, & Baldwin, 2018). Pour illustrer ce modèle, les auteurs donnent l'exemple suivant : « *THE book showed A picture OF THE author carrying A copy OF THE manuscript*. »

Ils ont découpé la phrase en cinq fragments :

1. THE book showed A;
2. A picture OF THE;
3. OF THE author carrying A;
4. A copy OF THE;
5. OF THE manuscript.

Pour calculer la probabilité de substitution, ils ont considéré les classes équivalentes suivantes (Kapetanios, Alshahrani, Angelopoulou, & Baldwin, 2018) :

- classe 1 : A **copy** OF THE, A description OF THE, A side OF THE;
- classe 2 : ONTO THE **copy**, ONTO THE table.

La similarité entre les mots « *copy* » et « *picture* » se calcule en utilisant la probabilité de substitution dans les deux classes :

- classe 1 : une probabilité $P1 = 1/3$;
- classe 2 : une probabilité $P2 = 1/2$.

La probabilité de substitution : P (« *picture*, *copy* ») = $(P1 + P2)/2 = 0{,}415$.

3.2.2 Sémantique distributionnelle

Les informations contextuelles fournissent une bonne approximation du sens d'un mot (Kapetanios, Alshahrani, Angelopoulou, & Baldwin, 2018). D'un point de vue sémantique, les mots similaires partagent les mêmes distributions contextuelles. Ces modèles utilisent les vecteurs qui sauvegardent le contexte.

Redington, Crater, & Finch (1998) ont présenté le modèle adjacent regroupé en accumulant la fréquence des mots qui apparaissent dans les deux positions avant et les deux positions après. Ils l'ont illustré à l'aide de l'exemple suivant :

1. Found a picture of the;
2. Found a picture in her;
3. A pretty picture of her;
1. Found a copy of a;
2. Found a copy below the;
3. Destroyed the copy of the.

Le tableau 3.1 montre la représentation du texte à l'aide des vecteurs pour les mots qui occupent les positions -2, -1, 1 et 2.

Tableau 3.1 Vecteurs de fréquence des mots (Redington, Crater, & Finch, 1998)

picture	2	0	1	1	2	0	2	1	0	1	2	0
copy	2	1	0	0	2	1	2	0	1	2	0	1
	found	destroyed	a	pretty	a	the	of	in	below	the	her	a
	Pos -2			Pos -1			Pos 1			Pos 2		

La représentation du vecteur du mot « *copy* » est [2, 1, 0, 0, 2, 1, 2, 0, 1, 2, 0, 1].

3.2.3 Analyse sémantique latente

L'agrégation des contextes d'apparition des mots fournit des contraintes mutuelles qui déterminent la similarité entre les mots (Kapetanios, Alshahrani, Angelopoulou, & Baldwin, 2018). L'analyse sémantique latente (ASL) représente les mots sous forme de phrases et de paragraphes en utilisant un grand espace sémantique. C'est une approche liée aux réseaux de neurones, elle implique une valeur singulière de décomposition. La similarité estimée par l'ASL représente les fréquences de contigüité,

le compteur de cooccurrences et les corrélations. C'est une similarité qui dépend aussi d'une analyse mathématique robuste, capable d'inférer des relations plus profondes. L'analyse sémantique latente se définit comme une technique, mathématique et statistique, d'extraction et d'inférence des relations du contexte dans une partie de discours (Landauer, Foltz, & Laham, 1998). C'est une analyse indépendante des dictionnaires humains, de base de connaissances, des réseaux sémantiques, de grammaires, de syntaxes ou de morphologie. L'ASL consiste à représenter le texte dans une matrice. Les lignes contiennent les mots uniques dans le texte, les colonnes contiennent un passage du texte, un paragraphe ou un document. Les cellules contiennent la fréquence des mots par document. Chaque fréquence utilise une pondération à l'aide d'une fonction qui exprime l'importance du mot dans un texte particulier et le degré d'importance de l'information transportée par le domaine de discours.

Voici un exemple (Landauer, Foltz, & Laham, 1998) :

C1 : Human machine interface or ABC computer application;

C2: A survey of user opinion of computer system response time;

C3: The EPRS user interface management system;

C4: System and human system engineering testing of EPS;

C5: Relation of user perceived response time to error measurement.

Le tableau 3.2 montre la représentation des phrases de l'exemple à l'aide de l'ASL.

Tableau 3.2 Représentation ASL du texte de l'exemple (Landauer, Foltz, & Laham, 1998)

	C1	C2	C3	C4	C5
Human	1	0	0	1	0
Interface	1	1	0	0	0
Computer	1	1	0	0	0
User	0	1	1	0	1
System	0	1	1	2	0
Response	0	1	1	2	0
Time	0	1	0	0	1
EPS	0	0	1	1	0
Survey	0	1	0	0	0

3.2.4 Allocation latente de Dirichlet

L'allocation latente de Dirichlet (ALD) est un modèle génératif probabiliste qui permet le forage du texte et l'extraction de l'information. L'ALD s'intéresse à l'association d'un document à un thème (Kapetanios, Alshahrani, Angelopoulou, & Baldwin, 2018). C'est un modèle bayésien à trois niveaux (Blei, Ng, & Jordan, 2003). Chaque niveau facilite la modélisation des éléments à travers des thèmes. Chaque thème se caractérise par des probabilités relatives. Cette probabilité fournit une représentation explicite du document.

Prenons l'exemple des phrases suivantes (Kapetanios, Alshahrani, Angelopoulou, & Baldwin, 2018) :

- *I like to eat broccoli and bananas;*
- *I ate a banana and spinach smoothie for breakfast;*
- *Chinchillas and kittens are cute;*
- *My sister adopted a kitten yesterday;*
- *Look at this cute hamster munching on a piece of broccoli.*

L'allocation de probabilité par thème s'effectue comme suit :

phrases 1 et 2 : 100 % pour le thème A (food);
phrases 3 et 4 : 100 % pour le thème B (cute animals);
phrases 5 : 60 % pour le thème A, 40 % pour le thème B;
thème A : 30 % broccoli, 15 % bananas, 10 % breakfast, 10 % munching;
thème B : 20 % chinchillas, 20 % kittens, 20 % cute, 15 % hamster.

Du point de vue de l'apprentissage machine, l'algorithme ALD effectue des affectations initiales. Le fonctionnement permet de parcourir chaque document et d'attribuer un thème à chaque mot. Dans le but d'améliorer les affectations initiales, l'ALD permet de calculer deux métriques :

- la proportion des mots assignés à un thème;
- la proportion des correspondances à un thème pour tous les documents.

Le grand inconvénient de l'ALD est l'absence de métriques qui permettent d'évaluer la performance et qui facilitent l'évaluation de la meilleure affectation (Kapetanios, Alshahrani, Angelopoulou, & Baldwin, 2018). L'ALD utilise l'approche du sac de mots sans prendre en considération le sens lexical du texte.

3.2.5 Word2vec

Word2vec[12] utilise des réseaux de neurones à deux couches pour traiter du texte. C'est une représentation qui produit des vecteurs en acceptant un corpus en entrée. Les vecteurs représentent les fonctionnalités des mots dans le corpus. Word2vec convertit le texte en format numérique compréhensible par les réseaux profonds. C'est un modèle de prédiction efficace qui peut apprendre le plongement lexical des mots (*Word Embedding*) à partir d'un texte brut.

[12] https://radimrehurek.com/gensim/models/word2vec.html

Word2vec peut prédire le prochain mot « mat » pour la phrase « *The cat was sitting on the…* » (Kapetanios, Alshahrani, Angelopoulou, & Baldwin, 2018). Le modèle peut deviner, avec une grande précision, le sens d'un mot en utilisant ses anciennes occurrences. Ce modèle permet aussi d'identifier les associations entre les mots. Il offre la possibilité de partitionner les documents et de les classer par sujet.

Word2vec produit un vocabulaire dans lequel chaque élément correspond à un vecteur. Ce vecteur peut alimenter un réseau de neurones profond ou tout simplement détecter les relations entre les mots. Il consiste à calculer la similarité, entre les mots, comme base de l'apprentissage, à l'aide de la similarité cosinus. Une similarité zéro est exprimée avec un angle de 90 degrés, alors qu'une similarité totale est exprimée avec un angle de 0 degré (Kapetanios, Alshahrani, Angelopoulou, & Baldwin, 2018).

Word2vec offre deux types de représentations (Mikolov, Chen, Corrado, & Dean, 2013) :

- sac de mots continu (cbow): c'est un modèle qui prédit les prochains mots à partir du contexte. Il traite tout le corpus comme une seule observation, ce qui le rend utile pour les textes de petite taille;
- Skip-Gram continu : c'est un modèle qui prédit les mots du contexte à partir des mots cibles. Il traite chaque paire de contexte cible comme une nouvelle observation, ce qui le rend meilleur pour les textes de grande taille.

La représentation des mots par des vecteurs facilite l'application des opérations mathématiques; ces vecteurs permettent d'identifier des régularités linguistiques. Citons l'exemple de régularité suivant : si nous considérons l'énoncé vecteur (« Paris »), vecteur (« France ») + vecteur (« Italie ») alors, nous obtiendrons un vecteur proche du vecteur (« Rome »).

Word2vec est défini principalement comme un modèle de prédiction du contexte. En utilisant une représentation de vecteurs, il permet l'optimisation de la fonction objective (*Loss*) afin d'améliorer la prédiction du prochain mot dans un contexte spécifique (Kapetanios, Alshahrani, Angelopoulou, & Baldwin, 2018).

3.2.6 Doc2vec

Le & Mikolov (2014) se sont inspirés de word2vec pour définir des vecteurs de paragraphes (doc2vec). De la même manière que word2vec, le vecteur de paragraphes correspond à un vecteur unique qui capte leur sémantique. La matrice contient des colonnes pour le mot et pour le paragraphe. Le vecteur de paragraphes représente l'information manquante à partir du contexte courant et il peut agir comme une mémoire pendant le traitement.

Nous distinguons les deux approches doc2vec[13] suivantes :

- le sac de mots distribués Dbow (*Distributed bag of word*) utilise un fonctionnement semblable au Skip-Gram, à l'exception de l'étiquette d'entrée, qui représente le document à la place du mot. Ce type d'architecture ignore l'ordre des mots, d'où le nom sac de mots distribués;
- la mémoire distribuée Dmpv (*Distributed memory for paragraph vector*) emploie un fonctionnement analogue au sac de mots continu avec les exceptions suivantes. Pour l'entrée, ce modèle introduit une étiquette de document supplémentaire en plus des mots multiples cibles. À l'inverse du sac de mots, il consiste à concaténer les vecteurs au lieu de les additionner. L'approche a pour objectif de prédire le contexte du mot en utilisant en entrée le vecteur du document et les vecteurs des mots.

3.2.7 Vecteurs globaux

Les vecteurs globaux (GloVe) fonctionnent d'après une approche semblable à word2vec (Pennington, Socher, & Manning, 2014). C'est un algorithme d'apprentissage non supervisé qui permet d'obtenir des représentations de vecteurs de mots. Il diffère de word2vec par le fait que l'apprentissage s'effectue en utilisant une

[13] https://radimrehurek.com/gensim/models/doc2vec.html

cooccurrence globale des mots agrégés, à partir d'un corpus. GloVe effectue son apprentissage en réduisant la dimensionnalité de la matrice des cooccurrences. Le calcul de la probabilité de cooccurrence lors d'une expérience[14] avec un corpus de plusieurs millions de mots a démontré que les mots similaires apparaissent souvent ensemble. Le résultat de l'exemple a montré que le mot « *ice* » a une cooccurence plus élevée avec le mot « *solid* » qu'avec le mot « gaz », tandis que « *steam* » apparait fréquemment avec « *gaz* » et « *solid* ». Autant « ice » que « steam » apparaissent avec « *water* ». Nous pouvons donc considérer « *water* » comme une propriété commune.

3.3 Évolution de l'apprentissage profond pour le TAL

Geoffrey Hinton[15] est le pionnier de l'apprentissage profond utilisé pour classifier les images (Kapetanios, Alshahrani, Angelopoulou, & Baldwin, 2018). Kapetanios, Alshahrani, Angelopoulou, & Baldwin (2018) ont affirmé que cette première utilisation a encouragé son adaptation pour l'analyse du texte, des génomes et d'autres données multidimensionnelles. Les chercheurs ont adopté word2vec pour mesurer la similarité sémantique entre les mots à l'aide des architectures d'apprentissage profond. Sans aucune intervention humaine, les architectures des réseaux de neurones permettent à la machine d'apprendre, selon différents niveaux d'abstraction, les connaissances nécessaires pour effectuer une tâche cognitive spécifique. Ces techniques ont connu de grands succès pour les TAL, notamment dans les tâches suivantes (Kapetanios, Alshahrani, Angelopoulou, & Baldwin, 2018) :

- modélisation des phrases;
- catégorisation du texte;

---

[14] https://nlp.stanford.edu/projects/glove/

[15] https://en.wikipedia.org/wiki/Geoffrey_Hinton

- étiquetage des rôles sémantiques;
- reconnaissance d'entités nommées;
- réponses aux questions;
- analyses des opinions;
- traduction automatique.

Vodrahalli (2015) a exploré les possibilités offertes par l'apprentissage profond pour le TAL. Il a étudié l'évolution de cette pratique en définissant le TAL comme un ensemble de tâches séquentielles qui permettent l'extraction de la sémantique à partir du texte. Il a listé les tâches les plus populaires :

- prédiction du prochain mot dans un contexte donné;
- similarité et désambigüisation des mots;
- analogie et réponse aux questions.

D'autres auteurs ont introduit les modèles probabilistes (Bengio, Ducharme, Vincent, & Jauvin, 2003). Ils ont utilisé les réseaux de neurones avec propagation avant. Ces modèles facilitent l'apprentissage simultané à l'aide d'une représentation qui combine un vecteur de mots et un modèle statistique. Ils ont exploré les probabilités des séquences des mots. L'objectif consiste à remédier à la malédiction de la dimension. Cette malédiction se caractérise par une différence entre la séquence des données de test et celle utilisée pendant l'apprentissage. Traditionnellement, avec les approches de N-Gram, nous pouvons obtenir une généralisation en concaténant des petites séquences chevauchées et observées dans le jeu de données d'apprentissage.

L'approche probabiliste a permis de retirer de chaque phrase des informations sur un nombre exponentiel de phrases ayant une proximité sémantique (Bengio, Ducharme, Vincent, & Jauvin, 2003). Le modèle permet d'apprendre simultanément deux éléments :

- une représentation distribuée pour chaque mot;
- une fonction probabiliste pour les séquences de mots exprimées par la représentation distribuée.

Bengio, Ducharme, Vincent, & Jauvin (2003) ont affirmé que le modèle probabiliste permet de mesurer la similarité entre les phrases selon leur proximité dans le texte. Les séquences qui contiennent plus de mots similaires à ceux déjà vus auront la plus grande probabilité. La généralisation s'effectue à l'aide de séquences semblables à celles observées pendant l'apprentissage.

Mikolov, Karafiát, Burget, Černocký, & Khudanpur (2010) ont amélioré les représentations distribuées des mots pour le TAL. Ils ont affirmé que le modèle probabiliste limite le contexte des réseaux de neurones avec propagation avant. Afin d'y remédier, ils ont proposé l'utilisation des réseaux de neurones récurrents (RNR). Ces réseaux gèrent le passé en assignant une taille illimitée aux connexions. L'apprentissage consiste à compresser le passé dans un espace de dimension réduite. Les RNR peuvent former des termes avec une mémoire courte en offrant une meilleure gestion de l'invariance de la position.

De plus, la représentation distribuée des mots comporte un enjeu d'évaluation et d'efficacité. Mikolov, Chen, Corrado, & Dean (2013) ont étudié ce volet en proposant deux nouvelles architectures : le sac de mots continu (cbow) et le Skip-Gram continu. Ces modèles permettent de mesurer la qualité des représentations à l'aide de la similarité entre les mots. Mikolov, Chen, Corrado, & Dean (2013) ont affirmé que la similarité ne se limite pas à une simple régularité syntaxique. Ils ont prouvé que l'expression « vecteur (*King*) — vecteur (*man*) — vecteur (*woman*) » donne un vecteur proche du vecteur (*Queen*). Ils ont travaillé à la maximisation de la précision des

vecteurs en développant des architectures qui préservent les régularités linéaires dans les mots. Ils ont conçu un ensemble de tests compréhensibles pour mesurer les régularités syntaxiques et sémantiques

Mikolov, Joulin, Chopra, Mathieu, & Ranzato (2015) ont affirmé que les RNR facilitent l'apprentissage des données séquentielles, mais la taille réduite de la mémoire est une limitation. Ils ont remédié à cette limitation en proposant une approche qui s'appuie sur une mémoire avec une grande capacité de stockage. Ils ont démontré que l'apprentissage à partir d'exemples réels, dans un langage naturel, s'effectue en appliquant l'algorithme du gradient et une architecture modifiée d'un RNR simple. La modification consiste à implémenter quelques couches cachées qui changent leur état lentement en mettant une partie du poids de la matrice proche de l'identité. De plus, les couches cachées, qui changent rapidement, s'adaptent aux modèles à court terme. Le modèle peut ainsi être mis à jour lentement dans un contexte qui requiert la rétention des informations à long terme.

Étant donné l'augmentation de la taille du texte à analyser, Mikolov, Sutskever, Chen, Corrado, & Dean (2013) ont étudié les représentations distribuées des mots et des phrases en mettant en valeur leur composition. Ils ont affirmé que les modèles Skip-Gram sont inefficaces pour apprendre des représentations vectorielles distribuées de haute qualité. Ces représentations capturent un grand nombre de relations syntaxiques et sémantiques entre les mots. Ils ont renforcé la qualité des vecteurs et ils ont amélioré la vitesse d'apprentissage en employant un sous-échantillonnage des mots les plus fréquents. Les auteurs ont appliqué l'échantillonnage négatif, la représentation des mots ne prend pas en considération leur séquence, comme solution de rechange simple à la softmax hiérarchique. Cette dernière se caractérise par son incapacité à représenter les phrases idiomatiques. Par exemple, les significations des mots « Canada » et « Air » ne peuvent pas être facilement combinées pour obtenir « Air Canada ». Mikolov, Sutskever, Chen, Corrado, & Dean (2013) ont adopté une méthode qui permet d'apprendre de meilleures représentations de vecteur pour des millions de phrases.

En groupant les mots similaires, le modèle permet d'atteindre une meilleure performance. Ils ont appliqué l'approche sur les modèles probabilistes avec succès. Mikolov, Chen, Corrado, & Dean (2013) ont proposé des modèles efficaces pour prendre en compte un grand volume de données textuelles non structurées. Une machine optimisée peut entrainer plus de cent millions de mots en une journée. Les représentations de mots produites par les réseaux de neurones s'avèrent intéressantes en raison de la qualité d'apprentissage, qui peut encoder plusieurs régularités et modèles linguistiques.

L'extension du vecteur de mots vers les vecteurs de phrases est réalisée en identifiant un grand nombre de phrases à l'aide d'une approche axée sur les données (Le & Mikolov, 2014).

L'ampleur du texte à analyser représente un enjeu considérable. En effet, des chercheurs se sont intéressés aux textes de plus grande taille. La portée des représentations du word2vec a évolué pour prendre en considération des textes avec une taille de plus en plus grande. Le & Mikolov (2014) se sont également intéressés aux représentations distribuées des paragraphes et des documents. Ces algorithmes requièrent une taille fixe des caractéristiques du vecteur en entrée. Ils ont également affirmé que le sac de mots offre cette possibilité même s'il a deux faiblesses :

- perdre l'ordre des mots;
- ignorer la sémantique des mots.

Le & Mikolov (2014) ont proposé un vecteur de paragraphes, dans un apprentissage non supervisé, qui apprend les représentations et capture la sémantique. Semblable au traitement relatif aux phrases, l'algorithme représente chaque document à l'aide d'un vecteur dense employé pendant l'apprentissage pour prédire les mots dans le document. Le modèle concatène le vecteur de paragraphes et le vecteur de mots et prédit les mots suivants pour un contexte donné. Chaque vecteur de paragraphes se caractérise par son unicité, alors que ceux des mots sont partagés. L'apprentissage s'effectue en utilisant la propagation arrière et l'algorithme du gradient.

3.4 Méthodes d'évaluation

L'évaluation des algorithmes de similarité sémantique pour l'extraction des associations de mots est complexe pour les raisons suivantes (Alshahrani & Kapetanios, 2016) :

- les chercheurs utilisent des méthodes d'évaluation sans avoir de moyens efficaces pour les mesurer;
- la notion de « contexte » se présente sous plusieurs facettes, notamment le contexte N-Gram, le contexte défini par un nombre de mots à gauche et à droite du mot observé (*window*) et le contexte qui considère tout le texte dans lequel le mot est observé;
- nous distinguons trois types d'associations possibles : structure syntaxique, structure sémantique et structure associative.

Alshahrani & Kapetanios (2016) ont présenté l'utilisation d'un thésaurus, tel que WordNet (Fellbaum, 1998), comme étant la méthode d'évaluation la plus prometteuse pour ce genre de tâches. WordNet permet de regrouper les noms, les verbes, les adjectifs et les adverbes dans des ensembles de synonymes cognitifs. Chaque ensemble exprime un concept distinct. Ce thésaurus fournit une structure lexicale et sémantique compréhensible et complète pour identifier les relations entre les mots en anglais.

Nous présentons, dans ce qui suit, les résultats des évaluations relatives aux fonctions et les modèles d'apprentissage profond.

3.5 Fonctions d'évaluation

L'objectif des modèles, tel que Skip-Gram, consiste à trouver les représentations utiles pour prédire les mots environnants dans une phrase ou dans un document (Mikolov, Sutskever, Chen, Corrado, & Dean, 2013). Mikolov, Sutskever, Chen, Corrado, & Dean (2013) ont conçu des extensions pour Skip-Gram pour détecter les mots suivants dans une phrase ou dans un document. Ils ont comparé les différentes fonctions d'apprentissage appliquées pendant l'apprentissage. Nous donnons en exemple le softmaxe hiérarchique et l'échantillonnage négatif.

Le softmaxe hiérarchique constitue une approximation efficace du softmaxe total (Morin & Bengio, 2005). C'est un modèle qui utilise l'arbre binaire de Huffman pour la couche de sortie qui contient les mots et leurs branches. Cela définit un parcours aléatoire qui assigne les probabilités aux mots. Morin & Bengio (2005) ont utilisé l'arbre binaire, en affectant des codes courts aux mots fréquents, qui permet d'obtenir un apprentissage plus rapide. En effet, ils ont observé que le regroupement des mots selon leur fréquence aide à accélérer l'apprentissage dans les modèles de réseau de neurones.

L'échantillonnage négatif, qui peut remplacer le softmaxe hiérarchique, a pour objectif d'établir une différence entre les mots cibles de la distribution et le bruit en utilisant la régression logistique (Mikolov, Sutskever, Chen, Corrado, & Dean, 2013). Les mots les plus fréquents ont une plus grande probabilité de faire partie de l'échantillonnage négatif. Mikolov, Sutskever, Chen, Corrado, & Dean (2013) ont évalué la performance de chaque fonction pour des tâches syntaxiques et sémantiques. Ils ont montré que l'échantillonnage négatif obtient de meilleurs résultats par rapport au softmaxe hiérarchique pour les tâches de raisonnement analogique. Ils ont également prouvé que le sous-échantillonnage des mots fréquents améliore la vitesse d'apprentissage et produit une meilleure représentation des mots.

Mikolov, Sutskever, Chen, Corrado, & Dean (2013) ont affirmé que les modèles Skip-Gram facilitent le raisonnement analogique linéaire des vecteurs. Ils ont prouvé également que l'apprentissage standard des RNR s'améliore d'une manière significative avec l'augmentation des données.

Afin de prouver que le sens des phrases ne représente pas nécessairement la composition du sens de leurs mots individuels. Mikolov, Sutskever, Chen, Corrado, & Dean (2013) ont entrainé les vecteurs des phrases en trouvant les mots fréquents et non fréquents dans des contextes spécifiques. Ils ont évalué la qualité des représentations en utilisant le raisonnement analogique comprenant les phrases.ils ont utilisé un jeu des données relatives aux nouvelles parues dans la presse.
Nous donnons en exemple deux phrases par catégorie (Mikolov, Sutskever, Chen, Corrado, & Dean, 2013) :

- pour la catégorie presse : « New York » et « New York Times »;
- pour la catégorie hockey : « Montreal » et « Montreal Canadiens ».

L'objectif de l'évaluation consiste à calculer la quatrième phrase en utilisant les trois premières en entrée. Le meilleur modèle a obtenu une précision de 72 %, pour un jeu de données de 3218 exemples (Mikolov, Sutskever, Chen, Corrado, & Dean, 2013). Mikolov, Sutskever, Chen, Corrado, & Dean (2013) ont exploité le même jeu de données pour entrainer plusieurs modèles Skip-Gram. Ils ont utilisé des vecteurs avec une dimensionnalité de 300 et deux contextes de taille 5 et de taille 15. Les résultats ont montré que l'échantillonnage négatif atteint une précision acceptable pour les contextes de taille 5 et une meilleure précision pour les contextes de taille 15. Le sous-échantillonnage a permis d'obtenir un apprentissage plus rapide et une meilleure précision.

3.6 Modèles d'évaluation

Dans cette section, nous nous appuyons sur les travaux d'évaluation effectués sur les modèles doc2vec (dbow et dmpv) et word2vec (Skip-Gram et cbow) pour les tâches suivantes :

- analyser les sentiments;
- détecter les phrases en double;
- calculer la similarité sémantique.

Nous présentons également les résultats des évaluations des modèles en utilisant un large corpus externe et un plongement des mots préentrainé.

3.6.1 Analyser les sentiments

La tâche consiste à mesurer les sentiments, positifs, négatifs ou neutres, à partir du texte. Le & Mikolov (2014) ont évalué le vecteur de paragraphes en comparant son taux d'erreur avec d'autres modèles, tels que les RNR, MVS, word2vec. L'expérimentation utilise le jeu de données de Stanford pour l'analyse des sentiments (*the Stanford Sentiment Treebank dataset)*. Le & Mikolov (2014) ont démontré que le taux relatif au vecteur de paragraphes a atteint 12,2 %, ce qui représente le meilleur taux d'erreur par rapport aux autres modèles. Pour la tâche de classification, la méthode a enregistré une amélioration absolue de 2,4 points de pourcentage en matière de taux d'erreurs, ce qui se traduit en 16 % d'amélioration relative.

De plus, Le & Mikolov (2014) ont évalué le vecteur de paragraphes pour le jeu de données composés de revues de films (*IMDB*), composées d'une ou de plusieurs phrases. Les résultats montrent que le vecteur de paragraphes peut atteindre, pour le taux d'erreur, des valeurs plus basses que la barrière des 10 %. Son taux atteint 7,42 %, ce qui représente une amélioration absolue de 1,3 %, soit 15 % d'amélioration relative par rapport aux meilleurs résultats (Le & Mikolov, 2014).

3.6.2 Détecter les questions similaires

La tâche consiste à détecter les questions en double dans un forum web. Lau & Baldwin (2016) ont découpé les données en douze sous-forums, et ils ont considéré le titre et le corps de la question comme faisant partie du contenu du document. Chaque sous-forum est formé séparément et un document contient en moyenne 130 mots. Ils ont également calculé la similarité cosinus en utilisant les vecteurs produits par doc2vec et word2vec. L'approche a permis de trier les paires de documents par ordre décroissant de similarité. Lors de cette évaluation, les chercheurs ont choisi des paramètres pour ajuster doc2vec, tels que la taille du vecteur, l'échantillonnage pour les mots les plus fréquents et l'échantillonnage négatif des mots. Les résultats ont montré que word2vec et doc2vec ont obtenu une meilleure précision. Lau & Baldwin (2016) ont montré que le modèle doc2vec a atteint une précision de 0,94 et word2vec a atteint une précision de 0,86.

3.6.3 Calculer la similarité sémantique du texte

La deuxième tâche consiste à calculer la similarité sémantique du texte. Cette tâche a pour objectif de prédire la similarité d'une paire de documents en attribuant un score entre 0 et 5 (Lau & Baldwin, 2016).

- Le score 0 indique l'absence de similarité;
- Le score 5 indique une équivalence sémantique.

Cette approche consiste à effectuer un apprentissage supervisé. Pour chaque domaine, Agirre & al. (2015) ont évalué doc2vec et word2vec en utilisant la similarité sémantique du texte en anglais (*English Semantic Text Similarity STS*). Le jeu de données couvre cinq domaines. Chaque domaine contient 375 à 750 paires annotées. Les phrases ont une longueur plus courte que lors de la première expérimentation, elles comptent en moyenne treize mots. Lau & Baldwin (2016) ont inclus les résultats d'un système supervisé qui utilise l'alignement des mots et la composition sémantique des phrases (Sultan, Bethard, & Sumner, 2015).

Les résultats ont démontré que l'alignement des mots a obtenu de meilleurs résultats que les autres modèles, soit un score de 0,83. Doc2vec a obtenu un score de 0,78; word2vec a atteint un score de 0,74; et N-Gram a obtenu un score de 0,61 pour le domaine « headlines » (Lau & Baldwin, 2016). Doc2vec a atteint une meilleure performance que word2vec avec le même écart pour les tâches précédentes. Cela suggère que ce n'est pas pertinent d'utiliser doc2vec pour les petits documents. Nous observons une différence marginale entre les modèles dbow et dmpv, ayant obtenu des scores de 0,77 et 0,78 respectivement, même si le premier a atteint un meilleur score dans l'expérimentation de la première tâche. Dbow favorise les contextes de fenêtres plus larges. D'un autre côté, Lau & Baldwin (2016) ont conclu que le sous-échantillonnage représente le paramètre le plus important. La performance de la tâche diminue considérablement avec l'utilisation d'une valeur sous-optimale. Ils ont également démontré que le modèle dmpv requiert plus d'itérations d'apprentissage pour converger.

3.6.4 Évaluation avec un large corpus externe

L'objectif de cette expérimentation consiste à évaluer doc2vec par rapport à un large corpus externe, Lau & Baldwin (2016) ont utilisé deux corpus :

- la collection complète de Wikipédia pour la langue anglaise (WIKI);
- la collection d'articles de presse associés entre 2009 et 2015 (AP-NEWS).

Ils ont effectué le prétraitement à l'aide de la librairie de Stanford pour le TAL (*Stanford CoreNLP*[16]). Ils ont considéré chaque paragraphe d'un article comme un document doc2vec. Ils ont mené l'expérimentation avec dbow uniquement en raison de sa performance pour les deux tâches précédentes. Avec le même corpus, ils ont entrainé doc2vec et N-Gram en incluant les vecteurs de mots entrainés (Mikolov, Sutskever, Chen, Corrado, & Dean, 2013). Par la suite, ils ont calibré les résultats de doc2vec par rapport à deux compétiteurs qui ont proposé des méthodes de plongement des documents :

- Skip-Thought : une approche qui emploie l'hypothèse distributionnelle du vecteur de mots vers le vecteur de phrases (Kiros & al., 2015);
- la base de données des paraphrases (BDPP) : une base qui optimise le plongement des mots en maximisant la similarité cosinus avec le plongement des phrases (Ganitkevitch, Van Durme, & Callison-Burch, 2013).

Les résultats sont les suivants : N-Gram a obtenu de meilleurs résultats que Skip-Gram pour la première tâche et Skip-Gram affiche de meilleurs résultats pour la deuxième tâche. Doc2vec a obtenu de meilleures performances par rapport à word2vec et N-Gram dans toutes les tâches. L'écart est plus prononcé pour le texte de grande taille. Le modèle dbow a obtenu les mêmes résultats pour les deux corpus.

---

[16] https://stanfordnlp.github.io/CoreNLP/

Skip-Thought a atteint une performance limitée, soit un score de 0,57, alors que doc2vec a obtenu un score de 0,96. Cette performance est limitée par rapport à la méthode simple word2vec, qui a obtenu un score de 0,77. Comme la BDPP utilise la moyenne des vecteurs des mots, ce modèle a obtenu un meilleur résultat pour les documents de petite taille, soit un score de 0,92 (Law et Baldwin, 2016). Nous pouvons conclure que les vecteurs de mots sont adaptés pour les documents de petite taille et que le modèle dbow est adapté pour les documents de grande taille.

3.6.5 Évaluation avec un plongement des mots préentrainé

Le & Mikolov (2014) ont présenté l'implémentation dbow de Gensim[17] avec une option qui permet d'exécuter une étape préliminaire de Skip-Gram pour mettre à jour le plongement des mots avant d'exécuter dbow. Même si le modèle dbow fonctionne, en théorie, avec un plongement aléatoire, ils ont conclu que la performance diminue pour ce genre de configuration. Ceci peut s'expliquer par sa fonction objective, qui vise à maximiser le produit scalaire entre les plongements des documents et des mots. La distribution aléatoire du plongement des mots rend difficile l'optimisation de la représentation des documents. Lau & Baldwin (2016) ont mené une expérience avec deux corpus externes (Wiki, AP-News) et en utilisant un apprentissage Skip-Gram pour initialiser les mots dans le modèle dbow. Cette recherche s'est inspirée du fait qu'une meilleure initialisation aidera à obtenir une convergence plus rapide du modèle tout en améliorant la qualité du plongement. Les résultats ont montré que l'initialisation du plongement des mots n'a pas un grand impact sur la performance, le score obtenu pour le domaine Unix, par exemple, a atteint 0,98 pour les deux modèles.

[17] https://radimrehurek.com/gensim/models/doc2vec.html

3.7 Conclusion

Nous avons parcouru les approches qui permettent d'extraire les associations des mots et de calculer les similarités sémantiques. Nous nous sommes intéressé aux évaluations des fonctions d'apprentissage ainsi qu'aux modèles word2vec et doc2vec pour des tâches distinctes, telles que l'analyse des sentiments et le calcul des similarités sémantiques, appliquées dans des contextes différents.

Mikolov, Sutskever, Chen, Corrado, & Dean (2013) ont démontré que l'échantillonnage négatif permet d'obtenir un apprentissage efficace avec une convergence plus rapide et une meilleure performance du modèle. Law et Baldwin (2016) ont conclu que doc2vec et word2vec obtiennent les meilleurs résultats en ce qui concerne la précision et le taux d'erreur. Word2vec aide à traiter les documents de petite taille, alors que doc2vec aide à traiter ceux de grande taille. Law et Baldwin (2016) ont conclu également que l'initialisation Skip-Gram du plongement des mots n'impacte pas la performance des modèles.

Notre objectif consiste à classifier les exigences des systèmes d'information dans des catégories spécifiques. Nous établirons par la suite les correspondances entre les artéfacts extraits de chaque catégorie. Notre approche comprend l'utilisation combinée de doc2vec et word2vec afin de classifier les exigences et d'en extraire les associations. Nous fournissons plus de détails dans le chapitre suivant à propos de l'approche préconisée ainsi que des moyens d'évaluation à appliquer.

CHAPITRE IV

APPROCHE ET MÉTHODOLOGIE

4.1 Introduction

Berry, Ferrari, & Genesis (2017) ont étudié, au début des années quatre-vingt, les problématiques d'utilisation du langage naturel pour le domaine de l'ingénierie des exigences. Ils ont exploré la possibilité d'appliquer le traitement du langage naturel pour automatiser les processus liés à l'analyse du texte dans les exigences. Ryan (1993) s'est intéressé à l'utilisation du TAL dans l'exécution des tâches qui traitent les spécifications des exigences rédigées en langage naturel. Par la suite, des chercheurs ont appliqué le TAL pour diverses tâches cognitives :

- la traçabilité des exigences (Antoniol, Canfora, De Lucia, & Merlo, 1999; Gotel & al., 2012; Gervasi & Zowghi, 2014);
- la classification des exigences (Casamayor, Godoy, & Campo, 2012a);
- l'analyse des revues des applications (Maalej, Kurtanović, Nabil, & Stanik, 2016);
- la synthèse des modèles (Robeer, Lucassen, Van Der Werf, Dalpiaz, & Brinkkemper, 2016);
- le traitement des ambiguïtés (Tjong & Berry, 2013);
- la détection des anomalies (Fabbrini, Fusani, Genesis, & Lami, 2001).

EN 2018, c'était le vingtième anniversaire du début des travaux relatifs à l'automatisation du processus de la traçabilité (Dekhtyar & Hayes, 2018). Les premiers travaux ont commencé avec le rétablissement des liens entre le code et la documentation des systèmes orientés objet. Antoniol, Canfora, De Lucia, & Merlo (1999) ont introduit une approche qui découvre les liens entre le code source, écrit dans un langage de programmation, et les spécifications des exigences, les documents de conception, les manuels des utilisateurs, les journaux de développement des systèmes et les erreurs. Les auteurs ont proposé d'autres approches pour découvrir les liens dans les exigences. Citons par exemple l'application de la technique de l'indexation sémantique latente (Marcus & Maletic, 2003). Les chercheurs ont également adopté des stratégies d'amélioration, telles que la rétroaction effectuée par un analyste pour améliorer les liens candidats découverts (Hayes, Dekhtyar, & Sundaram, 2006).

De plus, les ingénieurs de développement en génie logiciel gèrent une grande quantité d'informations. Ils structurent ces informations sous forme d'artéfacts à travers les étapes de réalisation du système. Borg, Runeson, & Ardö (2014) ont affirmé que les chercheurs se sont intéressés à la tâche de récupération des liens ou correspondances entre les artéfacts; ils l'ont considérée comme une problématique d'extraction de l'information (*Information Retrieval*).

L'objectif de la thèse consiste à établir, à partir des exigences textuelles, les correspondances entre les artéfacts du système. L'extraction de l'information est une technique qui consiste à trouver, à partir d'une large collection de données textuelles électroniques, des documents textuels pertinents non structurés et qui répondent à un besoin d'information spécifique (Manning, Raghavan, & Schütze, 2010).

Borg, Runeson, & Ardö (2014) ont souligné que les chercheurs ont distingué entre l'extraction de l'information et le TAL. Ils considèrent que l'extraction de

l'information correspond au processus de résolution qui permet d'établir les correspondances entre les artéfacts, alors que le TAL correspond aux techniques qui facilitent et supportent l'exécution du processus de résolution (Borg, Runeson, & Ardö, 2014). D'autres auteurs ont utilisé le terme « découverte des connaissances dans les données » (KDD pour *Knowledge Discovering in Data*) pour évoquer l'analyse intelligente de l'information, qui regroupe le domaine de l'extraction de l'information et le domaine de l'apprentissage machine (Dekhtyar & Hayes, 2018).
Kruchten (2004) a défini un artéfact comme toute information, finale ou intermédiaire, produite et maintenue lors du cycle de développement logiciel. Il a cité des exemples d'artéfacts, tels que les exigences, les spécifications conceptuelles, le code source, les spécifications des essais, les guides d'utilisation et rapports d'anomalies. La traçabilité consiste à établir et à utiliser les correspondances entre les artéfacts que les acteurs (concepteurs, analystes, programmeurs…) ont produits et manipulés lors du cycle de développement logiciel (Cleland-Huang, Gotel, & Zisman, 2012).

Dans ce chapitre, nous présentons le processus d'établissement des correspondances entre les artéfacts ainsi que les techniques appliquées pour permettre et faciliter la réalisation et l'exécution de ce processus. Nous décrivons le contexte du travail, les travaux connexes et l'approche proposée.

4.2 Contexte de l'approche

Établir les correspondances entre les artéfacts nécessite d'extraire des informations pertinentes en appliquant des modèles algébriques, probabilistes ou statistiques (Borg, Runeson, & Ardö, 2014). Nous décrivons dans cette section les techniques d'extraction ainsi que les différents modèles appliqués dans les recherches publiées sur la découverte de la traçabilité dans le domaine de traitement des exigences.

4.2.1 Techniques d'extraction

Les techniques d'extraction des connaissances s'exécutent en une ou plusieurs phases selon l'objectif de l'extraction. De Almeida & al. (2018) ont décrit les étapes de traitement utilisées pour analyser les exigences textuelles :

- structuration du texte : c'est une technique de formatage qui permet d'éviter les erreurs grammaticales. Des auteurs ont adopté cette structuration pour détecter l'ambiguïté. Elle facilite le traitement de plusieurs types de textes en entrée, dont la rédaction des exigences. Elle permet également de limiter ou pas la structure des documents à analyser. Elle consiste à appliquer un prétraitement avant l'analyse du texte, au moment de la rédaction;
- traitement du texte : c'est une technique qui consiste à analyser le texte rédigé en langage naturel et à en extraire des données pertinentes. Elle implique des analyses morphologiques, syntaxiques et sémantiques. Les auteurs ont organisé leurs solutions sous forme de séquences et de filtres où chaque fonction utilise le produit de la précédente. Nous donnons en exemple l'étiquetage de la partie du discours, la racinisation, la lemmatisation et l'étiquetage syntaxique;
- génération des modèles : c'est une technique qui consiste à appliquer un ensemble de règles pour extraire les connaissances en respectant des structures de fragments de textes. Les auteurs ont employé des modèles qui s'appuient sur des règles ou des méthodes statistiques;

- lexiques : c'est une technique d'apprentissage qui permet l'utilisation des connaissances du domaine. Les chercheurs ont appliqué des lexiques créés manuellement ou automatiquement. Les lexiques sont représentés sous forme de base de connaissances ou d'ontologies relatives au domaine. Ils facilitent l'application de techniques, telles que l'identification des entités, la détection des ambiguïtés et la lemmatisation. Les lexiques fournissent plus de contexte aux mots avec une sémantique propre au domaine;
- mesure de similarité : c'est une technique qui consiste à calculer la similarité entre les mots, les phrases et les paragraphes. Cette similarité sémantique permet de comparer et de vérifier la conformité entre les phrases, les modèles et les actions;
- identification des entités : c'est une technique qui permet d'identifier les entités dans le texte. Ces entités sont des acteurs, des activités, des événements ou des concepts extraits automatiquement à partir du texte. Ce sont des entités qui peuvent être corrélées avec des modèles et rôles grammaticaux. Elles facilitent la validation entre le texte et les modèles générés. Nous donnons en exemple l'étiquetage des rôles et l'apprentissage statistique.

4.2.2 Modèles d'extraction

Le processus d'établissement des correspondances consiste à inclure les liens, les corrélations, les dépendances, la détection de la similarité et la consolidation des documents. Nous utilisons les termes « liens » et « correspondance » sans distinction, de même que les termes « dépendance, relation et similarité entre artéfacts » (Borg, Runeson, & Ardö, 2014). Borg, Runeson, & Ardö (2014) ont recensé les modèles d'extraction appliqués dans le cadre de la traçabilité des exigences; nous les présentons dans le tableau 4.1. Afin de permettre la découverte de la traçabilité dans les exigences, les modèles d'extraction de l'information appliquent souvent le sac de mots (*Bag-Of-Words*). Cette technique consiste à représenter un document sous forme

de collection de mots non ordonnés (Manning, Raghavan, & Schütze, 2010). Zhai (2007) a distingué deux classes de modèles qui dépendent de la pertinence entre les requêtes et les documents, algébriques et probabilistes.

Les modèles algébriques considèrent la corrélation entre la pertinence des documents et leur similarité. Salton & al. (1975) ont présenté le modèle vectoriel, qui est le modèle le plus adopté par la communauté. Les requêtes et les documents sont représentés sous forme de vecteurs à grande dimension. La similarité entre les vecteurs est calculée par une fonction de calcul de la distance entre les vecteurs. La pertinence des termes individuels ne suffit pas, d'où l'application de la pondération relative selon leur pertinence. Nous distinguons deux types de pondération : binaire ou brute (Singhal, 2001).

- La pondération binaire s'appuie sur l'existence ou l'absence des termes.
- La pondération brute s'appuie sur la fréquence des termes ou la fréquence inversée dans les documents TF-IDF. Les chercheurs l'utilisent pour pondérer un terme en se basant sur la taille du document ainsi que sa fréquence par rapport à toute la collection.

Pour l'extraction probabiliste, la pertinence entre la requête et le document est estimée à l'aide d'un modèle probabiliste. L'extraction est exprimée comme un problème de classification des documents pertinents ou non pertinents selon une probabilité d'importance (Singhal, 2001). Robertson & Jones (1976) ont introduit le modèle d'extraction indépendante, qui s'appuie sur la distribution indépendante des termes. Ce modèle applique un classificateur bayésien naïf pour classifier les documents (Lewis, 1998). Le BM25 est le schéma de pondération le plus utilisé pour améliorer les résultats de ce type de modèle (Robertson & Zaragoza, 2009).

Afin d'établir les liens entre les artéfacts, des chercheurs ont calculé la similarité cosinus sémantique entre les mots. Celle-ci est la mesure la plus utilisée pour les

modèles algébriques. Elle consiste à calculer le cosinus de l'angle entre deux vecteurs. Cependant, des auteurs ont appliqué le coefficient de Dice et l'index de Jaccard (Manning, Raghavan, & Schütze, 2010). Afin de réduire le bruit du langage naturel (synonyme et polysémie), Deerwester, Dumais, Furnas, Landauer, & Harshman (1999) ont introduit l'indexation sémantique latente (ISL) pour réduire la dimension du modèle vectoriel. L'ISL permet de décomposer la dimension en nouvelles dimensions représentées par des combinaisons de termes (concepts). Les chercheurs ont appliqué une rétroaction effectuée par des spécialistes pour améliorer ce type de modèles.

L'application du réseau d'inférence probabiliste (Turtle et Croft, 1991) consiste à calculer et à modéliser la pertinence à l'aide de l'incertitude associée à l'inférence de la requête à partir du document. Ce modèle consiste à instancier chaque terme avec un certain poids (Zhai, 2007); plusieurs termes accumulent un score numérique de poids relatif à une combinaison entre le document et la requête.

Tableau 4.1 Recensement des modèles d'extraction utilisés pour la traçabilité des exigences (Borg, Runeson, & Ardö, 2014)

Modèle algébrique	Modèle probabiliste	Modèle statistique	Schémas de pondération	Mesures et fonctions de similarité	Stratégie d'amélioration
Modèle vectoriel	Modèle d'indépendance binaire	Modèle du langage	Binaire	Similarité cosinus	Rétroaction
Indexation sémantique latente (ISL)	Réseau d'inférence probabiliste	Indexation sémantique latente (Probabiliste)	Brut	Coefficient de Dice	Thésaurus
	Meilleure correspondance 25 (BM25)	Allocation latente de Dirichlet (ALD)	Fréquence des termes et fréquence inversée du document (TFIDF)	Index de Jaccard	Structure de la phrase
		Modèle des thèmes corrélés	BM25	Divergence de Jensen-Shannon	Partitionnement (*clustering*)

De plus, des chercheurs ont appliqué d'autres modèles, tels que les modèles statistiques du langage. Ces modèles permettent de classifier les termes de la requête à l'aide d'une probabilité que le langage du document peut générer (Ponte & Croft, 1998). Une autre version des modèles du langage consiste à décrire les documents comme un ensemble de thèmes. Chaque thème contient un ensemble de caractéristiques qui le distingue des autres (Zhai, 2008). Pour rétablir la traçabilité des exigences, les chercheurs ont appliqué les techniques suivantes :

- l'indexation sémantique et probabiliste latente (Hofmann, 2001);
- l'allocation latente de Dirichlet (Blei, Ng, & Jordan, 2003);
- la modélisation corrélée des thèmes (Blei & Lafferty, 2007);
- la modélisation relationnelle des thèmes (Chang & Blei, 2010).

Afin de calculer les liens entre les artéfacts, des chercheurs ont appliqué les mesures de similarité, telles que la divergence de Jensen-Shannon (Borg, Runeson, & Ardö, 2014). D'ailleurs, Borg, Runeson, & Ardö (2014) ont affirmé que des chercheurs ont appliqué les stratégies d'amélioration des modèles en introduisant la validation par des spécialistes comme moyen de rétroaction. Les chercheurs ont également employé le thésaurus pour permettre le rétablissement des traces dans les exigences. C'est un outil qui sert à contrôler le vocabulaire prédéfini pour un domaine spécifique (Borg, Runeson, & Ardö, 2014). Des domaines comme la médecine, la biologie, l'aéronautique contiennent des concepts et des relations explicites dans le langage naturel utilisé (Aitchison, Bawden, & Gilchrist, 2003). L'utilisation des thésaurus fournit au système d'extraction le vocabulaire restreint des termes usuels et leur sémantique, à utiliser pour indexer et rechercher l'information pertinente. Borg, Runeson, & Ardö (2014) ont souligné que les chercheurs ont également appliqué les phrases (*phrasing*), qui représentent un ensemble de deux ou plusieurs mots. Cette méthode a obtenu de meilleurs résultats que celle des modèles de mots. La détection des phrases pour l'indexation s'effectue à l'aide des analyses statistiques, des fréquences des termes et de la cooccurrence. Elle peut être réalisée en analysant la

structure grammaticale ou en appliquant l'étiquetage des parties du discours. Charikar, Chekuri, Feder, & Motwani (2004) ont adopté le partitionnement (*clustering*), qui s'appuie sur le fait que les documents pertinents similaires sont regroupés.

Baeza-Yates & Ribeiro (2011) ont évalué les modèles avec un objectif de classification par contexte d'utilisation. Ils ont utilisé des mesures comme la précision, le rappel et la mesure-F. La précision désigne la fraction de documents pertinents extraits, le rappel correspond au pourcentage des documents extraits, alors que la mesure-F représente la moyenne harmonique de la précision et du rappel.

4.3 Travaux connexes

Dans cette section, nous abordons les modèles appliqués pour établir la traçabilité entre les artéfacts produits lors du développement logiciel. Nous présentons les articles qui se sont appuyés sur le prétraitement du texte ainsi que les types de liens extraits d'après les différentes publications. Nous exposons les contextes d'évaluation et les types de données et d'exigences utilisés par les chercheurs. Nous énumérons les stratégies d'amélioration sélectionnées par les auteurs ainsi que les étapes nécessaires pour extraire les correspondances entre les artéfacts.

4.3.1 Modèles d'extraction

Des auteurs ont mené une étude systématique sur les correspondances entre les approches qui ont ciblé le domaine de l'extraction de l'information et la traçabilité dans les exigences (Borg, Runeson, & Ardö, 2014). Ils ont relevé que la plupart des recherches ont adopté les modèles algébriques, tels que le modèle vectoriel et l'indexation sémantique latente. Vingt-neuf articles sur les cent trente-deux publications étudiées préconisaient les modèles probabilistes, incluant quatorze applications des modèles statistiques. La figure 4.1 montre la taxonomie des modèles appliqués pour rétablir les liens dans les exigences dans le cycle de développement logiciel (Borg, Runeson, & Ardö, 2014). Les auteurs ont noté que cinq approches ne correspondent pas aux modèles de la taxonomie, que nous reprenons dans la figure 4.1. Nous donnons en exemple les techniques de l'intelligence distribuée (Sultanov & Hayes, 2010) et le B-splines, qui représente une combinaison linéaire de splines positives à support compact minimal (Capobianco, De Lucia, Oliveto, Panichella, & Panichella, 2009).

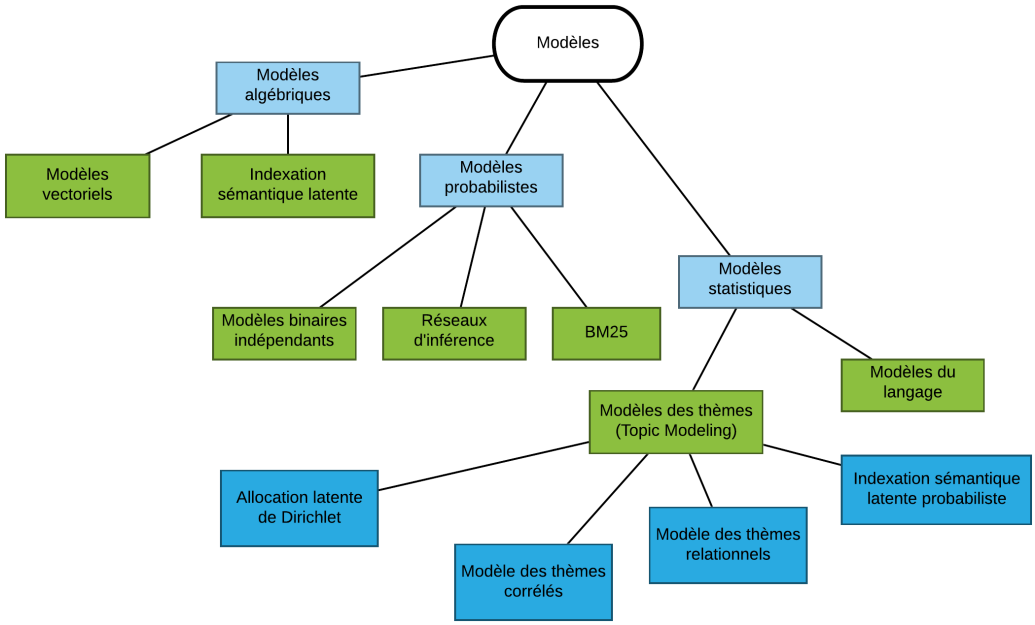


Figure 4.1 Taxonomie des modèles appliqués pour établir la traçabilité dans les exigences des systèmes

Borg, Runeson, & Ardö (2014) ont remarqué que la plupart des recherches appliquaient les modèles algébriques (Antoniol, Canfora, Casazza, De Lucia, & Merlo, 2002; Marcus & Maletic, 2003; Poshyvanyk, Gueheneuc, Marcus, Antoniol, & Rajlich, 2007; Zamani, Lee, Shokripour, & Anvik, 2014). Ils ont observé que les modèles vectoriels et les modèles de l'indexation sémantique latente représentent la majorité des applications pour rétablir la traçabilité et les correspondances entre les exigences.

Des chercheurs ont soutenu que les modèles vectoriels ont obtenu les meilleurs résultats (Borg & Runeson, 2013). Des chercheurs ont confirmé que la divergence Jensen-Shannon (Cha, 2007) a obtenu les mêmes résultats que le modèle vectoriel en mesurant la similarité pour établir la traçabilité (Abadi, Nisenson & Simionovici, 2008; Oliveto, Gethers, Poshyvanyk, & De Lucia, 2010; Gethers, Oliveto, Poshyvanyk, & De Lucia, 2011). De plus, Borg & Runeson (2013) ont conclu que le modèle vectoriel classique a atteint les mêmes performances que les autres modèles plus complexes, tels

que l'indexation sémantique latente (Marcus & Maletic, 2003) et l'allocation latente de Dirichlet (H.Asuncion, A.Asuncion, & Taylor, 2010).

Marcus & Maletic (2003) ont appliqué l'indexation sémantique latente (ISL), qui consiste à définir le vocabulaire du corpus. Le nombre de mots (termes) dans le vocabulaire détermine la dimension du corpus en utilisant la fréquence de l'occurrence des termes dans le document et dans toute la collection. Chaque terme est pondéré par une combinaison de sa pondération locale et de sa pondération globale. Pour deux documents d1 et d2, la similarité sémantique se calcule à l'aide du cosinus entre leurs vecteurs. La valeur du cosinus peut avoir des valeurs entre -1 et 1. Deux documents similaires obtiendront une similarité égale à 1.

Le calcul du cosinus consiste à spécifier un seuil qui détermine le degré de correspondance entre deux documents. A cet effet, Marcus & Maletic (2003) ont présenté deux options pour déterminer le seuil :

- déterminer d'une manière heuristique (les auteurs ont présenté une valeur égale à 0,7, qui correspond à un angle de 45°);
- faire spécifier une valeur fixe par l'utilisateur.

De plus, des chercheurs ont proposé des approches pour rétablir automatiquement les liens entre les modèles de conception orientés objet et le code source (Antoniol, Caprile, Potrich, & Tonella, 2001; Antoniol, Caprile, Potrich, & Tonella, 2000). Ils ont employé les attributs de classes comme élément important pour découvrir les liens. McMillan, Poshyvanyk, & Revelle (2009) ont appliqué une technique qui combine les informations textuelles et la structure de l'information. Ganitkevitch, Van Durme, & Callison-Burch (2013) ont adopté une méthode pour explorer les correspondances entre les types et les variables dans un programme Java et les diagrammes des cas d'utilisation. D'autres auteurs ont proposé une technique pour améliorer l'établissement des liens entre les documents textuels et le code source (Diaz & al., 2013).

Des chercheurs considèrent que les techniques simples d'extraction de l'information se classent parmi les plus utiles pour établir la traçabilité dans les exigences (Falessi, Cantone, & Canfora, 2013; Ali, Gueheneuc, & Antoniol, 2011). D'autres ont également exploré l'utilisation de la structure du texte pour établir les correspondances (Antoniol, Caprile, Potrich, & Tonella, 2001; Antoniol, Caprile, Potrich, & Tonella, 2000; McMillan, Poshyvanyk, & Revelle, 2009).

D'autres chercheurs ont quantifié les similarités entre les artéfacts en s'appuyant sur les similarités morphologiques (modèle vectoriel et ISL), sur la proximité des thèmes (ALD), sur la proximité sémantique des mots (plongement des mots) (Bella, Gervais, Bendraou, Wouters, & Koudri, 2018). Ils ont calculé la distance entre les mots en appliquant le WMD (*Word Mover's Distance*) (Kusner, Sun, Kolkin, & Weinberger, 2015).

Les techniques d'extraction dépendent de l'utilisation de l'information textuelle dans les documents pour établir les correspondances et les liens. Ces techniques évitent le problème du bruit dans le texte. Le bruit provoque le rétablissement des liens faux positifs avec une faible précision (Capobianco, De Lucia, Oliveto, Panichella, & Panichella, 2013).

4.3.2 Prétraitement

Afin de réduire le bruit dans le texte, les chercheurs ont appliqué des méthodes de prétraitement pour conserver uniquement les termes pertinents dans le texte. Borg, Runeson, & Ardö (2014) ont noté que la plupart des publications ont adopté le retrait des mots vides (*stopword*) et la racinisation (*Stemming*). Ces deux techniques représentent la combinaison la plus utilisée par les chercheurs. D'autres ont employé le fractionnement des identifiants pour le code source en séparant les mots composés et en remplaçant les espaces soulignés par un espace (exemple : GetCustomerId devient Get Customer Id).

Baeza-Yates & Ribeiro (2011) ont choisi d'autres techniques de prétraitement pour supprimer le bruit à partir du texte. Ils ont proposé de retirer les termes communs dans les documents. Dit, Guerrouj, Poshyvanyk, & Antoniol (2011) ont fractionné les identifiants. D'autres auteurs ont indexé les noms et ils ont appliqué une analyse morphologique, telle que la racinisation (Capobianco, De Lucia, Oliveto, Panichella, & Panichella, 2009; Chowdhury & McCabe, 1998). La fréquence inversée des documents permet de détecter les termes communs qui ajoutent du bruit au texte : si un terme apparait dans tous les documents, alors le modèle lui assignera une pondération faible (Antoniol, Canfora, Casazza, De Lucia, & Merlo, 2002).

Des auteurs ont rétabli les liens en appliquant une analyse numérique (Capobianco, De Lucia, Oliveto, Panichella, & Panichella, 2009). Ils ont réduit le bruit dans le texte en proposant un filtre de termes basé sur les mots vides et sur la nature grammaticale du terme extrait. Ils ont observé que le langage utilisé dans la documentation des spécifications du logiciel est un langage sectoriel. Ils ont donc uniquement pris en compte les noms qui représentent les termes les plus riches sémantiquement.

L'étiquetage de la partie du discours permet d'identifier tous les termes dans un corpus selon le contexte et la grammaire de la phrase. Etzkorn, Bowen, & Davis (1999) ont appliqué l'étiquetage pour identifier les entités dans le code source (noms des méthodes et attributs). Ils ont réussi à générer des sommaires pertinents relatifs aux modules du code source. Zou, Settimi, & Cleland-Huang (2006) ont établi dynamiquement les liens en construisant les phrases à partir des exigences. Leur approche s'appuie sur le glossaire du projet pour trouver des phrases supplémentaires et pondérer la contribution des termes et des phrases clés. Abebe & Tonella (2010) ont appliqué l'étiquetage pour extraire les concepts à partir des identifiants dans le code source. Ils ont conçu un arbre de dépendance et ils ont extrait une ontologie en faisant correspondre les entités linguistiques. Ils ont utilisé l'ontologie pour améliorer la précision d'identification. Shokripour, Anvik, Kasirun, & Zamani, (2015) ont développé un rapport d'anomalies en choisissant uniquement les noms (référentiel, identifiants et commentaires dans le

code source) à partir des sources d'information. Capobianco, De Lucia, Oliveto, Panichella, & Panichella (2009, 2013) ont proposé une méthode heuristique pour prendre en considération uniquement les noms afin d'améliorer la précision d'extraction. Une autre approche a consisté à adopter un étiquetage limité de la partie du discours en incluant uniquement les noms et les verbes pour établir les traces entre les exigences textuelles et le code source (Ali, Cai, Hamou-Lhadj, & Hassine, 2019). Les chercheurs ont également appliqué les schémas de pondération pour sélectionner les termes les plus pertinents dans un document. La fréquence des termes TF-IDF se classe parmi les techniques les plus utilisées et les plus appliquées par la communauté. La phase de prétraitement permet de retirer le bruit du texte afin d'établir les liens entre les artéfacts. Nous présentons dans la section suivante les types de liens et de correspondances que nous considérons dans ce travail.

4.3.3 Types de liens et de correspondances entre les artéfacts

Borg, Runeson, & Ardö (2014) ont classé les exigences en trois catégories :

- exigences extraites dans les différentes étapes de développement des systèmes. Nous donnons en exemple les exigences du marché, les exigences système, les exigences fonctionnelles, les cas d'utilisation et les spécifications de conception;
- exigences de vérification (Test), qui regroupent les activités de validation telles que les descriptions des cas d'essai et les scripts de test;
- exigences du code source utilisées pour réaliser et développer le système.

Le tableau 4.2 montre le nombre d'articles étudiés par type de lien. Notons que les liens extraits entre exigences à partir des étapes de code source ont été utilisés dans la majorité des travaux explorés (Borg, Runeson, & Ardö, 2014).

Tableau 4.2 Nombre de publications par type de lien

Type de lien	Nombre d'articles
Exigences – exigences	37
Code – exigences	32
Exigences – test	9
Exigences – code – test	6
Code – manuel	6
Code – anomalie	2
Code – test	2
Code – anomalie	1
Test – test	1
Test – anomalie	1

La majorité des exigences concernent les artéfacts conceptuels et le code source. En effet, les chercheurs ont ciblé les liens existants dans des contextes d'évaluation comprenant des documents textuels qui proviennent de différentes sources, pour la plupart universitaires.

4.3.4 Contexte d'évaluation

Borg, Runeson, & Ardö (2014) ont identifié cent trente-deux évaluations, dont quarante-deux études menées dans des contextes privés ou pour des exigences qui concernent l'agence américaine de l'espace (NASA). Dix-neuf évaluations reposaient sur des projets code source ouvert (*open source*) et soixante-cinq études exploitaient des ensembles de données qui proviennent du domaine universitaire. Parmi ces dernières, trente-quatre appartenaient à un ensemble de données produit par les étudiants. Notons le manque de contextualisation relative aux industries et aux domaines spécifiques qui permettrait d'analyser et d'expérimenter la traçabilité des exigences rédigées et exploitées dans le cadre de cas réels.

4.3.5 Stratégie d'amélioration des modèles

La plupart des stratégies d'amélioration des modèles d'extraction avaient comme objectif d'améliorer la précision et le rappel des modèles. Des auteurs ont appliqué la rétroaction comme stratégie d'amélioration (Antoniol, Canfora, Casazza, & De Lucia, 2000; Hayes & al., 2007). Cette méthode offre la possibilité de juger partiellement les résultats des modèles afin d'améliorer la recherche des liens. Hayes, Dekhtyar, & Sundaram (2006) ont également supporté leurs travaux avec les thésaurus pour traiter les synonymes et restreindre le langage selon le domaine à analyser. Le partitionnement des résultats a permis de mieux les structurer en classifiant les liens candidats pour faciliter la rétroaction des spécialistes du domaine (Duan & Cleland-Huang, 2007). L'utilisation des phrases en explorant les documents à la place du sac de mots a facilité la prise en compte des contextes et des séquences des mots (Zou, Settimi, & Cleland-Huang, 2006). L'application des expressions régulières a permis d'améliorer l'application du modèle vectoriel en établissant les correspondances entre les noms des classes et le texte dans les exigences (Chen & Grundy, 2011). D'autres chercheurs ont proposé une approche semi-supervisée pour classifier les liens candidats faux positifs manuellement et les utiliser pour construire un modèle prédictif (Bella, Gervais,

Bendraou, Wouters, & Koudri, 2018). Le modèle effectue l'apprentissage en utilisant un ensemble de données qualifié par un analyste.

4.3.6 Processus et étapes de réalisation

Extraire les correspondances entre les artéfacts s'effectue en exécutant un ensemble de phases l'une après l'autre pour la conception et la réalisation du modèle. Afin de permettre à un spécialiste de valider et de corriger au besoin les modèles appliqués, Poshyvanyk, Gueheneuc, Marcus, Antoniol, & Rajlich (2007) ont suivi cinq étapes principales :

1) créer un corpus pour l'indexer avec ISL;
2) indexer le corpus qui applique l'indexation ISL;
3) formuler les requêtes en sélectionnant manuellement l'ensemble de mots qui décrivent les caractéristiques;
4) classer les documents en calculant la similarité cosinus entre les vecteurs de mots;
5) analyser les résultats et qualifier les similarités calculées.

Nous observons que ces étapes ne contiennent pas une phase de prétraitement spécifique. Cette étape consiste à nettoyer le texte et à le préparer pour l'application de la sémantique latente au corpus. De Lucia, Marcus, Oliveto, & Poshyvanyk (2012) ont décrit un processus de quatre étapes clés qui requièrent une intervention et un jugement humains, utilisé dans les modèles algébriques. Ces étapes sont les suivantes :

1) analyser le document, extraire et prétraiter le texte : effectuer un prétraitement du texte contenu dans les documents à un niveau de granularité donnée (sections, fichier de classes ou exigences individuelles);
2) indexer le corpus avec les méthodes d'extraction du texte : extraire et pondérer les caractéristiques afin de créer un index;
3) générer les liens entre les artéfacts : le résultat de l'étape 2 sert à calculer la similarité entre les artéfacts afin de choisir les liens candidats;

4) analyser les liens candidats : mettre les liens candidats à la disposition d'un ingénieur ou d'un analyste pour révision, validation et confirmation des liens.

4.3.7 Observations et notes

Nous observons que les modèles algébriques (modèle vectoriel, indexation sémantique latente) représentent la majorité des applications dans le cadre de la traçabilité des exigences. Cependant, les chercheurs se sont intéressés récemment à l'application des modèles statistiques. Nous avons exploré les publications qui ont étudié l'application des modèles de thèmes (*Topic Modeling*); elles sont peu nombreuses par rapport à celles sur les modèles algébriques. Cet intérêt grandissant au fil des années encourage l'adoption de ce type de modèles pour établir les liens entre les artéfacts.
Nous notons également que dans plusieurs publications, les auteurs n'ont pas précisé les types d'artéfacts. Ils se sont concentrés, pour la plupart, sur l'exploration et l'extraction des liens entre les exigences textuelles et les classes ou entités conceptuelles. Comme cela apparait dans le tableau 4.2, dans la majorité des travaux, les auteurs ont étudié les exigences à travers les différents niveaux d'abstraction. Le code source vient en deuxième lieu, alors que peu d'études ont analysé les artéfacts relatifs aux tests. Nous notons une seule publication dans laquelle les auteurs ont considéré les manuels utilisateurs en tant qu'artéfacts. La plupart des évaluations des modèles ont employé des ensembles de données bipartis qui contiennent peu d'artéfacts. Ils ont utilisé des artéfacts produits dans des environnements universitaires ou qui proviennent des ensembles de données de l'agence spatiale (NASA). Nous avons noté que peu d'études ont appliqué les modèles aux environnements réels spécifiques à une industrie ou à un domaine.

4.4 Approche proposée

Nous abordons dans cette section les principes et lignes directrices de l'approche. Nous présentons par la suite l'outil technique que nous utilisons pour concevoir et évaluer les modèles, et nous décrivons les tâches traitées dans la thèse en exposant les modèles que nous adoptons pour les réaliser.

4.4.1 Principes et lignes directrices

Selon les observations de la section 4.3.7, les principes de l'approche que nous proposons pour ce travail sont les suivants :

- modèles statistiques : notre approche s'inspire des différents travaux connexes au domaine d'application de la thèse. Nous adoptons la méthode statistique ALD ainsi que le cosinus soft pour mesurer la relation entre les artéfacts. Les chercheurs ont appliqué les modèles vectoriels pour la plupart des travaux, alors que peu d'entre eux ont utilisé les modèles statistiques. Oliveto, Gethers, Poshyvanyk, & De Lucia (2010) ont comparé les résultats des modèles vectoriels, ISL et ALD et ils ont conclu que ces modèles produisent les mêmes performances et les mêmes résultats pour le contexte des liens entre les documents et le code source. L'ALD permet de capturer les informations manquées par les autres modèles. Nous appliquons un modèle statistique afin de classifier les documents selon les types d'artéfacts à extraire. Nous adoptons l'ALD pour distinguer les thèmes à travers les exigences et préparer les documents pour l'extraction des artéfacts avant d'établir les correspondances entre eux;
- prétraitement : nous appliquons une phase de prétraitement dans laquelle nous préparons le texte à analyser pour éviter les termes qui introduisent du bruit dans la classification et le calcul de similarité. Nous appliquons le retrait des mots vides et la racinisation sur le texte dans les exigences. Nous prenons en

considération le niveau mot et le niveau phrases pour une meilleure prise en compte du contexte lors de la classification et du calcul de la similarité entre les artéfacts extraits;

- spécification et précision des artéfacts : nous précisons la liste des artéfacts et leur appartenance aux étapes de développement. Nous considérons les thèmes suivants : les exigences, les cas d'utilisation, les processus d'affaires, les entités conceptuelles et les cas d'essai;
- ensemble de données et contexte d'évaluation : nous évaluons notre approche en l'appliquant sur des documents d'exigences et de spécifications réels. Les documents proviennent d'une organisation réelle qui développe des solutions logicielles pour son compte et pour le compte de ses clients;
- stratégie d'amélioration : afin d'améliorer les modèles, nous faisons intervenir un spécialiste qui valide les documents classifiés, les artéfacts extraits ainsi que les liens candidats calculés par l'approche. Nous exécutons notre stratégie d'amélioration dans un processus itératif;
- processus et étapes : notre approche consiste à exécuter des étapes spécifiques qui permettent de :
    - classifier les documents par thème,
    - extraire les artéfacts par thème,
    - calculer les similarités entre les artéfacts.

.

4.4.2 Outils Gensim

Nous présentons ici la librairie Gensim[18], qui permet d'appliquer les techniques de TAL pour extraire les connaissances à partir des exigences textuelles. Nous avons choisi Gensim en raison de la richesse des possibilités offertes pour le traitement du texte, le plongement des mots et les modèles thématiques par rapport aux autres librairies, telles que Scikit[19] et le langage R[20].

Gensim facilite l'utilisation des modèles de vecteurs des mots et des modèles de vecteurs des documents. Il permet de concevoir des modèles (ldamodel [21], word2vec[22], doc2vec[23] et FastText[24]) pour traiter des fichiers textes de grande taille. Ce traitement présente l'avantage de ne pas charger la totalité du texte dans la mémoire, ce qui offre une meilleure performance d'entrainement et de tests pour les modèles. C'est une librairie qui facilite la découverte des structures sémantiques en analysant les mots, les phrases ou les documents.

Nous construisons un corpus ou une collection de documents textuels afin de produire une représentation vectorielle du texte. Nous pouvons extraire par la suite un modèle

---

[18] https://radimrehurek.com/gensim/

[19] https://scikit-learn.org/

[20] https://www.r-project.org/

[21] https://radimrehurek.com/gensim/models/ldamodel

[22] https://radimrehurek.com/gensim/models/word2vec

[23] https://radimrehurek.com/gensim/models/doc2vec

[24] https://radimrehurek.com/gensim/models/fasttext

sous forme d'algorithmes pour créer des représentations différentes qui contiennent plus de sémantique.

L'architecture de Gensim supporte les concepts et les composantes suivantes (Rehurek & Sojka, 2010) :

- corpus : c'est une collection de documents numériques que nous utilisons comme entrée (*input*) afin d'inférer leur structure latente et leurs thèmes. Le corpus facilite le calcul de similarité sémantique entre les documents. La structure latente inférée permet d'assigner des thèmes pour les nouveaux documents sans aucune intervention humaine;
- vecteur : afin d'inférer la structure latente dans le corpus, le vecteur facilite la représentation du texte dans un format numérique et permet l'application de formules mathématiques. Un vecteur représente un seul document, les entrées montrent les occurrences des mots dans le corpus. La dimension du vecteur représente le nombre d'entrées dans le dictionnaire. Si le corpus contient *n* mots uniques alors chaque document s'attachera à un vecteur de dimension *n*;
- modèle : un modèle représente un terme abstrait qui consiste à transformer une représentation d'un document vers une autre. Gensim permet de représenter un document par un vecteur. Le modèle représente la transformation d'un vecteur vers un autre qui contient plus de sémantique;
- transformation : la transformation s'effectue à l'aide d'un ensemble d'algorithmes, tel que l'allocation latente de Dirichlet et la fréquence de termes. Cette dernière permet de représenter comment un mot peut avoir une grande valeur TF-TDF [25]. Elle permet également d'identifier la meilleure représentation du document ou d'une catégorie de documents. La fréquence des termes montre le

[25] https://radimrehurek.com/gensim/models/tfidfmodel

nombre de fois qu'un mot apparait dans un document et la fréquence des documents représente la fraction de documents dans lesquels le mot apparait. Le calcul s'appuie sur les statistiques qui couvrent tout le corpus;

- similarité : Gensim permet de déterminer la similarité entre deux documents ou entre deux représentations d'un texte. Elle consiste à calculer la similarité entre mots, phrases, paragraphes ou documents. La similarité cosinus détermine la similarité entre deux objets représentés par des vecteurs (Sidorov, Gelbukh, Gómez-Adorno, & Pinto, 2014).

4.4.3 Vue globale

Notre objectif consiste à établir, à partir des exigences textuelles, les correspondances entre les artéfacts qui appartiennent à un système d'information. Voici les trois tâches cognitives qui nous permettent d'atteindre cet objectif :

1. classifier les exigences : à partir d'un ensemble de documents relatifs aux exigences, nous effectuons la classification des documents en appliquant le modèle ALD;
2. extraire les artéfacts : à partir des documents classifiés, nous effectuons l'extraction des artéfacts pertinents pour les niveaux d'abstraction que nous ciblons (exigences, cas d'utilisation, processus d'affaires, entités conceptuelles et cas d'essai) en appliquant le modèle ALD;
3. établir les correspondances : calculer les similarités entre les artéfacts qui appartiennent au même niveau d'abstraction et entre les artéfacts qui appartiennent à des niveaux d'abstraction différents.

Nous commençons par classifier les exigences en documents ou paragraphes qui contiennent des informations relatives aux exigences, aux cas d'utilisation, aux processus d'affaires, aux entités conceptuelles et aux cas d'essai. Nous analysons par la suite les documents qui appartiennent à chaque classe pour en extraire les éléments pertinents et établir les correspondances entre ces artéfacts.

Nous présentons une approche qui permet d'établir les correspondances entre les artéfacts d'un système en appliquant les méthodes et algorithmes suivants :

- l'allocation latente de Dirichlet (ALD) : c'est un modèle qui permet de déterminer les thèmes extraits à partir des exigences. Établir les correspondances passera par une classification non supervisée des exigences et une extraction des artéfacts en adoptant une approche itérative d'ajustement;
- l'identification des entités : c'est une technique qui permet d'extraire les éléments pertinents du texte en s'appuyant sur sa structure grammaticale;
- le plongement des mots et des documents (word2vec et doc2vec) : ces modèles permettent d'utiliser les représentations vectorielles des mots, des phrases et des paragraphes et d'extraire les similarités entre les artéfacts et des modèles prédéfinis.

Notre approche s'appuiera sur une stratégie d'amélioration qui consiste à introduire une rétroaction de la part d'un spécialiste pour chaque résultat produit par le modèle (classification des exigences, extraction des artéfacts et établissement des correspondances)

Le tableau 4.3 montre la correspondance entre les tâches et les algorithmes. Nous appliquons l'ALD pour la classification et l'extraction des artéfacts et nous adoptons les vecteurs de mots et documents pour le calcul de similarité.

Tableau 4.3 Approche pour la classification des exigences et le calcul de similarité entre les artéfacts

Tâche cognitive	ALD	Identification des entités	Word2vec/doc2vec
Classifier les exigences	x		
Extraire les artéfacts		x	
Établir les correspondances entre les artéfacts			x

Notre approche s'appuie sur l'analyse des exigences textuelles. Nous utilisons les mots et les phrases en les convertissant en identifiants numériques. Chaque mot correspond à un identifiant unique dans le corpus. Le dictionnaire relie chaque mot à son identifiant. Gensim permet de créer le dictionnaire en convertissant le texte (documents, phrases et mots) en un corpus. Le dictionnaire sert à créer un corpus ou un sac de mots (*Bag-Of-Words*). La librairie considère le dictionnaire et le sac de mots comme des données en entrée pour les modèles que nous construisons pour extraire les thèmes, les artéfacts et les similarités sémantiques.

Les exigences se présentent sous différentes formes. Nous pouvons traiter un ou plusieurs fichiers textes qui ont des tailles petites, moyennes ou grandes. La librairie offre trois formes de texte :

- des phrases stockées dans des variables natives du langage Python[26];
- un fichier texte de petite ou grande taille;
- des fichiers textes multiples.

Par la suite, nous marquons les mots à l'aide des étiquettes (partie du discours). Nous considérons le corpus comme une collection ou un ensemble de documents sous forme de sac de mots. Pour chaque document, un corpus contient les identifiants de mots et leur fréquence.

Nous effectuons le prétraitement nécessaire pour préparer les documents pour la construction et l'entrainement des modèles. Le corpus contient du texte qui traite ou décrit les exigences, les cas d'utilisation, les processus d'affaires, les entités conceptuelles et les cas d'essai. Notre approche requiert des documents qui contiennent des paragraphes diversifiés et qui englobent des connaissances relatives aux artéfacts ciblés par la thèse.

Ensuite, Asuncion, Asuncion, & Taylor (2010) ont appliqué l'ALD pour établir les correspondances entre les artéfacts d'un système d'information. Ils ont enregistré les actions des développeurs au moment de leur exploration des documents. Un développeur qui travaille sur une partie du code ouvre plusieurs documents relatifs à la conception et aux cas d'essai. Leur approche consiste à enregistrer automatiquement le comportement du développeur lors du processus de développement. L'approche permet d'apprendre un modèle de thème probabiliste à travers les artéfacts. Ces

[26] https://www.python.org/

modèles facilitent la catégorisation sémantique et la visualisation des thèmes extraits. L'approche a permis d'explorer l'architecture du système et les thèmes ainsi que les artéfacts pertinents et les composantes d'architecture. H.Asuncion, A.Asuncion, & Taylor (2010) ont présenté une méthode qui permet d'établir les correspondances sans enregistrer les actions des développeurs . Ils ont appliqué l'ALD pour représenter tous les éléments sous forme de vecteur de poids de thème latent (Hayes & al., 2007). La différence entre les deux modèles réside dans la transformation de l'espace de mots clés pondéré en un espace de thème latent pondéré. Les liens entre les exigences et les éléments de conception sont calculés à l'aide de la distance euclidienne en s'appuyant sur un seuil qui détermine la pertinence du lien. De plus, Oliveto, Gethers, Poshyvanyk, & De Lucia (2010) ont affirmé que l'ALD est un modèle probabiliste génératif qui consiste à modéliser dans un mélange aléatoire à travers des thèmes latents. Ils ont utilisé la distance de Hellinger, qui consiste à calculer une similarité symétrique mesurée entre deux distributions de probabilités.

La figure 4.2 montre une vue globale de l'approche proposée avec les trois tâches :

1) classifier les exigences;
2) extraire les artéfacts;
3) établir les correspondances.

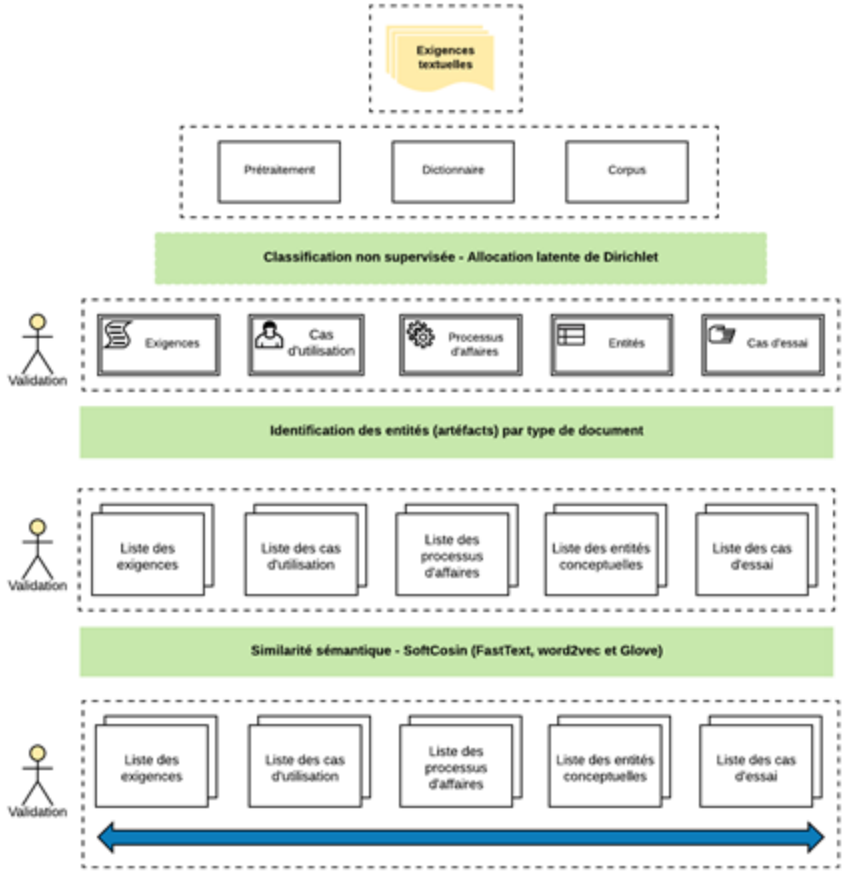


Figure 4.2 Vue globale de l'approche proposée

4.4.4 Classifier les exigences

Les analystes et concepteurs rédigent les exigences à l'aide d'un langage naturel, tel que l'anglais ou le français. Ces exigences reflètent plusieurs aspects du système à concevoir (exigences, cas d'utilisation, processus d'affaires, entités conceptuelles et cas d'essai). Nous analysons le texte rédigé en appliquant le modèle ALD à l'aide d'un corpus produit à partir des données textuelles en entrée. Nous passons par une étape de prétraitement des données, nous évaluons le résultat de la classification et nous visualisons les thèmes identifiés.

Nous effectuons l'expérimentation de notre approche en l'appliquant sur un ensemble de données relatives aux exigences (Ferrari, Spagnolo, & Genesis, 2017) ainsi que sur des exigences rédigées dans un contexte réel. Cet ensemble contient différents documents textuels que les analystes et concepteurs ont rédigés dans le cadre du développement des systèmes d'information. Nous illustrons l'approche à l'aide d'un extrait de ce même ensemble de données. La première étape consiste à préparer les exigences et à extraire des paragraphes et des phrases à partir de ces documents. La figure 4.3 montre un exemple de texte d'une exigence que nous utilisons pour décrire notre approche.

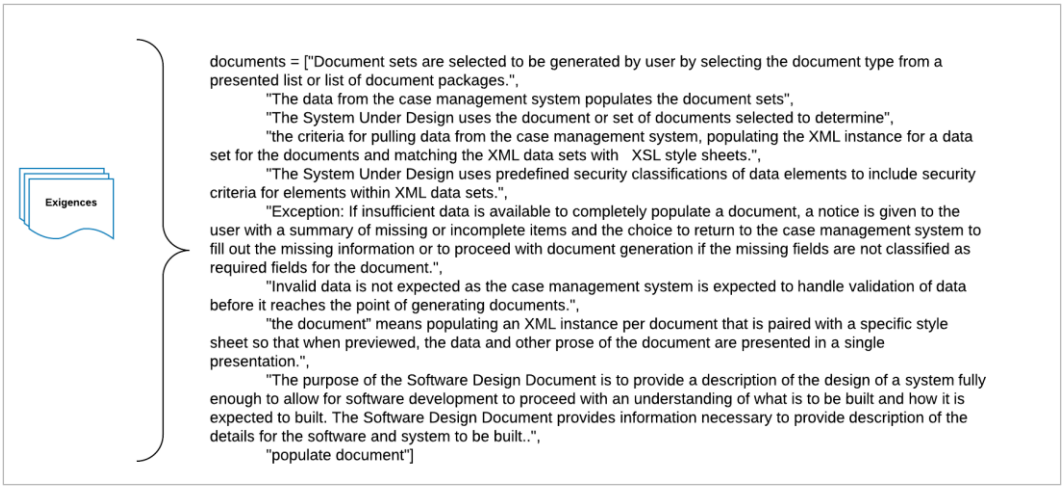


Figure 4.3 Exemple d'un extrait de texte à partir des exigences

Afin de préparer le corpus pour la construction des modèles, nous effectuons un prétraitement sur l'ensemble des données textuelles. Le prétraitement des données permet de produire une classification de meilleure qualité et d'éviter la complexité computationnelle (Niyogi & Pal, 2019). Le prétraitement consiste à appliquer la lemmatisation et la racinisation et à retirer les mots vides (*stopwords*) et non pertinents pour notre modèle.

La procédure de racinisation (*Stemming*) consiste à remplacer les mots par leur racine afin de réduire la dimension du corpus. Par exemple, cette technique remplace les mots

« analytics », « analytical », « analysis » par la racine « analytic ». Une racinisation excessive produira une précision faible et une racinisation restreinte produira un faible rappel (Niyogi & Pal, 2019).

Les mots vides représentent les mots non signifiants dans les phrases ainsi que les mots les plus fréquents (exemple : « the », « is », « at », « which »). Nous bonifions cette procédure à l'aide d'une intervention humaine qui permet d'ajouter des mots sélectionnés et jugés non pertinents par un analyste ou un acteur du système. Dans notre exemple, nous constatons que le mot « documents » apparait plusieurs fois dans toutes les phrases et tous les documents, ce qui le rend non pertinent pour appartenir à un thème.

Le prétraitement permet également d'appliquer l'étiquetage[27] (verbe, nom…) des mots en affectant des étiquettes à chacun et de nettoyer le texte en retirant la ponctuation et les caractères indésirables (virgule, numéro…) (Sohrabi, Vanani, & Shineh, 2018).

[27] https://www.ling.upenn.edu/courses/Fall_2003/ling001/penn_treebank_pos.html

La figure 4.4 montre un exemple d'étiquetage d'une phrase extraite de l'ensemble de données (Ferrari, Spagnolo, & Genesis, 2017).

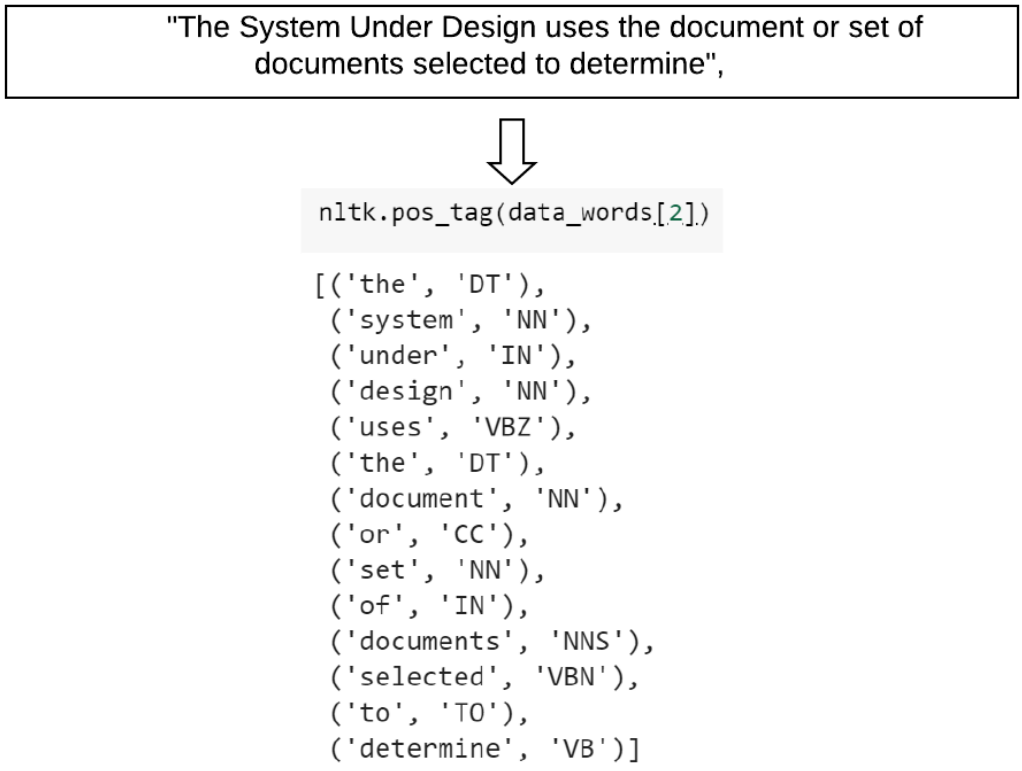


Figure 4.4 Exemple d'étiquetage du texte

Le prétraitement consiste également à construire des modèles Bigramme et Trigramme, c'est-à-dire des modèles statistiques qui aident à prendre en considération le contexte des mots (Koshy & Padmajavalli, 2018). Le Bigramme représente deux mots qui cohabitent fréquemment dans un document et le Trigramme représente trois mots qui cohabitent fréquemment dans un document.

Cette étape consiste à produire une représentation numérique du texte. La conversion des mots en identifiants numériques permet de générer une représentation vectorielle du texte (Koshy & Padmajavalli, 2018). Chaque mot correspond à un identifiant numérique unique; et chaque document peut s'encoder comme un vecteur à taille fixe qui désigne la taille du vocabulaire relatif aux noms connus. La valeur de chaque position dans le vecteur contient le nombre ou la fréquence de chaque mot dans le document encodé. Nous utilisons le modèle TF-IDF à la place des fréquences de termes. C'est un modèle qui permet de représenter les termes selon leur importance dans les documents. Afin d'identifier les termes les plus importants, nous produisons une fréquence de terme (TF) et une fréquence inversée du document (IDF) pour chaque document (d) qui contient des termes (t) :

- si un mot apparait fréquemment dans un document, il obtiendra un score élevé;
- si un mot apparait dans plusieurs documents, alors il ne correspondra à aucun identifiant unique et par conséquent, obtiendra un score faible.

Considérons la représentation de l'extraction faite à partir des documents de l'exemple de la figure 4.1 :

- doc1 = « The data from the case management system populates the document sets »;
- doc2 = « The System Under Design uses the document or set of documents selected to determine ».

Le tableau 4.2 montre une représentation sous forme de matrice des termes dans les deux documents.

Tableau 4.4 Matrice des termes des documents

	Termes							
Documents		Data	Case	Management	System	…	document	sets
	Doc1	1	1	1	1	…	1	1
	Doc2	0	0	0	1	…	2	1

Nous classifions les exigences en utilisant la modélisation des thèmes (*Topic Modeling*). C'est une approche qui facilite l'analyse du texte non classifié. Un thème se compose d'un ensemble de mots qui cohabitent fréquemment (Sohrabi, Vanani, & Shineh, 2018). H.Asuncion, A.Asuncion, & Taylor (2010) ont décrit la modélisation des thèmes comme une technique d'apprentissage machine qui permet d'inférer automatiquement la sémantique des thèmes à partir d'un corpus.

À partir des documents prétraités, nous commençons par créer le dictionnaire et le corpus requis pour la construction du modèle ALD. Nous montrons dans la figure 4.5 un exemple du dictionnaire (étiquette) et du corpus (identifiant numérique des mots).

```
print (id2word)
Dictionary(138 unique tokens: ['document', 'generate', 'list', 'package', 'present']...)
print (corpus)
[[(0, 3), (1, 1), (2, 2), (3, 1), (4, 1), (5, 2), (6, 1), (7, 1), (8, 1)], [(0, 1), (6, 1), (9, 1), (
```

Figure 4.5 Exemple d'un dictionnaire et d'un corpus produit par Gensim

Nous construisons les modèles avec les thèmes qui représentent les classes de documents que nous considérons dans notre travail. Nous classifions les documents textuels en cinq catégories (exigences, cas d'utilisation, processus d'affaires, entités conceptuelles et cas d'essai). Nous obtenons la contribution des mots pour chaque thème. Les mots présents dans tous les thèmes seront ajoutés à la liste des mots vides afin de les ignorer pendant la classification. Nous montrons dans la figure 4.6 la contribution des mots pour chaque thème. Le mot « document » est un candidat pour la liste des mots vides en raison de sa présence dans tous les thèmes.

```
pprint(lda_model.print_topics())
```

```
[(0,
  '0.091*"allow" + 0.047*"software" + 0.036*"design" + 0.036*"provide" + '
  '0.036*"build" + 0.027*"document" + 0.026*"system" + 0.026*"description" + '
  '0.015*"proceed" + 0.015*"information"'),
 (1,
  '0.246*"document" + 0.051*"set" + 0.051*"datum" + 0.044*"populate" + '
  '0.043*"user" + 0.032*"preview" + 0.032*"summary" + 0.024*"system" + '
  '0.016*"xml" + 0.016*"data"'),
 (2,
  '0.048*"attorney" + 0.048*"court" + 0.048*"clerk" + 0.032*"different" + '
  '0.032*"manager" + 0.032*"state" + 0.032*"signature" + 0.032*"digital" + '
  '0.032*"secure" + 0.032*"specific"'),
 (3,
  '0.037*"expect" + 0.022*"validation" + 0.022*"point" + 0.022*"invalid" + '
  '0.022*"reach" + 0.022*"handle" + 0.017*"generate" + 0.007*"case" + '
  '0.007*"management" + 0.007*"system"'),
 (4,
  '0.067*"document" + 0.058*"system" + 0.031*"design" + 0.031*"level" + '
  '0.031*"base" + 0.031*"software" + 0.022*"focus" + 0.022*"interaction" + '
  '0.013*"work" + 0.013*"modification"')]
```

Figure 4.6 Contribution des mots dans les thèmes produits par le modèle

Nous évaluons et visualisons les résultats du modèle ALD en montrant les éléments suivants :

- calculer la précision et le rappel du modèle;
- visualiser les thèmes dominants, le pourcentage de contribution ainsi que les mots-clés pour chaque thème (tableau 4.5);
- extraire les phrases représentatives par thème (tableau 4.6);
- extraire le nuage des mots par thème (figure 4.7);
- visualiser la répartition des termes pertinents à l'aide d'une échelle multidimensionnelle (figure 4.8).

Tableau 4.5 Pourcentage de la contribution des documents par thème

Document	Thème	% de contribution	Mots-clés	Texte
0	1	0,94	Document, set, datum, populate, user, preview	[document, set, select, generate, use]
1	3	0,53	Expect, validation, point, invalid	[invalid, datum, expect, case, management]
2	4	0,99	Document, system, design, level, base	[software, design, document, base, level]

Tableau 4.6 Texte représentatif par thème

Thème	% de contribution	Mots-clés	Texte représentatif
1	0,97	Software, design, build, document	[purpose, software, design, document, system, allow]
2	0,93	Expect, validation, generate, case, management	[handle, validation, expect, case]

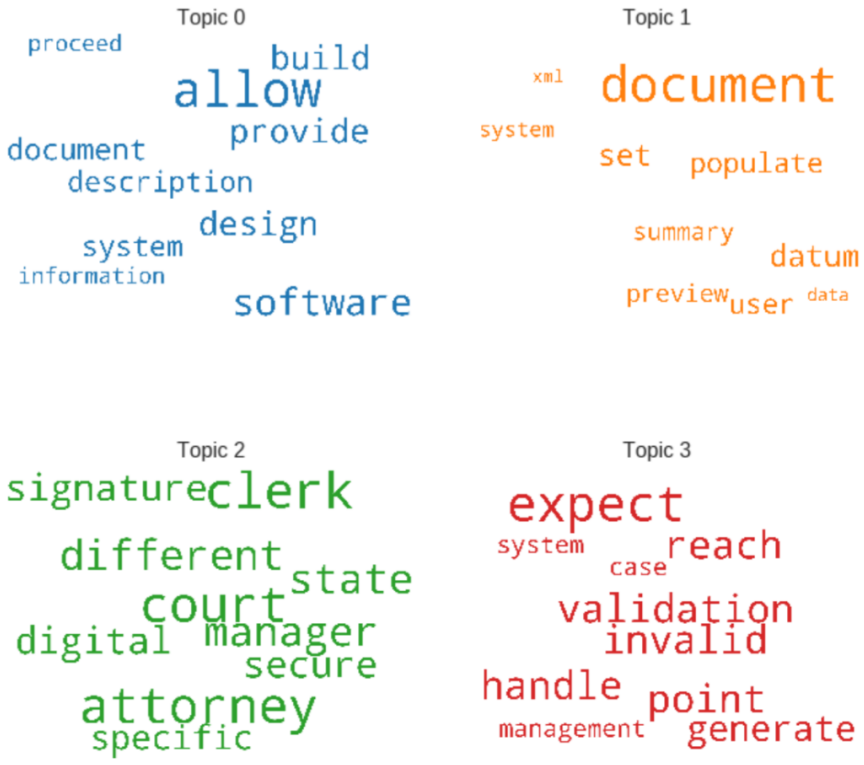


Figure 4.7 Représentation du nuage des mots par thème

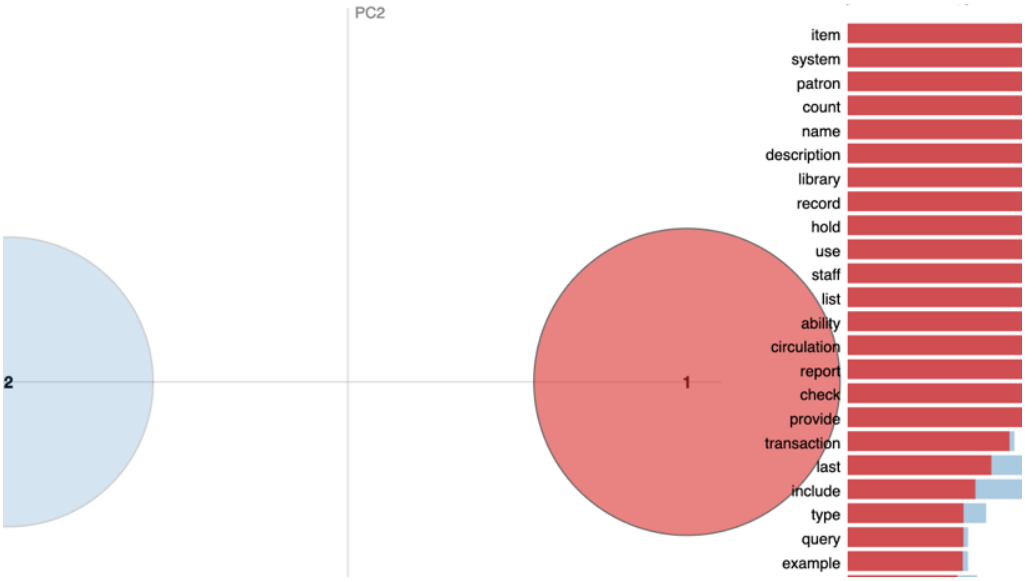


Figure 4.8 Répartition des thèmes et termes pertinents à l'aide d'une échelle multidimensionnelle

4.4.5 Extraire les artéfacts

À partir des documents classifiés, nous effectuons l'extraction des artéfacts en appliquant la reconnaissance des entités nommées. Nous utilisons les résultats des modèles Bigramme et Trigramme et l'étiquetage des phrases pour extraire la liste des artéfacts qui appartiennent à la classe en question.

De plus, Zeng (2008) a présenté les éléments linguistiques en concevant un modèle d'objet récursif. Il a décrit les modèles pour les mots, les verbes, les adjectifs et les phrases. Les mots représentent les unités fondamentales pour extraire la sémantique dans les phrases (Zeng, 2008). Afin d'analyser les exigences textuelles, Zeng (2008) a distingué les noms, les verbes, les adjectifs, les adverbes, les pronoms, les déterminants, les prépositions et les conjonctions. Il a proposé d'utiliser les noms pour désigner un objet, comme une personne, une place, une qualité, une idée ou une action, dans l'univers.

Zeng (2008) a interprété la composition d'une phrase comme suit :

- le nom désigne l'acteur qui a effectué une action;
- le verbe représente une action;
- les pronoms se substituent aux noms et aux phrases nominales;
- le verbe est employé pour indiquer une action de l'acteur sur un objet ou un état d'un objet.

Zeng (2008) a distingué les types de verbes suivants :

- verbes de lien (is, am, are);
- verbes transitifs, qui expriment une action entre deux objets (move, change…);
- verbes intransitifs, qui impliquent un seul objet (stay, fly, walk).

Ces trois types de verbes forment des relations de prédicats. À l'inverse des auxiliaires, qui couvrent le sens du verbe principal, ces types désignent des contraintes sur le verbe principal. Par ailleurs, Zeng (2008) a affirmé qu'un adjectif qualifie ou modifie un nom ou un pronom, alors qu'un adverbe modifie ou précise un verbe ou une phrase de verbe ou un adjectif. Les déterminants sont des mots nécessaires aux phrases, mais qui ne portent pas le sens en eux-mêmes. Les prépositions apparaissent toujours comme le premier mot d'une phrase prépositionnelle. Une conjonction est une classe de mots invariables qui relie deux ou plusieurs mots ou deux ou plusieurs phrases.

Les phrases, qui représentent un groupe de mots reliés, fonctionnent comme partie du discours. Zeng (2008) a distingué plusieurs types de phrases, tels que les phrases nominales, les phrases verbales et les syntagmes. Nous énumérons les modèles de Zeng (2008) dans le tableau 4.7.

Tableau 4.7 Modèles de phrases complètes Zeng (2008)

Modèle	Structure de phrase
Modèle 1	Sujet + verbe intransitif
Modèle 2	Sujet + verbe de lien + sujet complément
Modèle 3	Sujet + verbe transitif + objet direct
Modèle 4	Sujet + verbe transitif + objet indirect + objet direct
Modèle 5	Sujet + verbe transitif + objet direct + complément d'objet

Nous appliquons ces modèles de phrases pour extraire les artéfacts à partir des documents textuels. Nous visons l'extraction des listes relatives aux exigences, aux cas d'utilisation, aux processus d'affaires, aux entités conceptuelles et aux cas d'essai. L'extraction est effectuée à partir des documents déjà sélectionnés dans l'étape de classification. Nous appliquons notre approche sur des documents classifiés contenant les artéfacts à extraire. Un document de processus d'affaires contient uniquement les données relatives au thème processus d'affaires. Lors de l'expérimentation, un analyste validera manuellement la liste des artéfacts extraits afin d'adapter et d'évaluer la tâche d'extraction. Nous présentons dans ce qui suit les méthodes d'extraction par type de document.

Exigences : à partir des documents qui appartiennent au thème des exigences, nous effectuons l'extraction des exigences en sélectionnant les phrases qui correspondent aux étiquettes relatives aux modèles des phrases complètes du tableau 4.7.

Exemple

Texte : The system shall support an analyst level. The system shall support an guest level.

Étiquettes: `system/NN', b'support/VB', b'analyst/NN', b'level/NN']`

`[b'system/NN', b'support/VB', b'analyst/NN', b'level/NN', b'system/NN', b'support/VB', b'guest/NN', b'level/NN']`

Le résultat de l'extraction est le suivant :

1. system support analyst level;
2. system support guest level.

Cas d'utilisation : à partir des documents qui appartiennent au thème des cas d'utilisation, nous effectuons l'extraction des cas en utilisant les étiquettes produites par le modèle.

Nous considérons les étiquettes relatives à la structure suivante (Ilieva, 2007) : conjonction, sujet, prédicat, objet et conjonction. Nous appliquons la séquence décrite dans la représentation du tableau 4.8 (Ilieva, 2007).

Tableau 4.8 Structure de représentation des exigences textuelles (Ilieva, 2007)

Séq.	Conjonction	Sujet	Prédicat	Objet	Conjonction
1		Customer	Enter	Their pin number	and
2	if	The payment	Is	accepted	then
3		Money	Is retrieved		
4		Customer	Is informed		That
5		transaction	Was successful		

Ilieva (2007) a proposé un algorithme pour convertir les phrases passives en gardant l'ordre des objets (Sujet, prédicat, objet) :

- remplir les cases vides relatives aux sujets;
- la conjonction « and » entre deux phrases passives introduit une répétition qui peut montrer que les deux phrases partagent le même sujet;
- les phrases passives (be + participe passé) prennent le sujet de la phrase précédente;
- le participe passé s'attache à l'objet le plus proche. Si l'objet ne contient pas de texte, la règle consiste à utiliser le sujet de la phrase précédente;
- les concepts placés dans le sujet deviennent des objets directs;
- les objets indirects préservent leur position;
- le verbe passe de la forme passive à une forme active.

Nous considérons que chaque modèle (acteur + action) représente un artéfact qui appartient à la classe des cas d'utilisation et fera partie de la liste des artéfacts que nous utilisons pour établir les correspondances.

Exemple

Texte : 3.3.1 Analyst- Use Cases (“Enter Document into Workflow” for future update. Use case name: Create New Document Primary actor: Guest. use case, the actor's goal is to generate a document

Étiquettes: `[b'analyst/NN', b'use/NN', b'case/NN', b'document/NN', b'workflow/NN', b'future/NN', b'update/NN'] [b'use/NN', b'case/NN', b'name/NN', b'create/VB', b'new/JJ', b'document/NN', b'primary/JJ', b'actor/NN', b'guest/NN', b'use/NN', b'case/NN', b'actor/NN', b'goal/NN', b'be/VB', b'generate/VB', b'document/NN']`

Le résultat de l’extraction est le suivant :

1. document workflow future update;
2. create new document primary actor.

Processus d’affaires : à partir des documents qui appartiennent au thème des processus d’affaires, nous effectuons l’extraction des processus en utilisant les étiquettes produites par le modèle.

Le BPMN (Business Process Model and Notation) est un outil de modélisation normalisé des processus (Friedrich, Mendling, & Puhlmann, 2011). Cet outil se compose de quatre catégories d’éléments :

- objets de flux : c’est une catégorie qui contient les activités, les événements et les branchements;
- couloir (*Swimlane*) : c’est une catégorie qui contient les pistes et les corridors
- artéfacts : c’est une catégorie qui contient les données, les objets et les annotations textuels;
- objets de connexions : c’est une catégorie qui contient les flux de contrôle, les messages et flux de messages et les associations.

Un modèle BPMN correspond au résultat de l'analyse des descriptions textuelles. Nous nous intéressons à l'extraction des activités, des actions et des acteurs à partir du texte. Cette combinaison représente l'artéfact à extraire à partir des documents.

Les phrases qui appartiennent à la classe des processus d'affaires sont analysées et découpées en petites phrases. Friedrich, Mendling, & Puhlmann (2011) ont affirmé que l'extraction des objets s'effectue pour chaque action et se combine pour chaque objet. Les acteurs sont identifiés de la même manière et sont liés à leurs actions. Dans chaque phrase, nous mettons en évidence la relation de conjonction qui cause l'extraction des acteurs, actions et ressources. Nous ajoutons les phrases extraites à la liste des artéfacts que nous considérons pour établir les correspondances avec les autres artéfacts :

- les verbes représentent les actions;
- les sujets représentent les acteurs;
- les objets représentent les ressources.

Exemple

Texte : business work process: Analyst creates a new document

Étiquettes: `[b'business/NN', b'work/NN', b'process/NN', b'analyst/NN', b'create/VB', b'new/JJ', b'document/NN']`

Le résultat de l'extraction est le suivant :

1. analyst create new document.

Entités conceptuelles : à partir des documents qui appartiennent au thème des entités conceptuelles, nous effectuons l'extraction des entités en utilisant les étiquettes produites par le modèle.

Nous distinguons les artéfacts suivants (Afreena & Bajwa, 2011) :

- type objet : c'est un concept général qui expose un ensemble de caractéristiques pour distinguer les objets (exemple : student, library…);
- nom individuel : c'est un nom qualifié qui correspond à un seul objet (exemple : Montreal is a famous city in Canada);
- concept de verbe : c'est un concept qui spécifie les relations entre les concepts (exemple : library has books);
- caractéristique : c'est une propriété abstraite qui appartient à un objet (exemple : A name of the student is John).

Nous utilisons les règles d'extraction des entités suivantes (Afreena & Bajwa, 2011) :

- les noms communs (acteurs, objets thématiques…) représentent les types objets ou les concepts généraux (exemple : user, cup…). Ces noms correspondent aux classes dans la modélisation conceptuelle;
- les noms propres représentent les concepts individuels;
- les auxiliaires et les verbes d'action représentent les concepts de verbes. Pour construire les types de faits, nous pouvons utiliser la combinaison type d'objet/concept individuel + verbe, qui forme un fait unaire (exemple : vision system *senses*). La combinaison type d'objet/concept individuel + verbe + type d'objet forme un fait binaire (exemple : belt *conveys* part);
- les faits binaires peuvent révéler les faits de type association. L'exemple « belt conveys part » montre une association binaire entre « belt » et « part ».

Nous nous intéressons uniquement à l'extraction des entités et des associations comme artéfacts qui appartiennent à la classe des entités conceptuelles.

Exemple

Texte : The analyst is responsible for supporting the customer needs

Étiquettes : `[b'analyst/NN', b'be/VB', b'responsible/JJ', b'support/VB', b'customer/NN', b'need/VB']`

Le résultat de l'extraction est le suivant :

1. entité : analyst, customer;
2. relation : support.

Cas d'essai : à partir des documents qui appartiennent au thème des cas d'essai, nous effectuons l'extraction des cas en utilisant les étiquettes produites par le modèle.

Nous procédons à l'extraction des cas d'essai en utilisant des mots-clés. Yue, Ali, & Zhang (2015) ont introduit la modélisation restreinte des cas d'essai RTCM (*Restricted Test Case Modeling*). C'est une modélisation qui consiste à employer des mots-clés pour extraire les connaissances relatives aux cas. Zhang (2015) ont adopté les mots « VALIDATES THAT » et « VERIFY THAT » comme modèle pour la reconnaissance des essais.

Nous bonifions le principe d'extraction des mots-clés (Yue, Ali, & Zhang, 2015) en nous appuyant sur les mots similaires sémantiquement aux mots-clés existants. Nous parcourons le texte en effectuant une extraction des phrases à l'aide du modèle (Yue, Ali, & Zhang, 2015) ainsi que de tout autre mot analogue à ceux du modèle. Nous enrichissons la liste des mots-clés avec les nouveaux mots similaires découverts dans le texte.

Exemple

Texte : Verify that the document is created

Check that the new document is not empty

Étiquettes : `[b'verify/NN', b'document/NN', b'be/VB', b'create/VB']`

`[b'check/NN', b'document/NN', b'be/VB', b'create/VB']`

Le résultat de l'extraction est le suivant :

1. verify document create;
2. check document create.

4.4.6 Établir les correspondances

Nous établissons les correspondances en calculant la similarité sémantique entre les artéfacts. Les artéfacts se présentent sous forme de mots ou de phrases. Le calcul de la similarité des objets spécifiques fait partie des tâches de base de plusieurs méthodes pour les problèmes de TAL (Sidorov, Gelbukh, Gómez-Adorno, & Pinto, 2014). En effet, Sidorov, Gelbukh, Gómez-Adorno, & Pinto (2014) ont affirmé que la similarité du texte joue un rôle important pour des cas de détection de modèle (*pattern*) ou d'analyse des sentiments. Le modèle vectoriel constitue le moyen le plus utilisé pour représenter les objets. Ces objets sont représentés sous forme de vecteurs numériques relatifs aux fonctionnalités. Les valeurs textuelles symboliques correspondent à des valeurs numériques.

Les représentations distribuées des mots (Mikolov, Chen, Corrado, & Dean, 2013) ont connu un grand succès en appliquant une similarité cosinus entre les vecteurs de ces mots. Ce calcul est directement relié aux mots en commun entre les deux objets. Dans le cas d'absence de mots en commun entre les deux vecteurs, cette mesure est égale à zéro. Cependant, les mots qui sont reliés sémantiquement peuvent présenter des similitudes même si la mesure est égale à zéro.

La mesure cosinus pour les modèles vectoriels indique la similarité entre deux objets. Elle est calculée en tant que produit normalisé des deux vecteurs. La valeur est normalisée à l'aide de vecteurs de longueur euclidienne unitaire. Pour les valeurs positives, le cosinus est compris entre 0 et 1. Sidorov, Gelbukh, Gómez-Adorno, & Pinto (2014) ont généralisé la similarité cosinus en proposant la similarité cosinus soft (*soft cosin similarity*). C'est une mesure qui prend en considération des caractéristiques (*features*) semblables à celle d'un texte dans un modèle d'espace vectoriel. L'introduction des caractéristiques se compare à l'introduction de nouvelles dimensions dans le modèle vectoriel. Charlet & Damnati (2017) ont utilisé la similarité cosinus soft pour comparer du texte relatif aux questions dans un ensemble de données.

Ils ont proposé une approche qui combine différentes similarités textuelles. Afin de calculer cette mesure, ils ont introduit la matrice de relation du modèle cosinus classique dans un sac de mots. Cette mesure calcule les relations entre les mots en considérant les aspects lexicaux et sémantiques.

Nous pouvons établir les correspondances entre les artéfacts en calculant leur similarité. Deux artéfacts similaires sont des candidats pour une correspondance potentielle entre eux. Nous considérons les exemples suivants en appliquant les modèles FastText, word2vec et Glove[28]. Gensim permet de télécharger ces modèles préentrainés et fournit les librairies nécessaires pour les appliquer aux documents. Ce sont des modèles qui s'appuient sur des corpus de grande taille construits avec les données textuelles, telles que les données Wikipédia[29] et les données relatives aux nouvelles de Google[30].

La figure 4.9 montre les résultats de calcul de la mesure cosinus soft (*Softcosin*) qui compare trois exemples de phrases. Nous observons la différence des résultats de comparaison selon le modèle appliqué. La comparaison des phrases 2 et 3 donne une mesure plus grande avec le modèle FastText, ce qui s'explique par la richesse du modèle. Nous explorons les résultats de calcul entre les artéfacts en utilisant les trois modèles.

---

[28] https://nlp.stanford.edu/projects/glove/

[29] https://www.wikipedia.org/

[30] https://news.google.com/

```
--------------------------------------------------
Phrase 1 : The system shall support an analyst level
Phrase 2 : The system shall support an guest  level
Phrase 3 : analyst creates new document primary
--------------------------------------------------
--------------------------------------------------
FastText
--------------------------------------------------
Similarité entre phrases 1 et 2 : 0.8549301847055667
Similarité entre phrases 1 et 3 : 0.5358973445307854
Similarité entre phrases 2 et 3 : 0.5049069101090059
--------------------------------------------------
Word2vec
--------------------------------------------------
Similarité entre phrases 1 et 2 : 0.7303074689617659
Similarité entre phrases 1 et 3 : 0.2257040882387705
Similarité entre phrases 2 et 3 : 0.07461359667758512
--------------------------------------------------
Glove
--------------------------------------------------
Similarité entre phrases 1 et 2 : 0.7698262929277992
Similarité entre phrases 1 et 3 : 0.3603196270962705
Similarité entre phrases 2 et 3 : 0.24831155568092186
--------------------------------------------------
--------------------------------------------------
```

Figure 4.9 Exemple de calcul de similarité sémantique cosinus soft (*Softcosin*)

4.5 Conclusion

Nous avons présenté dans ce chapitre l'approche proposée pour établir les correspondances entre les artéfacts produits dans un système d'information. Nous avons décrit les techniques d'extraction des connaissances à partir du texte ainsi que les outils que nous appliquons; Gensim est la librairie TAL principale utilisée dans notre travail. Notre approche consiste à classifier, à l'aide de l'ALD, les exigences et les documents de spécifications. La classification sera basée sur les catégories suivantes : les exigences, les cas d'utilisation, les processus d'affaires, les entités conceptuelles et les cas d'essai. Nous identifions les artéfacts à partir de chaque catégorie de documents en appliquant la reconnaissance des entités et les structures grammaticales des phrases. Nous adoptons des structures linguistiques spécifiques pour chaque classe de document. Nous établissons les correspondances entre les artéfacts en calculant la similarité cosinus soft (*softcosin*) pour mesurer la similarité. Nous mesurons la similarité pour :

- les artéfacts qui appartiennent à la même classe (exemple : deux processus d'affaires);
- les artéfacts qui appartiennent à des classes différentes (exemple : cas d'utilisation et une entité conceptuelle).

Dans le chapitre V, nous procédons à l'expérimentation et à l'évaluation de notre approche en l'appliquant à un ensemble de données relatives aux exigences et spécifications des systèmes d'information. Nous analysons les résultats en mesurant les performances pour l'ensemble de données utilisé.

CHAPITRE V

ÉVALUATION DE L'APPROCHE PROPOSÉE

5.1 Introduction

Aletras & Stevenson (2013) ont souligné l'importance et la popularité de la modélisation des thèmes pour la classification non supervisée des documents (Blei, Ng, & Jordan, 2003). Wei & Croft (2006) ont appliqué des méthodes intrinsèques pour évaluer la performance des modèles. Ils ont utilisé la précision pour les tâches d'extraction de l'information. Wei & Croft (2006) ont également adopté les méthodes statistiques pour mesurer la vraisemblance prédictive en calculant la perplexité (Wallach, Murray, Salakhutdinov, & Mimno, 2009).

Aletras & Stevenson (2013) ont affirmé que ce genre d'évaluations ne fournit pas les informations nécessaires pour expliquer le processus d'interprétation par les acteurs humains. Ils ont valorisé la difficulté d'identifier des thèmes cohérents par rapport aux termes qui les constituent. En effet, les thèmes présentent une difficulté d'interprétation et une non-utilité dans certains cas. Ceux qui correspondent à une réalité spécifique ont une grande importance dans la visualisation des collections de documents, nous donnons en exemple les documents relatifs aux exigences et spécifications textuelles rédigés lors du cycle de développement des systèmes. La visualisation des thèmes générés par le modèle permet d'explorer le contenu de ces documents en fournissant une vue globale sur la collection et sur les termes les plus fréquents qui composent chaque thème.

Chang, Gerrish, Wang, Boyd-Graber, & Blei (2009) ont distingué entre les modèles de vraisemblance et les modèles de probabilités de vraisemblance. Ils ont affirmé que les acteurs humains préfèrent la cohérence des thèmes produits par des méthodes qui appliquent les probabilités. Chang, Gerrish, Wang, Boyd-Graber, & Blei (2009) se sont intéressés aux mesures automatiques de cohérence des thèmes en les comparant aux jugements humains. La cohérence des thèmes se définit comme la moyenne sémantique entre les termes qui les composent (Newman, Lau, Grieser, & Baldwin, 2010).

De plus, lors de l'exploration des thèmes relatifs aux exigences, nous calculerons la cohérence et la perplexité afin de déterminer la validité des thèmes. Nous prenons en considération la meilleure combinaison de ces deux mesures et nous la validons à l'aide du jugement d'un spécialiste du domaine.

Nous identifions les correspondances entre les artéfacts extraits par thème en calculant la similarité sémantique. Nous évaluons les liens candidats à l'aide d'une similarité supérieure ou égale à un seuil de 0,7 (Marcus & Maletic, 2003). Nous utilisons par la suite le jugement d'un expert du domaine pour valider les liens candidats.

5.2 Rétroaction et jugement humain

La rétroaction humaine facilite l'évaluation et l'amélioration des modèles. Dans le cas de l'établissement des correspondances et de la traçabilité des exigences, la rétroaction aide à identifier les documents pertinents et non pertinents à partir de l'extraction automatique de l'information. Les chercheurs ont appliqué ce jugement pour évaluer les modèles vectoriels afin d'ajuster leur pondération. Pour une requête donnée, l'utilisateur identifie deux ensembles de documents : les documents pertinents et les documents non pertinents. Les deux ensembles sont disjoints, mais ne couvrent pas la totalité des documents extraits (Hayes & al., 2007). Nous appliquons cette rétroaction pour évaluer le modèle, elle consiste à valider les résultats par trois analystes d'affaire et un spécialiste du domaine. Son rôle peut être assumé par toute personne capable de représenter l'ingénieur des exigences; il effectue les tâches suivantes (IIBA, 2015) :

- comprendre les problèmes et les objectifs de l'entreprise;
- analyser les besoins et les solutions;
- faciliter la collaboration avec les parties prenantes;
- analyser les processus.

Dans le cadre de l'évaluation de notre modèle, l'analyste s'assure de la pertinence des résultats obtenus :

- un lien pertinent représente toute relation qui existe réellement entre deux éléments du document d'exigences;
- un lien non pertinent représente toute relation qui n'existe pas entre deux éléments.

Les résultats dépendent de l'expertise des analystes. Ces derniers connaissent le domaine couvert par le document d'exigences. Dans un cas réel, nous désignerions une équipe d'analystes qui pourrait se charger de cette tâche.

Dans le cadre de ce travail, nous simulons cette rétroaction en analysant en détail les documents que d'autres analystes ont rédigés. Nous appliquons également notre modèle sur des exigences rédigées par les analystes d'un partenaire industrie1. Ce dernier a mis ses ressources à notre disposition pour comprendre le domaine et pour faciliter l'évaluation et la rétroaction. Nous appliquons une rétroaction simulée et une rétroaction réelle.

La rétroaction permet de distinguer les liens pertinents et non pertinents (Hayes & al., 2007). Nous appliquons la précision et le rappel pour évaluer la précision du résultat obtenu par le modèle et la rétroaction. La précision mesure le pourcentage des documents pertinents, alors que le rappel mesure le pourcentage des documents pertinents extraits. Pour une collection de N documents, nous avons R documents pertinents par rapport à une extraction réelle effectuée par un analyste. Le modèle analyse n documents, il extrait r documents pertinents par rapport à l'extraction automatique.

La précision se calcule comme suit :

$$\text{précision} = \frac{r}{n}.$$

Le rappel se calcule comme suit :

$$\text{rappel} = \frac{r}{R}.$$

Une grande précision signifie que le modèle a extrait la plupart des liens pertinents et réels détectés par rapport aux documents extraits par la rétroaction. Un rappel très grand signifie que le modèle a extrait avec succès la plupart des liens réels détectés par rapport aux documents pertinents extraits par la rétroaction.
La fiabilité et la précision du modèle dépendent principalement de la rétroaction des analystes qui connaissent les systèmes. Ils appliquent le modèle comme support qui facilite la détection des liens pertinents et non pertinents.

5.3 Mesures d'évaluation des modèles

Nous nous intéressons aux thèmes découverts à l'aide de la modélisation des thèmes. La validité des thèmes dépend de la compréhension et du degré de cohérence par rapport à un jugement humain. Les chercheurs ont évalué ces modèles en utilisant des vocabulaires fixes établis par les experts du domaine. Nous prenons en considération deux mesures que nous utilisons pour évaluer les thèmes découverts par notre modèle :

- la perplexité (Wallach, Murray, Salakhutdinov, & Mimno, 2009);
- la cohérence (Newman, Lau, Grieser, & Baldwin, 2010).

5.3.1 Perplexité

La modélisation des thèmes consiste à générer les thèmes à partir d'un document. Chaque document dans une collection est modélisé comme une distribution multinomiale. Elle consiste à appliquer une généralisation à *S* sujets. Pour chaque sujet, elle applique une généralisation à *M* mots ou termes. À travers ces distributions, nous distinguons :

- un petit nombre de mots importants et pertinents (probabilité de vraisemblance élevée) par sujet;
- un petit nombre de sujets pertinents par document.

La perplexité mesure la probabilité de distribution relative aux mots et aux sujets. La meilleure distribution est celle avec la plus petite perplexité.
La perplexité se calcule à l'aide de la probabilité de vraisemblance *P* entre les mots *W*. En considérant le nombre total des mots *N*, la formule est la suivante :

$$PP(W) = e^{\frac{1}{N}\sum_{i=1}^{N} log(P(Wi))}.$$

5.3.2 Cohérence

Stevens, Kegelmeyer, Andrzejewski, & Buttler (2012) affirment que dans le contexte de la modélisation des thèmes, un corpus se compose de *D* documents qui contiennent *M* mots. Ils ont souligné deux types de relations à travers les thèmes, dans lesquelles le paramètre *T* représente le nombre de thèmes :

- une matrice (M, T) qui décrit la pondération de chaque mot dans chaque sujet. Les mots qui obtiennent la plus grande pondération se présentent comme les mots dominants;
- une matrice (T, D) qui décrit la pondération de chaque thème dans un document. Les thèmes qui obtiennent la plus grande pondération se présentent comme les thèmes dominants.

En considérant que les documents sont générés à l'aide d'un modèle probabiliste, l'ALD permet l'apprentissage des relations entre les mots, les thèmes et les documents (Blei, Ng, & Jordan, 2003). Ce modèle requiert l'existence d'un nombre fixe de thèmes utilisés à travers le corpus. Chaque thème ou sujet est associé avec une distribution multinomiale à travers un vocabulaire. Nous distinguons deux types de distributions :

- une distribution qui représente la probabilité d'apparition de chaque thème dans un document (thème dominant dans un document);
- une distribution qui représente la probabilité qu'un mot soit utilisé dans un thème (mot ou terme dominant dans un thème).

La cohérence des thèmes mesure le degré de similarité sémantique entre les mots les plus fréquents ou dominants dans un thème. Cette mesure aide à distinguer entre les thèmes interprétables sémantiquement et les thèmes produits par une inférence statistique.

La cohérence se calcule à l'aide de la formule suivante (Stevens, Kegelmeyer, Andrzejewski, & Buttler, 2012) :

$$coherence(V) = \sum_{(i,j) \in V} score(i, j, \epsilon).$$

V : représente l'ensemble de mots qui décrivent le thème.

ϵ : représente le facteur de lissage et d'approximation pour garantir le retour d'un nombre réel (égal à 1).

Nous considérons deux mesures de cohérence d'un thème (Stevens, Kegelmeyer, Andrzejewski, & Buttler, 2012). Les deux mesures calculent la cohérence d'un thème en considérant la somme de la similarité distributionnelle entre les mots fréquents d'un thème.

La mesure UCI (Newman, Lau, Grieser, & Baldwin, 2010) consiste à calculer la dépendance probabiliste mutuelle entre deux mots qui appartiennent à un thème. Cette mesure permet de calculer les probabilités des cooccurrences entre les mots dans un corpus externe. Elle permet d'effectuer une comparaison sémantique externe avec des sémantiques connues, telle que Wikipédia[31].

[31] https://en.wikipedia.org/wiki/

La formule d'UCI est la suivante :

$$\text{score}(v_i, v_j, \varepsilon) = \log \frac{p(v_i, v_j) + \varepsilon}{p(v_i)\ p(v_j)}.$$

La mesure UMass (Mimno, Wallach, Talley, Leenders, & McCallum, 2011) consiste à calculer la cooccurrence entre les mots en se basant sur le corpus original intrinsèque et non pas sur un corpus externe.
La formule d'UMass est la suivante :

$$\text{score}(v_i, v_j, \varepsilon) = \log \frac{D(v_i, v_j) + \varepsilon}{D(v_j)}.$$

Stevens, Kegelmeyer, Andrzejewski, & Buttler (2012) ont expérimenté les deux mesures, ils ont obtenu de meilleurs résultats pour la cohérence UCI. Notre évaluation consiste à appliquer la perplexité et la cohérence UCI et à comparer les résultats.

5.4 Application et validation de l'approche

Ferrari, Spagnolo, & Genesis (2017) ont exploré les documents publics relatifs aux exigences des systèmes d'information. Ils ont affirmé que leur ensemble de données fournit un balancement en matière de structure des exigences. Ils ont également présenté leur travail comme un outil puissant pour aider les chercheurs dans la tâche d'évaluation de leurs méthodes qui appliquent le TAL. Leur recherche peut supporter les tâches de traçabilité et de classification des exigences. Ils ont également affirmé que que jusque-là, la plupart des chercheurs utilisaient les exigences de la NASA. Les concepteurs ont rédigé ce document pour concevoir un instrument scientifique transporté par un satellite. Cet ensemble concerne un seul projet, ceci rend difficile la généralisation des expérimentations.

Les travaux de recherche partagent deux faiblesses (Ferrari, Spagnolo, & Genesis, 2017) :

- difficulté à reproduire les expérimentations;
- difficulté à généraliser les résultats.

De plus, nous utilisons les exigences rédigées dans un cadre professionnel pour des systèmes que les concepteurs ont développés pour des industries spécifiques. Nous appliquons notre approche sur dix documents d'exigences issus de huit sources distinctes :

- sept exigences que nous avons extraites à partir d'une base de documents (Ferrari, Spagnolo, & Genesis, 2017) qui regroupent des exigences extraites de plusieurs sources de données sur le web;
- trois documents rédigés par Iristel [32] . C'est une compagnie de télécommunication canadienne qui opère plusieurs systèmes liés à ses activités ainsi qu'à ses unités organisationnelles.

À partir de cet ensemble (Ferrari, Spagnolo, & Genesis, 2017), nous avons extrait les documents des exigences en sélectionnant sept documents. Ces documents décrivent des sujets courants pour lesquels nous pouvons simuler la rétroaction humaine. Nous avons ajouté trois documents produits dans un contexte réel (Iristel). Les analystes ont rédigé ces documents pour développer de nouveaux systèmes qui supportent leurs unités d'affaires. Les tableaux 5.1 et 5.2 fournissent une fiche descriptive par exigence choisie pour notre évaluation.

[32] www.iristel.com

Tableau 5.1 Fiche descriptive des exigences utilisées (Ferrari, Spagnolo, & Genesis, 2017)

Exigence	E1
Description	Ce document décrit les exigences fonctionnelles et non fonctionnelles pour un module d'administration d'une librairie.
Source	Galecia Group[33]
Exigence	E2
Description	Ce document décrit les exigences fonctionnelles pour un système virtuel qui facilite les échanges entre les professeurs et les étudiants.
Source	NJIT[34]
Exigence	E3

[33] https://galecia.com/https://galecia.com/

[34] http://www.njit.edu/http://www.njit.edu/

Description	Ce document décrit les exigences fonctionnelles pour un système de traitement des fichiers PDF qui facilite la fusion et le découpage d'un ou de plusieurs fichiers.
Source	PDFSam[35]
Exigence	E4
Description	Ce document décrit les spécifications fonctionnelles pour le journal de bord électronique des navires de pêche britanniques.
Source	UK Fishing vessels[36]
Exigence	E5
Description	Ce document décrit les exigences fonctionnelles pour un système logiciel relatif à l'interface graphique HATS-GUI.
Source	Sandia National Laboratories[37]
Exigence	E6

[35] https://pdfsam.org/

[36] https://www.gov.uk/environment/commercial-fishing-fisheries-and-vessels

[37] https://www.sandia.gov/

Description	Ce document décrit les exigences fonctionnelles pour un système virtuel VCD qui facilite les échanges entre les opérateurs économiques européens.
Source	PEPPOL[38]
Exigence	E7
Description	Ce document décrit les exigences fonctionnelles pour le développement d'un système relatif à une maison intelligente.
Source	HomeOwner inc.[39]

[38] https://peppol.eu/

[39] HomeOwner inc. - La plus grande chaîne national américaine de vente au détail répondant aux besoins des propriétaires.

Tableau 5.2 Fiche descriptive des exigences utilisées (Iristel inc.)

Exigence	E8
Description	Ce document décrit les exigences fonctionnelles pour un système d'orchestration applicatif.
Source	Iristel inc.
Exigence	E9
Description	Ce document décrit les exigences fonctionnelles pour un système d'échange entre les plateformes technologiques.
Source	Iristel inc.
Exigence	E10
Document	C'est un document qui décrit les exigences fonctionnelles pour automatiser des services spécifiques pour les clients.
Source	Iristel inc.

5.5 Expérimentations et évaluations

Notre évaluation empirique s'appuie sur les cinq ensembles de données relatives à des documents textuels. Les analystes ont produit ces documents afin d'initier les travaux de conception et de développement d'un système spécifique. En appliquant notre approche sur les cinq documents, nous évaluons les thèmes générés à l'aide des mesures de cohérence et de perplexité. Nous portons un jugement humain sur les résultats afin d'apporter les améliorations nécessaires pour adapter le nombre de thèmes ou valider les liens candidats.

Lors de l'étape de classification, nous explorons les thèmes de chaque document (exigences, cas d'utilisation, processus d'affaires, entités conceptuelles et cas d'essai), nous ajustons le nombre de thèmes pour chaque document à l'aide d'une rétroaction humaine. Cette rétroaction s'appuie sur les mesures calculées et la visualisation des résultats. Nous analysons les résultats à l'aide du visualiseur des thèmes LDAvis (Sievert & Shirley, 2014).

Lors du calcul de similarité, nous calculons la précision, le rappel et la F-mesure et nous construisons la matrice de confusion pour les résultats afin d'évaluer l'identification des correspondances entre les artéfacts. La précision représente la fraction des liens extraits et pertinents parmi tous les liens détectés. Le rappel représente la fraction des liens extraits et pertinents sélectionnés parmi tous les liens réels et sélectionnables. La F-mesure représente une moyenne harmonique entre la précision et le rappel (Baeza-Yates & Ribeiro-Neto, 2011).

Borg, Runeson, & Ardö (2014) ont distingué deux cas d'utilisation pour évaluer les traces entre les artéfacts :

- le premier consiste à sélectionner un artéfact et à rechercher ses liens candidats avec les autres éléments. Ce cas s'applique à l'extraction de l'information ayant pour but de trouver dans une grande collection de documents ceux qui répondent à un besoin d'information. Dans ce cas, les chercheurs favorisent la précision plutôt que le rappel. En effet, les analystes ont souvent besoin de reporter un seul élément (cas d'essai), ce qui produit un petit nombre de faux positifs par rapport aux liens candidats;
- le deuxième consiste à générer la totalité des correspondances. Par exemple, cette dernière comprend les liens entre *n* processus et *m* concepts. Et dans ce cas, le rappel apparait le plus adéquat pour évaluer le résultat de l'expérimentation.

5.6 Classifier les documents et extraire les artéfacts

Nous procédons à la classification des cinq exigences en explorant le nombre de thèmes par document. Nous calculons la cohérence et la perplexité pour chaque exécution afin de choisir le nombre de thèmes optimal pour chaque document. L'exécution consiste à appliquer un prétraitement et à déterminer un nombre de thèmes optimal, pour lequel nous validons la cohérence à l'aide d'un jugement humain. Nous comparons les résultats pour un maximum de neuf thèmes par document. Nous analysons le contenu de chaque thème et nous déterminons sa classe d'appartenance. Cette rétroaction humaine nous permet d'apporter les améliorations nécessaires pour adapter l'exploration et le dénombrement des thèmes. Le contenu de chaque catégorie se compose de mots, de phrases et de paragraphes. Les phrases se regroupent dans le cadre d'une exigence, d'un cas d'utilisation, d'un processus d'affaires, d'une entité conceptuelle ou d'un cas d'essai.

Nous nous intéressons aux éléments et artéfacts qui obtiennent un taux d'appartenance supérieur à 80 %. Afin de distinguer entre les classes de documents, nous nous appuyons sur les documents d'exigences et de spécification de la méthodologie Macroscope[40]. C'est un cadre de référence intégré qui propose des modèles de documents normalisés et qui facilite la rétroaction humaine en matière de modèles approuvés par la communauté du développement logiciel. Les professionnels l'utilisent

[40] https://macroscope.ca.fujitsu.com/fr/

comme référence pour la rédaction des exigences et des spécifications de conception et de développement des systèmes d'information.

5.6.1 Exigence E1

La première exécution montre que la meilleure cohérence est celle obtenue pour deux thèmes. L'annexe A contient le détail d'exécution pour chaque nombre de thèmes, les mesures de cohérence et de perplexité ainsi que les termes dominants. Voici le résultat de l'application du modèle ALD pour deux thèmes.

```
Perplexité        : -6.06
Cohérence         : 0.46
Les termes dominants par thème.
--------------------------------------------------
Thème 1 item system patron count name description library record hold use
Thème 2 pine report would library management priority req source process system
```

Nous observons que la meilleure cohérence est celle relative aux deux thèmes. La figure 5.1 montre la distance entre les deux thèmes ainsi que les termes dominants qui les composent.

En examinant les termes par thème, nous notons que le modèle regroupe des mots qui appartiennent à deux thèmes distincts. Nous constatons que les thèmes extraits contiennent des termes grâce auxquels nous pouvons distinguer les exigences et les processus. L'exemple suivant montre la différence entre deux phrases qui appartiennent aux deux classes.

Exigence: *management processes shall use a fully relational database back-end.*

Processus: *the system provides a last copy report that would single out a library systems last copies, to assist catalogers who need to edit WorldCat entries*.

Afin d'explorer plus de thèmes et d'essayer d'extraire les autres catégories ciblées par la thèse, nous évaluons l'existence d'un troisième et d'un quatrième thème. Le résultat de l'application du modèle ALD pour trois thèmes est le suivant :

```
Perplexité        : -6.11
Cohérence         : 0.43
Les termes dominants par thème
--------------------------------------------------
Thème 1 item count system name patron use library description record ability
Thème 2 pine report would management req priority source library process system
Thème 3 library system staff patron check description transaction template display
item
```

Nous identifions un troisième thème relatif à la conception et aux entités conceptuelles. La figure 5.2 montre la distance entre les deux thèmes ainsi que les termes dominants qui les composent. L'exemple suivant montre des phrases qui appartiennent à cette classe.

```
Conception: patron characteristics, transaction history, in-transit items report.
```

L'évaluation de quatre thèmes a démontré que le dernier représente une sous-variante supplémentaire de la classe des exigences. Les catégories relatives aux processus et aux concepts sont restées intactes.

Nous concluons que le nombre optimal de thèmes est de trois, même si cela ne correspond pas à la meilleure cohérence. Nous notons le rôle important de la rétroaction, qui a permis d'adapter et d'améliorer le modèle.

Pour cet exemple, nous n'avons pas pu découvrir explicitement les classes relatives aux cas d'utilisation et aux cas d'essai. Cependant, nous pouvons les extraire à partir des catégories existantes. Les processus d'affaires contiennent l'information relative

aux cas d'utilisation du système. En effet, un cas d'utilisation peut regrouper un ou plusieurs processus d'affaires. Nous pouvons aussi générer les cas d'essai : toutes les phrases relatives aux exigences et aux processus contiennent des extraits que nous pouvons employer pour extraire les cas d'essai. Sachant que le document de cette exigence ne dispose pas d'une section spécifique pour les essais, l'exemple suivant montre un cas d'essai qui peut être extrait à partir d'une exigence.

Exigence: *management Processes shall be accessible through a web-browser or a Windows-compatible client*".

Cas d'essai: *test case should be executed to validate the system using a web browser and client windows install.*

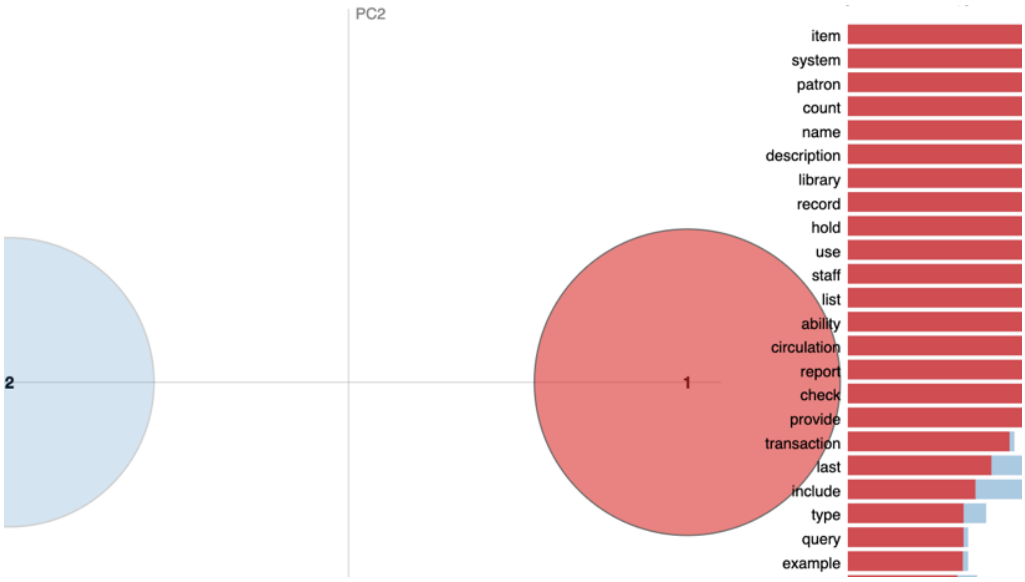


Figure 5.1 Distance et fréquence des termes pour les deux thèmes – E1

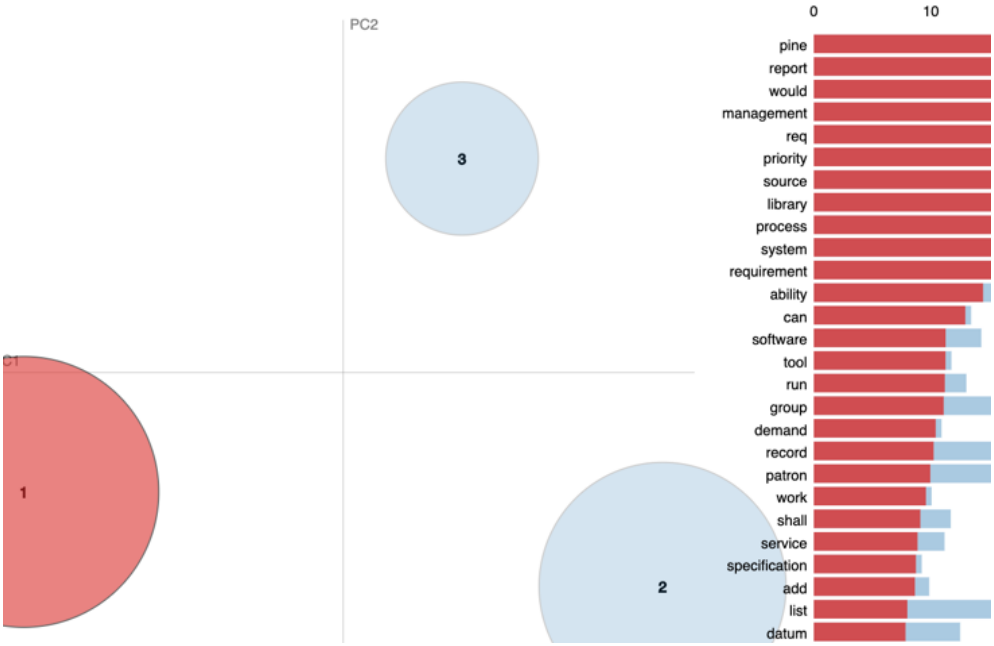


Figure 5.2 Distance et fréquence des termes pour les trois thèmes – E1

5.6.2 Exigence E2

L'exploration du nombre de thèmes a montré que la cohérence la plus intéressante est celle produite pour deux et neuf thèmes. Nous nous sommes intéressé aux deux modèles et nous avons choisi d'explorer plus en détail les neuf thèmes pour vérifier leur cohérence à l'aide d'un jugement humain. L'annexe B contient le détail d'exécution pour chaque nombre de thèmes, les mesures de cohérence et de perplexité ainsi que les termes dominants par thème.

Voici le résultat de l'application du modèle pour neuf thèmes.

```
Perplexité      : -5.92
Cohérence       : 0.48
Les termes dominants par thème :
---------------------------------------------------
Thème 1 video student audio feature professor able lecture document give real.
Thème 2 user show feature response interface new menu font choose different.
Thème 3 system user use instal browser must excel web software file.
Thème 4 space file would upload allow release section folder student document.
Thème 5 test file response select window show user student professor display.
Thème 6 user action file click button select window menu upload response.
Thème 7 requirement virtual user software description action functional log priority
ed.
Thème 8 virtual user exam space need req response window pop file.
Thème 9 user system req response must limit feature available show log.
```

Nous observons que les neuf thèmes fournissent une richesse d'information qui facilite l'exploration de ce document. Nous identifions le premier thème, qui regroupe les concepts; les thèmes deux et six regroupent les processus et les cas d'utilisation; le thème cinq contient des termes plus proches des cas d'essai. Le reste des thèmes se classent comme des exigences. Voici des exemples de phrases extraites du document pour chaque thème.

Exigence : *an uploading feature would be the only component available in the first release, which would aid in handing in assignments.*

Cas d'utilisation: *user will be able to select online users and invite them to join the conference.*

Concepts: *professor to hold a video lecture for the whole class.*

Processus: *to launch an application click on the application icon, the application will open up on a separate screen.*

Cas d'essai : *the file should be moved to the new destination folder and removed from the previous folder.*

Pour cet exemple, le modèle n'a pas permis de séparer les thèmes relatifs aux cas d'utilisation et aux processus d'affaires. La cohérence et la rétroaction humaine ont facilité le regroupement des termes dans des catégories compréhensibles par les équipes de conception du système. Malgré l'absence de connaissances préalables, le modèle nous a permis de classifier le document en neuf thèmes. La figure 5.3 montre la distance entre les thèmes et les termes dominants qui les composent.

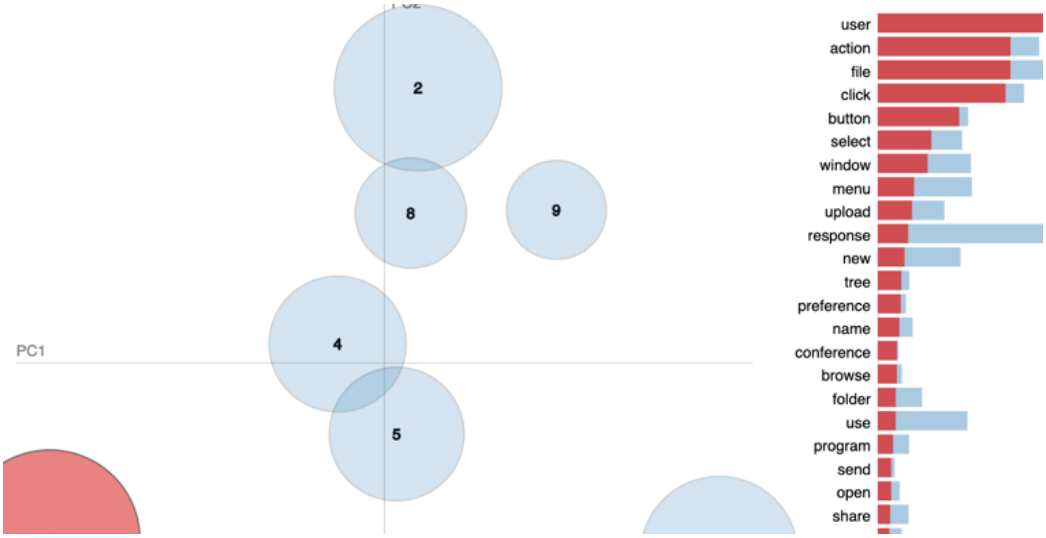


Figure 5.3 Distance et fréquence des termes par thème – E2

5.6.3 Exigence E3

Nous observons que la combinaison de la meilleure cohérence et de la rétroaction humaine a permis de consolider l'exploration automatique des thèmes. L'annexe C contient le détail d'exécution pour chaque nombre de thèmes, les mesures de cohérence et de perplexité ainsi que les termes dominants par thème.

Voici le résultat de l'application du modèle pour sept thèmes.

```
Perplexité      : -5.82
Cohérence       : 0.41
Les termes dominants par thème
--------------------------------------------------
Thème 1 file user document pdf select output page want version rotate
Thème 2 panel part log project consist gui follow purpose pdfsam message
Thème 3 use environment case java user save load file select page
Thème 4 user prefix number use stimulus_response_sequence level order start change move
Thème 5 requirement document pdf merge split software specification page functional user
Thème 6 feature system interface plugin application user gui screenshot setting pdfsam
Thème 7 split source description software page file program gpl license input
```

Nous notons que les sept thèmes fournissent plus de détails pour explorer cette exigence. Nous identifions le premier thème, qui regroupe les exigences et tests pour les fonctionnalités de base. Le deuxième thème regroupe les exigences relatives à l'interface utilisateur. Le troisième thème contient des cas d'essai. Le quatrième décrit les processus relatifs aux fichiers produits. Le cinquième regroupe les processus de manipulation des fichiers. Le sixième contient les concepts et attributs. Le septième décrit les processus relatifs aux fichiers en entrée. La figure 5.4 montre les termes dominants et les distances entre les thèmes.

Voici des exemples d'éléments extraits pour chaque thème.

Exigence: *users want an easy way to manipulate specific pages of a pdf document through a user friendly graphical interface with simple functions.*

Exigence: *the GUI Visual document composer panel consists of the following parts: Selection panel: The Selection panel is divided into two parts.*

Tests: *it has been tested on various versions of Microsoft Windows, GNU/Linux distributions and Mac OS X.*

Processus: *this prefix variable ensures unique output filenames and it's replaced with the current page number.*

Processus: *users can merge many pdf documents or subsections of them together.*

Concept: *user Classes and Characteristics.*

Processus: *the input file will be split at every page linked by the bookmarks of the selected level.*

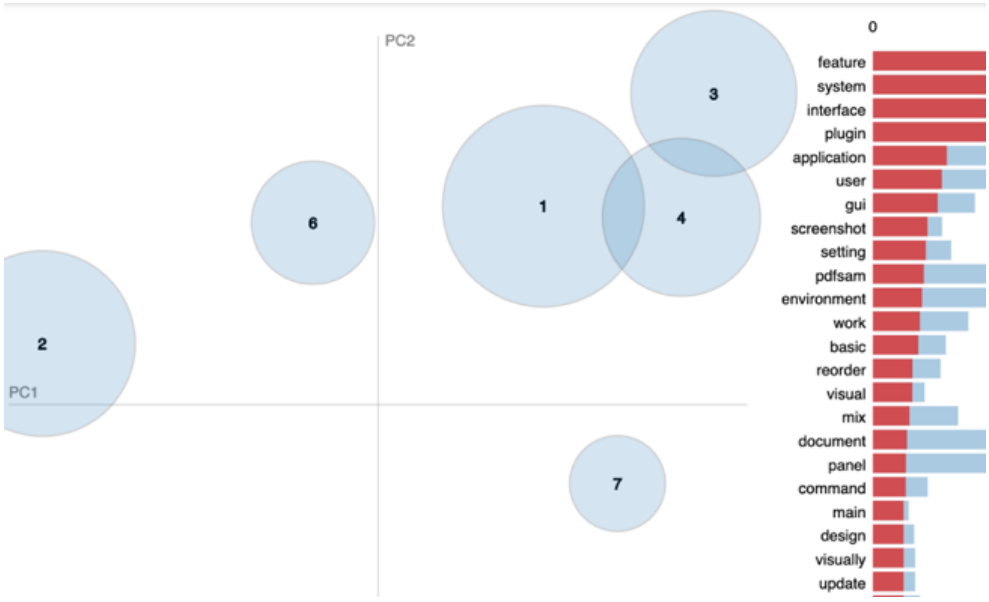


Figure 5.4 Distance et fréquence des termes par thème – E3

5.6.4 Exigence E4

Nous observons que la combinaison de la meilleure cohérence et de la rétroaction humaine a permis de consolider l'exploration automatique des thèmes. L'annexe D contient le détail d'exécution pour chaque nombre de thèmes, les mesures de cohérence et de perplexité ainsi que les termes dominants par thème.

Voici le résultat de l'application du modèle pour quatre thèmes.

```
Perplexité      : -5.61
Cohérence       : 0.61
Les termes dominants par thème
--------------------------------------------------
Thème 1 attribute declaration element vessel code fishing uk sub cif ers
Thème 2 must provide logbook product electronic elss message datum transmission
use
Thème 3 define format fix description pattern must content yyyy conform time
Thème 4 support feature product elss uk operation datum fishery fishing supplier
```

Nous notons que les quatre thèmes fournissent plus de détails pour explorer cette exigence. Nous identifions le premier et le troisième thème, qui regroupent les éléments conceptuels. Le deuxième et le quatrième thème regroupent les exigences. La figure 5.5 montre les termes dominants et les distances entre les thèmes.

Voici des exemples d'éléments extraits pour chaque thème.

Concept: *Ref No, Status, Notes.* Exigence: *product MUST only permit correction messages C to be sent only during a current trip up to the End.* Concept: *free form comments – a maximum of 500 O character.* Exigence: *users Product MUST correlate each Acknowledgment C message with the outgoing report and operation.*

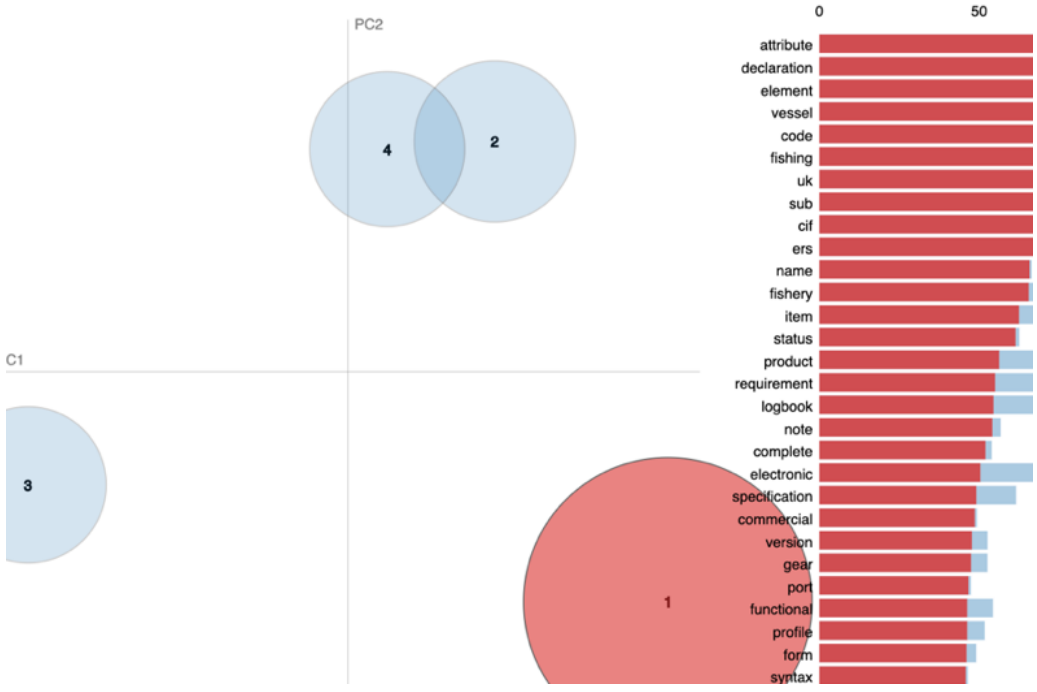


Figure 5.5 Distance et fréquence des termes par thème – E4

5.6.5 Exigence E5

Nous observons que la combinaison de la meilleure cohérence et de la rétroaction humaine a permis de consolider l'exploration automatique des thèmes. L'annexe E contient le détail d'exécution pour chaque nombre de thèmes, les mesures de cohérence et de perplexité ainsi que les termes dominants par thème.

Voici le résultat de l'application du modèle pour sept thèmes.

```
Perplexité      : -5.46
Cohérence       : 0.51
Les termes dominants par thème
--------------------------------------------------
Thème 1 tree search string sub sdt table match navigation criterion contain
Thème 2 use software requirement case specification end refer step continue scenario
Thème 3 node display sdt select graph user shall expand descendant correspond
Thème 4 hat gui shall user search new current enter directory figure
Thème 5 text window display cursor system shall keyboard srsreq location highlight
Thème 6 file program transformation hat pretty language target output print parse
Thème 7 application user select file srsreq shall editor hat system save
```

Nous notons que les quatre thèmes fournissent plus de détails pour explorer cette exigence. Le premier, le troisième, le cinquième et le septième regroupent des éléments qui décrivent les processus. Le deuxième thème contient des descriptions et des références relatives aux cas d'utilisation. Le quatrième et le sixième thème décrivent des exigences. La figure 5.6 montre les termes dominants et les distances entre les thèmes.

Voici des exemples d'éléments extraits pour chaque thème.

Processus: *system asks user for the criteria to search text.*
Cas d'utilisation: *preconditions: application has been selected (Refer to Use Case 3.*
Processus: *system displays the tree by expanding descendants of the node in the displayed graph.*

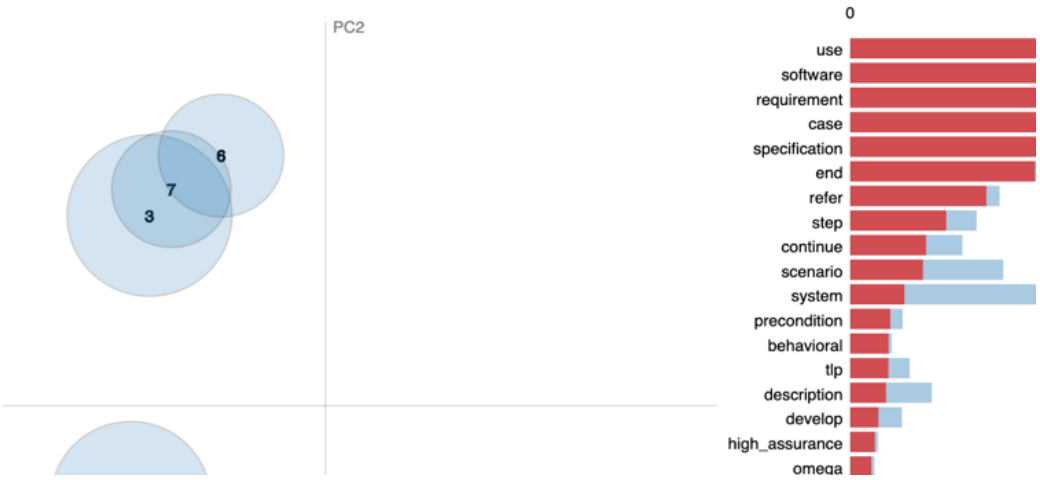


Figure 5.6 Distance et fréquence des termes par thème – E5

5.6.6 Exigence E6

Nous observons que la combinaison de la meilleure cohérence et de la rétroaction humaine a permis de consolider l'exploration automatique des thèmes. L'annexe F contient le détail d'exécution pour chaque nombre de thèmes, les mesures de cohérence et de perplexité ainsi que les termes dominants par thème.

Voici le résultat de l'application du modèle pour cinq thèmes.

```
Perplexité      : -6.48
Cohérence       : 0.58
Les termes dominants par thème
--------------------------------------------------
Thème 1 service provider national vcd economic operator level article attestation legal
Thème 2 national criterion european mapping need authority evidence attestation tender assessment
Thème 3 datum package vcd system evidence specific provide operator economic context
Thème 4 stage document requirement implementation define common concept different pilot model
Thème 5 requirement functional approval pende vcd peppol specification version deliverable document
```

Nous notons que les cinq thèmes fournissent plus de détails pour explorer cette exigence. Les éléments du premier, du deuxième et du cinquième thème couvrent les aspects généraux et détaillés du système. Le troisième thème décrit les cas d'utilisation et le quatrième regroupe des éléments conceptuels. La figure 5.7 montre les termes dominants et les distances entre les thèmes.

Voici des exemples d'éléments extraits pour chaque thème.

Exigence: *database hosted by a national service provider to receive an attestation by using a direct reference.* Cas d'utilisation: *this use case is similar to the use case "Create VCD simple package" of stage 2 but adds the necessary functionality regarding context specific data and merging existing VCD packages into the new VCD advanced package.* Concept: *quantity attributes, Personal details.*

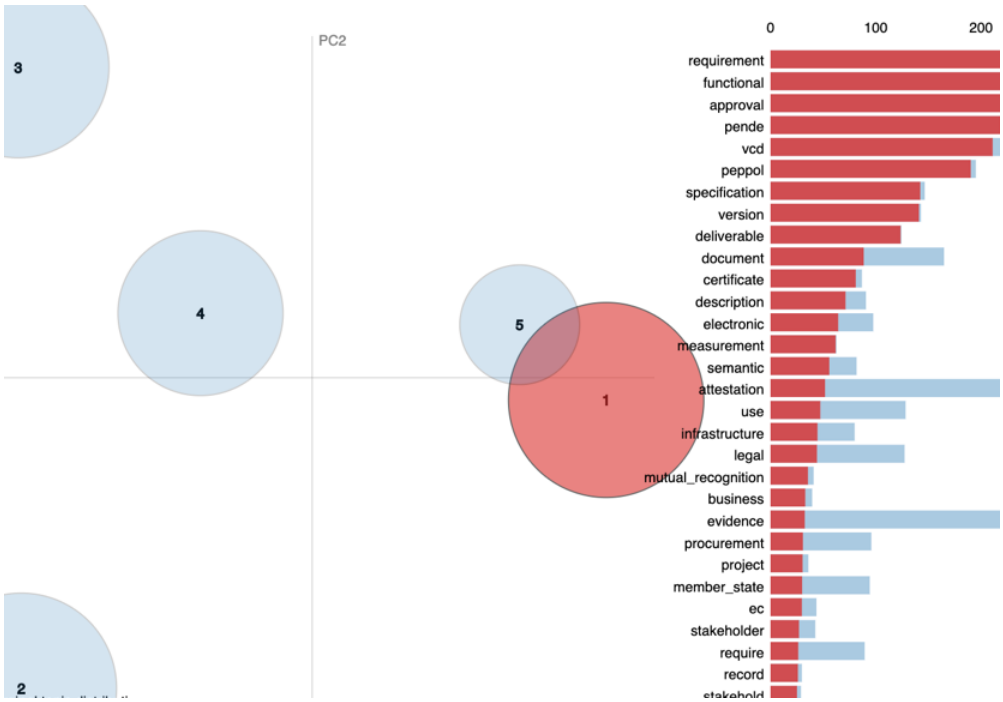


Figure 5.7 Distance et fréquence des termes par thème – E6

5.6.7 Exigence E7

Nous observons que la combinaison de la meilleure cohérence et de la rétroaction humaine a permis de consolider l'exploration automatique des thèmes. L'annexe G contient le détail d'exécution pour chaque nombre de thèmes, les mesures de cohérence et de perplexité ainsi que les termes dominants par thème.

Voici le résultat de l'application du modèle pour cinq thèmes.

```
Perplexité      : -5.96
Cohérence       : 0.41
Les termes dominants par thème
--------------------------------------------------
Thème 1 ieee shall temperature control home user system engineer humidity thermostat
Thème 2 shall sensor device home system digitalhome controller power user switch
Thème 3 home digital project srs system time plan use default set
Thème 4 requirement system security shall description user alarm use construction cycle
Thème 5 digitalhome system shall version homeowner requirement product digital specification document
```

Nous notons que les cinq thèmes fournissent plus de détails pour explorer cette exigence. Les cinq thèmes contiennent des exigences relatives aux sous-thèmes. Le premier thème décrit des exigences relatives aux températures supportées par le système. Le deuxième thème décrit la couverture de la transmission à l'intérieur de la maison. Le troisième thème contient des informations relatives à des exigences non fonctionnelles, telles que la sauvegarde et la restauration. Le quatrième thème traite les aspects de sécurité et le cinquième détaille les aspects relatifs à l'utilisateur du système. La figure 5.8 montre les termes dominants et les distances entre les thèmes.

Voici des exemples d'éléments extraits pour chaque thème.

Exigence: *the system shall support Fahrenheit and Celsius temperature values.*
Exigence: *the Gateway device shall operate up to a 1000-foot range for indoor transmission.*
Exigence: *the Digital Home System shall incorporate backup and recovery mechanisms.*
Exigence: *the system shall include security sound and light alarms.*
Exigence: *DigitalHome Planner shall provide a user with the capability to direct the system to set various preset home parameters (temperature, humidity, security contacts, and on/off appliance/light status) for certain time periods*.

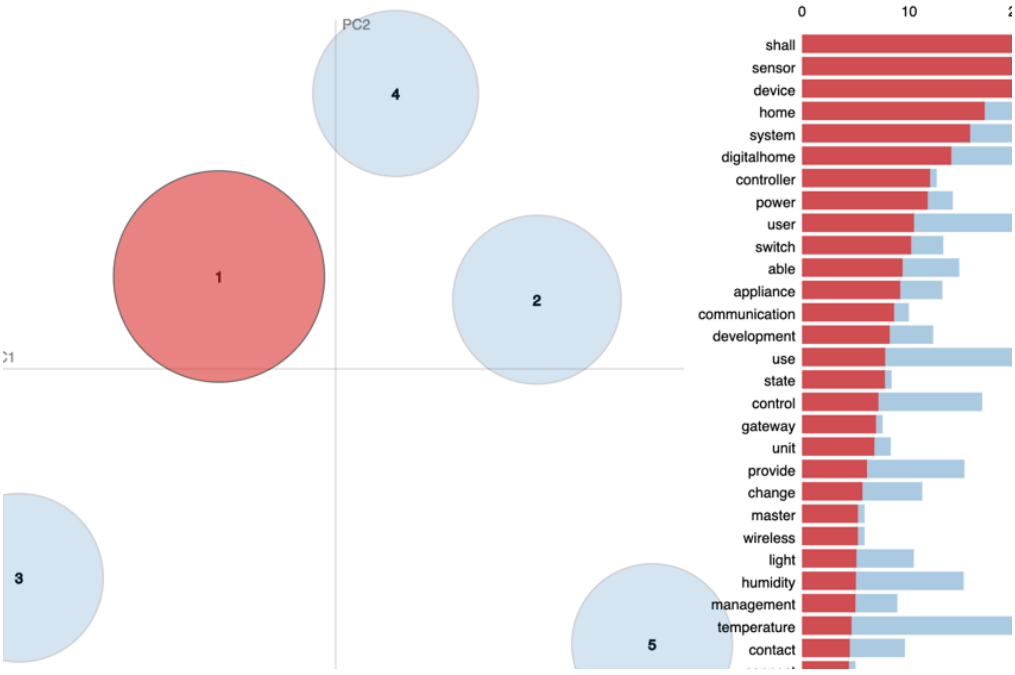


Figure 5.8 Distance et fréquence des termes par thème – E7

5.6.8 Exigence E8

Nous évaluons un document rédigé par les équipes de conception de la compagnie pour décrire les tâches d'automatisation d'un processus spécifique. Nous observons que la combinaison de la meilleure cohérence et de la rétroaction humaine a permis de consolider l'exploration automatique des thèmes. L'annexe H contient le détail d'exécution pour chaque nombre de thèmes, les mesures de cohérence et de perplexité ainsi que les termes dominants par thème.

Voici le résultat de l'application du modèle pour cinq thèmes.

```
Perplexité      : -5.67
Cohérence       : 0.48
Les termes dominants par thème
--------------------------------------------------
Thème 1 change customer service provision care status description product url method
Thème 2 process address new follow migration exist service port lnp task
Thème 3 payment user method store account quote use option service class
Thème 4 customer account email add crm credit check use request create
Thème 5 would current account department process user reseller sale input customer
```

Nous notons que les cinq thèmes fournissent plus de détails pour explorer cette exigence. Nous identifions le premier thème, qui regroupe les concepts. Le deuxième thème regroupe les processus relatifs à l'unité d'affaires LNP (Local Number Porting). Le deuxième thème représente les processus d'affaires pour la facturation. Le troisième thème est lié au service à la clientèle. Le cinquième thème regroupe les activités des ventes.

Voici des exemples de phrases extraites du document pour chaque thème.

Concept: *Provider entity, Reseller entity, Customer entity* Processus (unité d'affaires: LNP): *create task for the LNP department and ask them to start the porting process and attach the LOA and the Welcome Letter to it.* Processus (unité d'affaires: Facturation): *explore the various methods in the Payment Class and present payment options to the user.* Processus (unité d'affaires: Relation client) : *transfer the call/email to billing to process a payment.* Processus (unité d'affaires : Vente) : Administrator – An Iristel representative having access to reseller and customer accounts.

Nous identifions cinq thèmes qui combinent les processus et les concepts. La rétroaction a facilité la reconnaissance des processus par unité d'affaires. La figure 5.9 montre les termes dominants et les distances entre les thèmes.

Le modèle ne permet pas d'identifier les cas d'utilisation et les cas d'essai. Cependant, nous pouvons regrouper les processus d'affaires en un ou plusieurs cas d'utilisation. Nous pouvons également extraire les exigences et les cas d'essai à partir des phrases dans ce document qui contiennent les mots-clés, tels que « *shall, should, must, should, require* ».

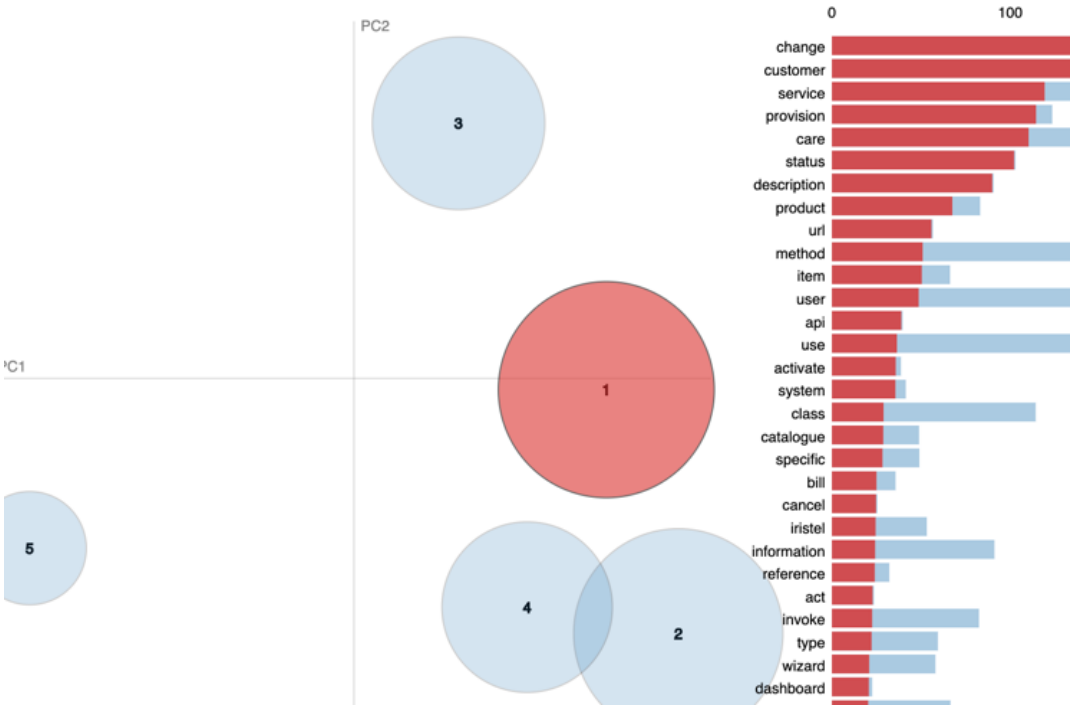


Figure 5.9 Distance et fréquence des termes par thème – E8

5.6.9 Exigence E9

Nous évaluons un document rédigé dans le cadre du développement d'un autre système chez Iristel. Nous observons que la combinaison de la meilleure cohérence et de la rétroaction humaine a permis de consolider l'exploration automatique des thèmes. L'annexe I contient le détail d'exécution pour chaque nombre de thèmes, les mesures de cohérence et de perplexité ainsi que les termes dominants par thème.

Voici le résultat de l'application du modèle pour cinq thèmes.

```
Perplexité      : -6.46
Cohérence       : 0.43
Les termes dominants par thème
--------------------------------------------------
Thème 1 lnp wholesale customer account call continue ponnumber invoice charge desire
Thème 2 number customer rout profile order function request return array account
Thème 3 wholesale number customer iristel time invoice call ordernumber may account
Thème 4 customer service rep product call sale iristel email may inbound
Thème 5 call customer list update ip function order reject pend iristel
```

Nous notons que les cinq thèmes fournissent plus de détails pour explorer cette exigence. Nous identifions le premier thème, qui regroupe les concepts. Le deuxième thème regroupe les exigences. Les trois thèmes restants regroupent les processus pour les trois unités d'affaires : ventes, facturation et service client. La figure 5.10 montre les termes dominants et les distances entre les thèmes.

Voici des exemples d'éléments extraits pour chaque thème.

Concept: *wholesale.EditPon('ponData')*.
Exigences: *the function will return an array with all the user's available routing*.
Processus (vente): *When creating the account, the Sales Rep needs to specify what is the maximum number of additional lines.*
Processus (facturation): *products will be selected from an available product catalogue.*
Processus (service client): *prompt the user to manage contact(s) information related to the account.*

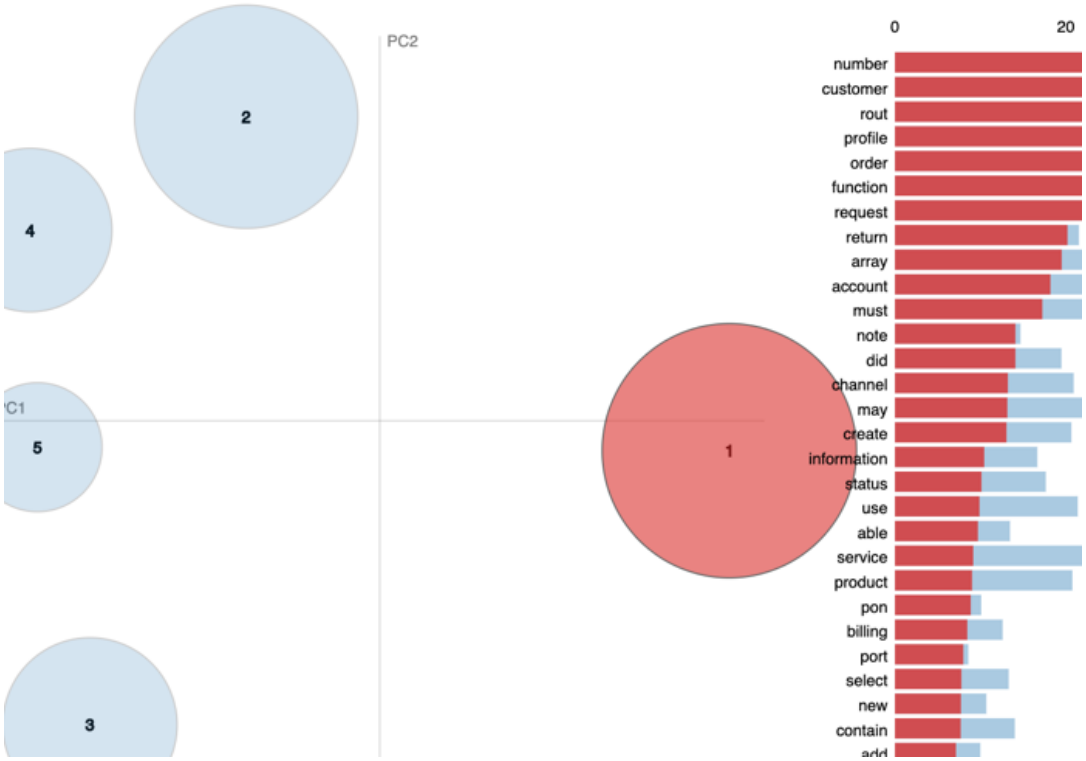


Figure 5.10 Distance et fréquence des termes par thème – E9

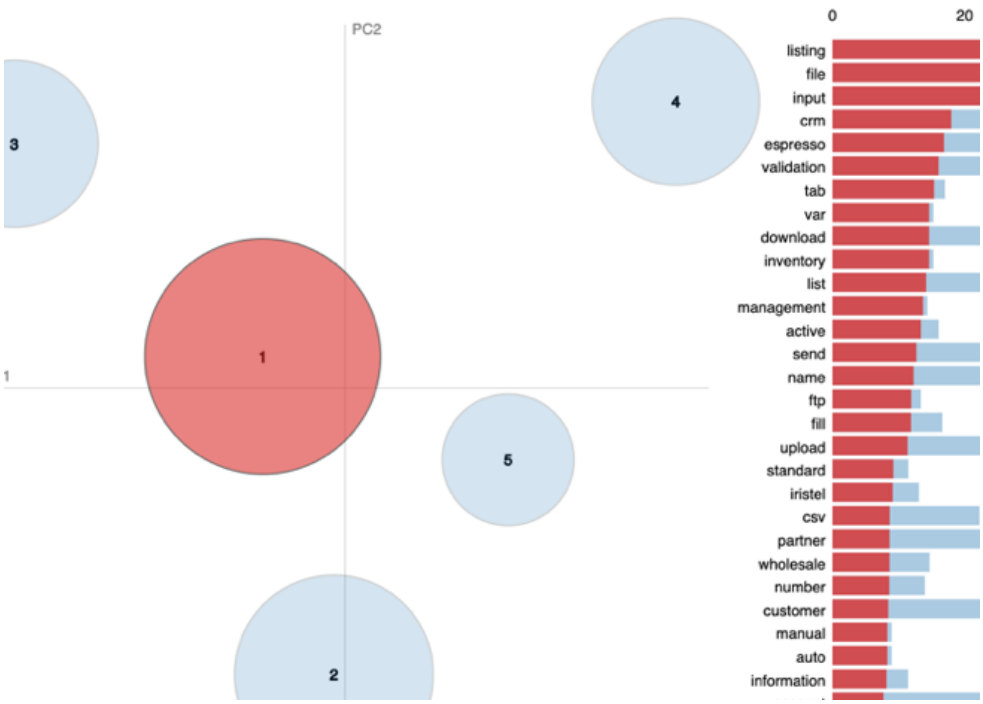
5.6.10 Exigence E10

Nous évaluons un document rédigé par les analystes chez Iristel. De la même manière que pour les deux premiers documents, nous observons que la combinaison de la meilleure cohérence et de la rétroaction humaine a permis de consolider l'exploration automatique des thèmes. L'annexe J contient le détail d'exécution pour chaque nombre de thèmes, les mesures de cohérence et de perplexité ainsi que les termes dominants par thème.

Voici le résultat de l'application du modèle pour cinq thèmes.

```
Perplexité        : -5.90
Cohérence         : 0.42
Les termes dominants par thème
--------------------------------------------------
Thème 1 name account partner accept list submission reject crm customer code
Thème 2 listing request create case mail task service fix would order
Thème 3 crm entry user new filter listing street change npa notification
Thème 4 exchange code npa file nxx template csv format download table
Thème 5 listing file input crm espresso validation tab var download inventory
```

Nous notons que les cinq thèmes fournissent plus de détails pour explorer cette exigence. Nous identifions le premier thème, qui regroupe les concepts. Le deuxième thème regroupe les exigences relatives à la prise des commandes pour les différents services. Le troisième thème regroupe les exigences relatives à un service de notification. Le quatrième thème regroupe les exigences relatives à la fonctionnalité de téléchargement. Le cinquième thème regroupe les exigences relatives au service impacté par le changement. La figure 5.11 montre les termes dominants et les distances entre les thèmes.



Voici des exemples de phrases extraites pour chaque thème.

Concept : *Account name, Account code, Adjustment date.*
Exigences (commandes) : *possibility to find about and request activation of services.*
Exigences (notifications) : *an event notification every time a new NPA and/or NXX is released.*
Exigences (téléchargement) : *New Bulk Upload to be created for scenarios when a new Exchange Code-NPA-NXX is to be added.*
Exigences (service spécifique): *reduce the amount of manual work of Customer Care Team.*

Figure 5.11 Distance et fréquence des termes par thème – E10

5.7 Établir les correspondances entre les artéfacts

À partir des éléments qui appartiennent à chaque thème, nous évaluons la similarité sémantique qui nous permet d'identifier les liens candidats entre ces éléments et, par conséquent, entre les thèmes. Les liens extraits définissent la traçabilité entre les éléments qui appartiennent à une exigence. Nous calculons la similarité sémantique (*softcosin*) entre les éléments en appliquant le modèle FastText[41].

Nous présentons les résultats par document en parcourant les éléments qui ont obtenu un degré d'appartenance supérieur ou égal à 80 % pour un thème. Nous adoptons une extraction des liens candidats. Un spécialiste révise ces liens pour les confirmer ou les infirmer. Un lien est considéré comme une correspondance si nous obtenons un score sémantique supérieur ou égal à 0,7.

Pour chaque exigence, nous calculons la précision, le rappel et la F-mesure et nous construisons la matrice de confusion pour les résultats. La matrice se caractérise par sa simplicité, elle permet de visualiser les résultats et d'analyser les performances du modèle.

---

[41] https://radimrehurek.com/gensim/models/fasttext.html

Nous distinguons les quatre valeurs suivantes :

- TP (*True Positives*) : la prédiction et la valeur réelle sont positives;
- TN (*True Negatives*) : la prédiction est négative, alors que la valeur réelle est positive;
- FP (*False Positives*) : la prédiction est positive, alors que la valeur réelle est négative;
- FN (*False Negatives*) : la prédiction est négative, alors que la valeur réelle est positive.

5.7.1 Exigence E1

La tâche de classification nous a permis de découvrir trois thèmes avec leurs éléments respectifs pour l'exemple de la librairie. Le calcul de similarité entre les artéfacts a permis de produire les liens candidats. La validation des liens candidats par un spécialiste a permis de produire les trois mesures.

Précision : 70 %
Rappel : 45 %
F-mesure : 55 %

Voici un exemple de comparaison pour ce calcul.

<u>Correspondance :</u>
Processus: *The operator generate a weekly report of transactions (holds placed, holds filled, and check-outs) per library and per system*.
Exigence : *Ability to report per library and per system*.
Similarité = 0.73
<u>Pas de correspondance :</u>
Processus : *The system displays the number of check-outs and placed holds per patron and per customer*.
Concept : *System Administrator*.
Similarité = 0

La figure 5.12 montre la matrice de confusion pour cet exemple. Le modèle permet d'identifier les correspondances avec une précision acceptable (70 %). Cependant, il présente une performance inférieure pour le rappel (45 %). Il permet également de mieux détecter les liens pertinents (5 liens pour 1 faux positif) mais présente des difficultés à détecter les liens non pertinents (11 liens pour 6 faux négatifs).

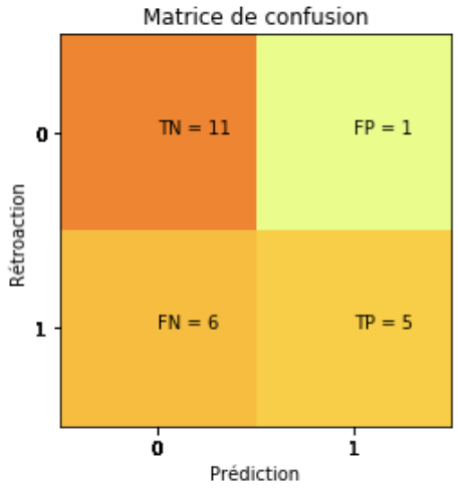


Figure 5.12 Matrice de confusion des correspondances pour E1

5.7.2 Exigence E2

La classification nous a permis de découvrir neuf thèmes, que nous avons regroupés à l'aide de la rétroaction humaine. Le calcul de similarité entre les artéfacts a permis de produire les liens candidats. La validation des liens candidats par un spécialiste a permis de produire les trois mesures.

Précision : 75 %
Rappel : 77 %
F-mesure : 76 %

Voici un exemple de comparaison pour ce calcul.

<u>Correspondance:</u>
Processus : *The user on the other side clicks on accept button*.
Processus : *The user on the other side clicks on deny button*.
Similarité = 0.97
<u>Pas de correspondance:</u>
Processus : *Double click on the Audio Button*.
Exigence: *A user can create a profile sharing his or her information*.
Similarité = 0

La figure 5.13 montre la matrice de confusion de cet exemple. Le modèle permet d'identifier les correspondances avec une précision acceptable (75 %). Il présente la même performance pour la précision et le rappel. Ce dernier obtient un score supérieur de 2 points de pourcentage à la précision. Le modèle permet de détecter les faux positifs et les faux négatifs avec presque le même taux de succès pour les liens pertinents et non pertinents.

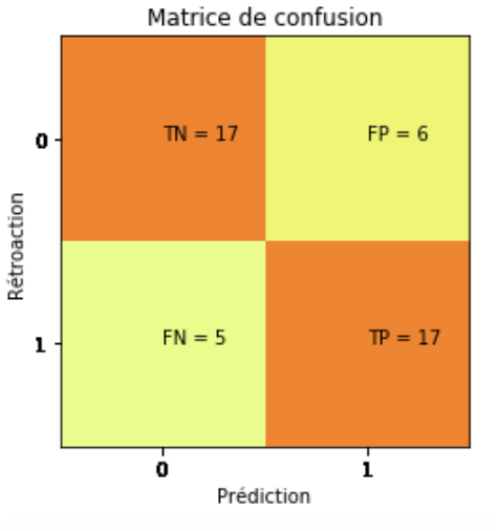


Figure 5.13 Matrice de confusion des correspondances pour E2

5.7.3 Exigence E3

La classification nous a permis de découvrir sept thèmes, que nous avons regroupés à l'aide de la rétroaction humaine. Le calcul de similarité entre les artéfacts a permis de produire les liens candidats. La validation des liens candidats par un spécialiste a permis de produire les trois mesures.

Précision : 65 %
Rappel : 86 %
F-mesure : 74 %

Voici un exemple de comparaison pour ce calcul.

<u>Correspondance:</u>
Processus: *the user can change the order and/or select the pages of the file/files.*
Processus: *number of pages to switch document: the user with this option can define the step size of the mix*.
Similarité = 0.86
<u>Pas de correspondance:</u>
Exigence: PDF Split and Merge is a multi-functional tool for manipulating pdf files.
Exigence: *Language supported Arabic, Asturian, Bosnian, Brazilian Portuguese…*
Similarité = 0

La figure 5.14 montre la matrice de confusion de cet exemple. Le modèle permet d'identifier les correspondances avec une précision moyenne (65 %). Il présente un rappel meilleur que la précision. L'application du modèle sur cette exigence obtient une performance acceptable par rapport aux autres documents analysés.

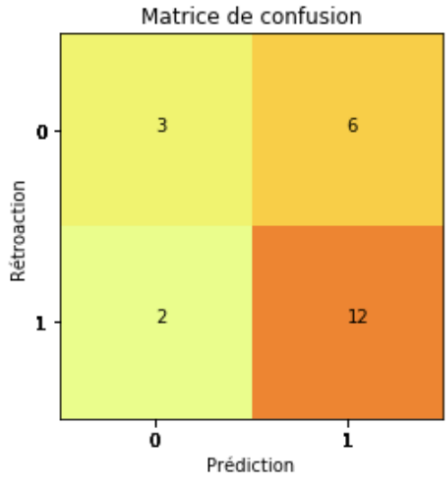


Figure 5.14 Matrice de confusion des correspondances pour E3

5.7.4 Exigence E4

La classification nous a permis de découvrir sept thèmes, que nous avons regroupés à l'aide de la rétroaction humaine. Le calcul de similarité entre les artéfacts a permis de produire les liens candidats. La validation des liens candidats par un spécialiste a permis de produire les trois mesures.

Précision : 86 % Rappel : 89 % F-mesure : 88 %

Voici un exemple de comparaison pour ce calcul.

<u>Correspondance:</u> Exigence: *All electronic reports from UK fishing vessels are required to be transmitted to the Electronic Recording and Reporting System (ERS).* Processus: *provides the data definitions that are required from UK fishing vessels in order to fulfil the regulatory requirements to report to the UK fisheries administrations.* Similarité = 0.79 <u>Pas de correspondance:</u> Exigence: *Applies to vessels using towed gear, long lines and fixed nets.* Exigence: *Article 4 refers to the reports required from fishing vessels.* Similarité = 0.17

La figure 5.15 montre la matrice de confusion de cet exemple. Le modèle permet d'identifier les correspondances avec une bonne précision (86 %). Il présente un rappel supérieur de 3 points de pourcentage à la précision. L'application du modèle sur cette exigence obtient une bonne performance par rapport aux autres documents analysés.

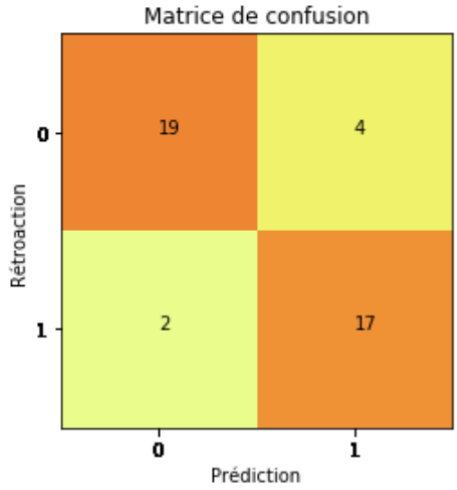


Figure 5.15 Matrice de confusion des correspondances pour E4

5.7.5 Exigence E5

La classification nous a permis de découvrir quatre thèmes, que nous avons regroupés à l'aide de la rétroaction humaine. Le calcul de similarité entre les artéfacts a permis de produire les liens candidats. La validation des liens candidats par un spécialiste a permis de produire les trois mesures.

Précision : 72 % Rappel : 78 % F-mesure : 75 %

Voici un exemple de comparaison pour ce calcul.

<u>Correspondance:</u> Processus: *If no editor is associated with either the file or the file type, then the default editor shall be used.* Processus: *the editor started by the HATS-GUI shall be the editor associated with the file in the application configuration.* Similarité = 0.86 <u>Pas de correspondance:</u> Exigence: *If no editors have been configured, then the HATS-GUI shall inform the user that an editor must be configured before starting an editor.* Exigence: *To execute a transformation language program, the parsed transformation language program and the user- defined library file are sent to HATS-SML.* Similarité = 0.32

La figure 5.16 montre la matrice de confusion de cet exemple. Le modèle permet d'identifier les correspondances avec une précision acceptable (72 %). Il présente un rappel supérieur de 6 points de pourcentage à la précision. L'application du modèle sur cette exigence obtient une performance moyenne par rapport aux autres documents analysés.

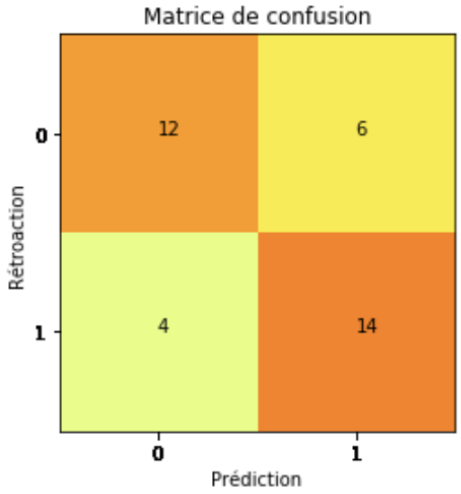


Figure 5.16 Matrice de confusion des correspondances pour E5

5.7.6 Exigence E6

La classification nous a permis de découvrir quatre thèmes, que nous avons regroupés à l'aide de la rétroaction humaine. Le calcul de similarité entre les artéfacts a permis de produire les liens candidats. La validation des liens candidats par un spécialiste a permis de produire les trois mesures.

Précision : 79 % Rappel : 71 % F-mesure : 75 %

Voici un exemple de comparaison pour ce calcul.

<u>Correspondance:</u> Exigence: *Outcomes from CEN BII are syntax neutral, which means that different syntaxes can be bound to the semantic requirements for the documents to be exchanged meaningfully despite their syntax*. Processus: *This means that the VCD can be developed from a collection of paper documents to a file that can be processed according to agreed CEN BII profiles by the use of the two standards.* Similarité = 0.74 <u>Pas de correspondance:</u> Exigence: *Each layer has a greater level of specialization and tends to be more complex than the layers below it*. Exigence: *The purpose of an e-business transport infrastructure is to transport business documents between business partners.* Similarité = 0.19

La figure 5.17 montre la matrice de confusion de cet exemple. Le modèle permet d'identifier les correspondances avec une précision acceptable (79 %). Il présente une précision meilleure que le rappel. L'application du modèle sur cette exigence obtient une performance moyenne par rapport aux autres documents analysés.

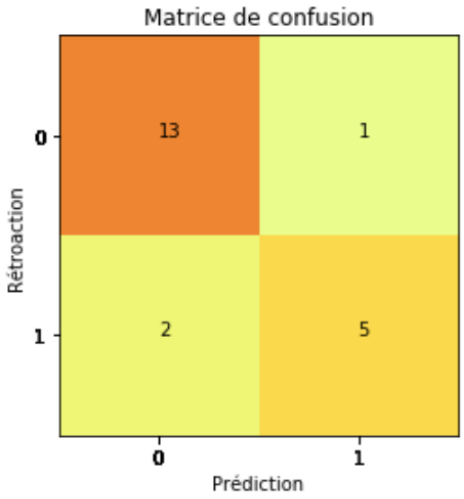


Figure 5.17 Matrice de confusion des correspondances pour E6

5.7.7 Exigence E7

La classification nous a permis de découvrir quatre thèmes, que nous avons regroupés à l'aide de la rétroaction humaine. Le calcul de similarité entre les artéfacts a permis de produire les liens candidats. La validation des liens candidats par un spécialiste a permis de produire les trois mesures.

Précision : 78 % Rappel : 79 % F-mesure : 78 %

Voici un exemple de comparaison pour ce calcul.

<u>Correspondance:</u> Exigence: *DigitalHome Planner shall provide a user with the capability to direct the system to set various preset home parameters*. Exigence: *the DigitalHome shall be equipped with various environmental controllers.* Similarité = 0.41 <u>Pas de correspondance:</u> Exigence*: there will be no actual physical home and all sensors and controllers will be simulated.* *Exigence: it is made up of a list of the principal features of the system*. Similarité = 0.12

La figure 5.18 montre la matrice de confusion de cet exemple. Le modèle permet d'identifier les correspondances avec une précision acceptable (78 %). Il présente un rappel presque similaire à la précision. L'application du modèle sur cette exigence obtient une performance moyenne par rapport aux autres documents analysés.

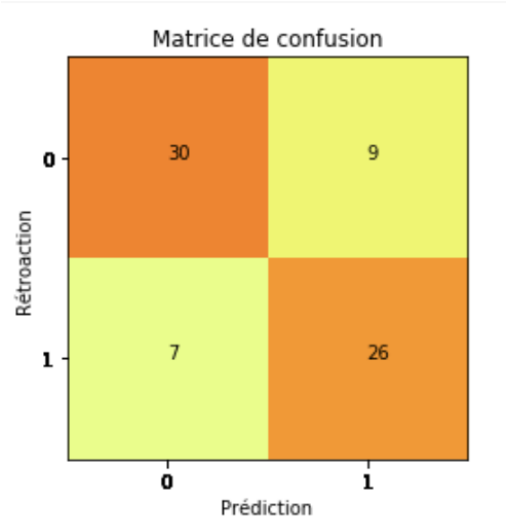


Figure 5.18 Matrice de confusion des correspondances pour E7

5.7.8 Exigence E8

La classification nous a permis de découvrir les différents thèmes qui composent ce document. Le calcul de similarité entre les éléments des différents thèmes a fourni la liste des liens candidats. La validation des liens candidats par un spécialiste a permis de produire les trois mesures.

Précision : 84 % Rappel : 93 % F-mesure : 88 %

Voici un exemple de comparaison pour ce calcul.

Correspondance : Concept: *the information included in this document: customer information, point of contact, secret question*. Processus: *Set the credit limit account in CRM based on the result of the credit check* Similarité = 0.7 Pas de correspondance: Processus: *Go back on the lead and make sure the address provided is complete* Exigence: *International Number requests can reach Customer Care directly form the customer by phone or email* Similarité = 0.33

La figure 5.19 montre la matrice de confusion de cet exemple. Le modèle permet d'identifier les correspondances avec une bonne précision (84 %) et un meilleur rappel (93 %). Le modèle a produit de meilleurs résultats pour ce document que pour les deux premiers exemples.

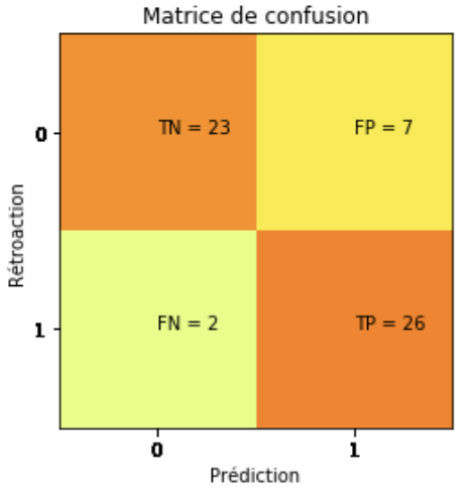


Figure 5.19 Matrice de confusion des correspondances pour E8

5.7.9 Exigence E9

La classification a facilité la découverte des différents thèmes qui composent ce document. Le calcul de similarité entre les éléments qui appartiennent aux thèmes a fourni la liste des liens candidats. La validation des liens candidats par un spécialiste a permis de produire les trois mesures.

Précision : 74 % Rappel : 75 % F-mesure : 74,5 %

Voici un exemple de comparaison pour ce calcul.

<u>Correspondance :</u> Processus: *The customer fills in the subscription form* Exigence: *Prepopulate a form with the lead's information* Similarité = 0.77 <u>Pas de correspondance:</u> Processus: *The customer will be prompted for additional information needed to perform a credit check* Exigence: *Create task for the LNP department and ask them to start the porting process* Similarité = 0.3

La figure 5.20 montre la matrice de confusion de cet exemple. Le modèle permet d'identifier les correspondances avec une précision presque similaire à celle du deuxième document (74 %) et un rappel supérieur de 1 point de pourcentage à la précision. L'application du modèle sur cette exigence obtient une performance acceptable par rapport aux documents analysés.

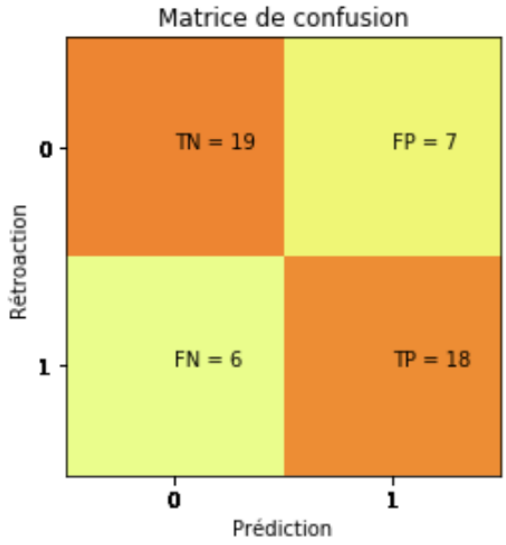


Figure 5.20 Matrice de confusion des correspondances pour E9

5.7.10 Exigence E10

La classification nous a permis de découvrir les différents thèmes qui composent ce document. Le calcul de similarité entre les éléments qui appartiennent aux thèmes a fourni la liste des liens candidats. La validation des liens candidats par un spécialiste a permis de produire les trois mesures.

Précision : 82 % Rappel : 90 % F-mesure : 86 %

Voici un exemple de comparaison pour ce calcul.

<u>Correspondance:</u> Exigence: *New report will include both the Accepted and Rejected entries* Processus: *Agent send Wholesaler Report to customer with both Accepted and Rejected entries* Similarité = 0.74 <u>Pas de correspondance:</u> Exigence: *CRM validation customer report will include in CSV File all Phone Numbers* Processus: *Agent change name for existing Wholesaler Report issued at Listings Partners validation* Similarité = 0.33

La figure 5.21 montre la matrice de confusion de cet exemple. Le modèle permet d'identifier les correspondances avec une bonne précision (82 %) et un meilleur rappel (90 %). L'application du modèle sur ce document obtient une meilleure performance par rapport aux deux premiers documents. Le modèle obtient une meilleure performance pour détecter les liens pertinents et non pertinents.

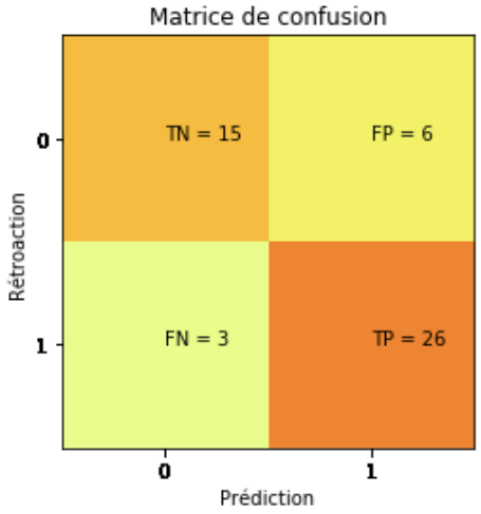


Figure 5.21 Matrice de confusion des correspondances pour E10

5.7.11 Exigence E11

Dans cet exemple, nous avons comparé les éléments qui appartiennent aux exigences E8, E9 et E10. Nous regroupons ces trois documents dans un document E11. Le calcul de similarité entre les éléments des différents thèmes a fourni la liste des liens candidats. La validation des liens candidats par un spécialiste a permis de produire les trois mesures.

Précision : 86 %
Rappel : 91 %
F-mesure : 89 %

Voici un exemple de comparaison pour ce calcul.

Correspondance:

Processus: *Create task for the LNP department and ask them to start the porting process and attach the LOA to it*

Exigence: *It is required to send the file to Partners for Phone Directory Assistance*

Similarité = 0.76

Pas de correspondance:

Exigence: *Live Chat prompts will be displayed on all customer-facing web interfaces*.

Exigence: *Resulting Active 411 Listings Report will only display those specific records*

Similarité = 0.5

La figure 5.22 montre la matrice de confusion de cet exemple. Le modèle permet d'identifier les correspondances avec une bonne précision (86 %) et un meilleur rappel (91 %). L'application du modèle pour ce document obtient une meilleure performance que les autres exigences. Le modèle obtient une meilleure performance pour détecter les liens pertinents et non pertinents.

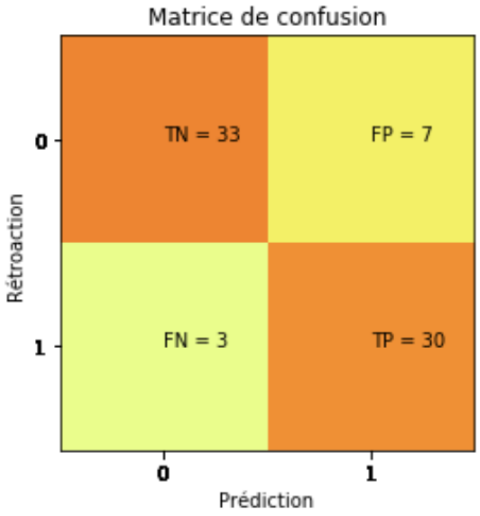


Figure 5.22 Matrice de confusion des correspondances entre E8, E9 et E10

5.8 Conclusion

Nous avons évalué notre approche en l'appliquant sur dix documents issus de huit sources différentes. Nous nous sommes principalement intéressé aux tâches de classification et d'établissement des correspondances. Le tableau 5.3 résume les résultats obtenus pour chaque tâche et pour chaque document. Afin d'établir les correspondances entre les trois dernières exigences, nous avons appliqué l'approche pour établir les correspondances sur une sixième exigence qui regroupe ces trois documents.

Lors de la phase de classification, nous avons exploré les catégories des artéfacts à partir des exigences. L'approche a permis d'extraire le niveau conceptuel pour huit documents évalués. Nous avons également identifié les processus dans six documents évalués. Le septième document contient des informations relatives aux exigences uniquement, nous n'avons pas pu identifier des flux fonctionnels dans le texte.
Pour le deuxième document, nous avons extrait toutes les catégories que nous ciblons dans notre travail. En effet, nous avons distingué, en plus des trois catégories, les cas d'essai et les cas d'utilisation. En explorant le document manuellement, nous avons constaté qu'il contient des paragraphes rédigés avec une structure précise relative aux différentes catégories.

Tableau 5.3 Tableau récapitulatif des résultats des évaluations

	Classification					Correspondances		
	Exigence	Cas d'utilisation	Processus	Concepts	Cas d'essai	Précision	Rappel	F-mesure
E1	x		x	x		70 %	45 %	55 %
E2	x	x	x	x	x	76 %	77 %	76 %
E3	x		x	x	x	65 %	86 %	74 %
E4	x			x		86 %	89 %	88 %
E5	x	x	x			72 %	78 %	75 %
E6	x	x		x		79 %	71 %	75 %
E7	x					78 %	79 %	78 %
E8			x	x		84 %	92 %	88 %
E9	x		x	x		74 %	75 %	74 %
E10	x			x		82 %	90 %	86 %
E11						86 %	91 %	89 %

Nous avons identifié les catégories non détectées par l'allocation latente de Dirichlet en appliquant les méthodes classiques d'analyse grammaticale des phrases extraites. Nous pouvons convertir les artéfacts des processus en cas d'essai. Un cas d'utilisation peut regrouper un ou plusieurs processus d'affaires.

Notre approche s'exécute en deux étapes principales. La première étape consiste à explorer les thèmes traités par l'exigence, la deuxième étape établit les correspondances entre les différents artéfacts. L'approche permet d'extraire des informations pertinentes et ne requiert aucune connaissance préalable des domaines.

Les résultats des expérimentations ont montré que la qualité de rédaction des exigences compte parmi les critères de succès de l'approche. En effet, nous obtenons de meilleurs résultats pour les documents rédigés en appliquant les normes reconnues par les industries.

Nous appuyons l'évaluation de notre approche par une rétroaction humaine qui aide à améliorer les résultats et à ajuster les paramètres du modèle tout au long des étapes de sa réalisation. L'application d'une rétroaction humaine a permis de consolider les résultats des approches. Elle aide à apporter les adaptations pour regrouper ou changer le nombre de thèmes générés par la modélisation des thèmes. Le calcul de cohérence supporté par la rétroaction humaine a permis d'obtenir de meilleurs résultats pour découper les documents en plusieurs thèmes cohérents et reconnaissables par les acteurs du système. La cohérence a permis de bien choisir le nombre de thèmes pour la plupart des exigences.

Nos expérimentations ont permis d'établir les correspondances entre les éléments à l'aide de la mesure cosinus soft. L'application d'un seuil préétabli a permis de distinguer les liens pertinents et les liens non pertinents. La rétroaction humaine a permis de valider les liens candidats ainsi que les liens inexistants entre les éléments. La qualité des résultats de la similarité dépend principalement de la qualité des

documents rédigés et du niveau d'abstraction appliqué au moment de leur rédaction. La rétroaction humaine a joué un rôle important dans l'évaluation des liens candidats.

CHAPITRE VI

CONTRIBUTIONS ET PERSPECTIVES

6.1 Contributions

Dans ce chapitre, nous décrivons notre contribution en explorant les différents aspects traités par l'approche proposée. Nous présentons également les pistes de recherche potentielles en nous appuyant sur les possibilités des modèles et sur les articles publiés par la communauté de la traçabilité des exigences.

Le développement des techniques et algorithmes de traitement et de compréhension du texte a facilité les recherches dans le domaine de l'ingénierie des exigences. Plus précisément, Borg, Runeson, & Ardö (2014) ont développé des modèles qui ont contribué à des améliorations significatives des techniques de TAL afin de résoudre les problématiques liées à la traçabilité des exigences.

Depuis plusieurs décennies, les chercheurs se sont intéressés à la traçabilité des systèmes en analysant les exigences textuelles. Ils ont mis en évidence différents aspects de recherche, tels que les modèles mathématiques et les statistiques appliquées.

Nous présentons ces contributions en montrant les relations avec les aspects suivants :

- les modèles appliqués : modèles algébriques, statistiques;
- le type de correspondance : lien entre les exigences, liens entre les cas d'essai et les exigences;
- la technique d'amélioration et les évaluations appliquées : la rétroaction, les thésaurus.

Borg, Runeson, & Ardö (2014) ont mené une étude empirique qui a facilité l'exploration des travaux pertinents du domaine. Leur étude consiste à explorer ce que les chercheurs ont réalisé dans le cadre du rétablissement de la traçabilité des exigences (*IR-based trace recovery*). Ils ont dessiné une carte relative aux publications dans lesquelles ils ont proposé des approches qui permettent l'extraction des correspondances et des traces entre les artéfacts.

Nous avons ciblé trois grandes questions de recherche. Ces questions nous ont permis de cerner l'apport de notre travail. Notre contribution s'articule autour des questions de recherche suivantes (Borg, Runeson, & Ardö, 2014) :

- explorer les modèles appliqués et les stratégies d'amélioration fréquemment adoptés pour rétablir les traces entre les artéfacts logiciels;
- recenser les types d'artéfacts les plus utilisés dans ce domaine;
- étudier les types de preuves les plus adoptés par les chercheurs.

6.1.1 Modèle et similarité

Depuis 2008, des chercheurs ont appliqué plusieurs types de modèles pour rétablir les correspondances entre les artéfacts (Borg, Runeson, & Ardö, 2014). Ils ont utilisé les modèles algébriques dans vingt-six publications. D'autres chercheurs ont adopté l'indexation sémantique latente dans une vingtaine de recherches. Les chercheurs ont également appliqué les modèles probabilistes dans dix-neuf publications.

Dans le sous-ensemble des modèles probabilistes, les chercheurs ont expérimenté le rétablissement de la traçabilité à l'aide de modèles statistiques. Ces modèles ont facilité la classification des documents au moyen de la probabilité et des fréquences de mots. A cet effet, nous nous sommes intéressé aux méthodes de modélisation des thèmes. C'est une technique qui permet de modéliser les documents textuels composés d'un mélange de thèmes. Dans le cadre des travaux qui visent la traçabilité des exigences, citons les modèles suivants :

- l'indexation sémantique latente probabiliste (Hofmann, 2001);
- l'allocation latente de Dirichlet (Blei, Ng, & Jordan, 2003);
- la modélisation des thèmes corrélés (Blei & Lafferty, 2007);
- la modélisation des thèmes relationnels (Chang & Blei, 2010).

Borg, Runeson, & Ardö (2014) ont recensé cinq publications dans lesquelles les auteurs ont appliqué la modélisation des thèmes. Dans ces cinq publications, les chercheurs se sont appuyés sur une connaissance préalable du domaine et du contenu des exigences.

Dans notre travail, la tâche initiale de classification ne requiert pas une connaissance préalable du domaine. Nous avons exploré les documents afin de découvrir les thèmes qui les composent. Nous avons identifié les thèmes, qui varient selon le contexte et l'objectif de rédaction des documents. Cette exploration a facilité la validation du contenu avant d'établir les correspondances entre les artéfacts. En effet, l'application du modèle ALD nous a permis de retenir uniquement les documents, les paragraphes et les phrases pertinents. Nous avons également combiné le modèle avec une évaluation qui calcule automatiquement des mesures, telles que la cohérence des thèmes extraits. Sans aucune connaissance préalable, nous avons identifié les thèmes relatifs aux exigences, aux cas d'utilisation, aux processus d'affaires, aux entités conceptuelles et aux cas d'essai.

6.1.2 Type de correspondance et type d'artéfacts

Borg, Runeson, & Ardö (2014) ont confirmé que des chercheurs ont ciblé plusieurs types d'artéfacts qui appartiennent à un système d'information. Citons des artéfacts tels que les exigences du marché, les exigences système, les exigences fonctionnelles, les cas d'utilisation et les spécifications conceptuelles. Notre approche permet d'identifier les thèmes qui composent un document; chaque thème représente un type d'artéfact déterminé. L'établissement des liens entre ces documents s'effectue à l'aide d'un calcul de la similarité sémantique. Nous avons spécifié la valeur 0,7 comme seuil de correspondances : nous considérons qu'une correspondance existe si la valeur de similarité est supérieure ou égale à 0,7. Borg, Runeson, & Ardö (2014) ont affirmé que la plupart des chercheurs ont ciblé les relations entre les exigences uniquement avec différents niveaux d'abstraction : trente-sept travaux ont mis l'accent sur ce type de lien. Dans trente-deux recherches, les auteurs se sont intéressés aux correspondances entre le code source et les exigences. Les relations entre le code source et les cas d'essai ont fait l'objet de dix travaux. Borg, Runeson, & Ardö (2014) ont recensé une seule publication dans laquelle les auteurs ont traité la relation entre le code source et les

anomalies; et une étude sur la relation entre les cas d'essai. Dans la plupart des autres publications, les auteurs n'ont pas précisé clairement le type d'artéfacts étudié, mais se sont contentés de les nommer d'une manière générique.

À cet égard, notre approche permet d'établir les correspondances entre plusieurs types de documents. La classification initiale facilite l'exploration du contenu des documents et l'extraction des artéfacts. Nous avons visé les exigences, les cas d'utilisation, les processus d'affaires, les entités conceptuelles et les cas d'essai. Notre approche s'applique d'une manière générique et cible plusieurs types d'éléments en parallèle. En effet, nous avons exploré les exigences sans aucune connaissance préalable du domaine et nous avons établi les correspondances sur le résultat de cette classification. Nous pouvons rétablir les correspondances entre plusieurs types d'artéfacts, exigences, cas d'utilisation, processus, entités conceptuelles et cas d'essai. Le calcul de correspondance s'effectue sans prendre en considération le type d'artéfacts.

6.1.3 Technique d'évaluation et d'amélioration

Borg, Runeson, & Ardö (2014) ont exploré cent-trente-deux publications pour étudier les évaluations appliquées par les chercheurs. Dans la plupart de ces évaluations, les modèles ont été mis en application à l'aide de documents rédigés dans un cadre universitaire. D'autres chercheurs se sont appuyés sur des exigences écrites par les analystes de la NASA ainsi que sur des documents produits dans des projets de code source libre (*open source*). L'exploration a démontré que les chercheurs ont largement utilisé deux documents d'exigences principales : le projet de développement logiciel de la NASA et un projet universitaire pour un système clinique. Nous avons ciblé d'autres sources de documents pour appliquer l'approche dans le cadre de domaines distincts. Si les autres publications ont évalué leur approche à l'aide d'une ou deux sources à la fois, nous avons adopté plusieurs types de documents produits dans un cadre réel.

Nous avons évalué notre approche à l'aide de dix exigences produites dans le cadre de huit contextes réels distincts. L'évaluation dans un cadre réel a permis de mesurer le degré d'application et de généralisation de l'approche. Les modèles conçus dans le cadre de la recherche pourront être appliqués dans des cas d'utilisation réels, ce qui constitue une contribution à l'industrie du logiciel.

À ce sujet, des chercheurs ont appliqué plusieurs types de stratégies d'amélioration. La plupart des techniques ciblent la précision et le rappel du résultat produits par les modèles. La rétroaction humaine est la stratégie la plus utilisée par la communauté de la traçabilité des exigences (Borg, Runeson, & Ardö, 2014). Ces rétroactions permettent d'améliorer les modèles, mais aussi de les juger et d'en mesurer les performances. Nous avons utilisé la rétroaction humaine comme moyen efficace pour apporter la connaissance du métier relatif au domaine d'application. En effet, c'est un facteur d'amélioration qui a permis d'appliquer notre approche dans plusieurs domaines d'expertise et d'en obtenir les mêmes performances. Cette rétroaction a fourni aux spécialistes du domaine la possibilité de juger partiellement les résultats de l'approche dans un environnement réel.

6.2 Dimension cognitive et informatique

Établir les correspondances entre les artéfacts à partir des exigences est un domaine de recherche qui relève de l'extraction des connaissances ou de l'information (Borg, Runeson, & Ardö, 2014). L'écosystème du domaine se compose d'un ensemble d'éléments de recherche. Les chercheurs appliquent des modèles mathématiques et statistiques sur des exigences textuelles; les acteurs et spécialistes humains apportent leur expertise pour juger des résultats que les modèles automatiques produisent dans un environnement expérimental ou réel. Citons les trois facteurs de recherche suivants :

- ensemble de données : cet élément se présente sous forme de documents textuels que les analystes et concepteurs rédigent pour concevoir le système;
- acteurs : ce sont les acteurs humains qui apportent un jugement sur les résultats;
- environnement : c'est l'environnement d'exécution, qui peut être expérimental ou réel.

Ingwersen & Järvelin (2005) ont présenté un cadre cognitif intégré d'évaluation pour l'extraction de l'information. Ils ont distingué quatre niveaux de contexte :

- niveau 1 : le premier niveau se présente comme une cave d'évaluation de l'extraction. C'est un contexte orienté vers la technologie avec des caractéristiques et contextes simplistes. Les chercheurs calculent la performance à l'aide de la précision et du rappel sur la base des résultats obtenus;
- niveau 2 : c'est un deuxième niveau qui constitue un premier pas vers l'application réelle du modèle. Les chercheurs mettent un pas à la sortie de la cave. Ils ciblent le contexte en s'appuyant sur la manière dont les acteurs humains identifient les informations pertinentes à partir des résultats de la machine;
- niveau 3 : c'est un troisième niveau avec un meilleur degré de réalisme relativement aux deux premiers niveaux. Les acteurs humains évaluent

l'exécution des tâches dans un environnement d'expérimentations. L'objectif consiste à évaluer l'effet causal de l'approche ou du modèle qui exécute une tâche spécifique, telle que la classification ou l'identification des correspondances entre les artéfacts. C'est un niveau qui s'applique à un contexte d'études quantitatives et qualitatives;

- niveau 4 : c'est un niveau plus réel que le troisième. Les chercheurs évaluent les modèles dans un contexte socio-organisationnel réel. Ce niveau s'applique à une étude qualitative qui domine les évaluations.

Notre approche s'articule autour des niveaux trois et quatre : elle s'appuie sur le facteur humain dans un environnement expérimental ou réel. Nous avons mené notre évaluation dans un environnement expérimental avec des données réelles. Notre modèle compte deux tâches principales : la classification et le calcul de similarité. Notre évaluation consiste à comparer les résultats obtenus par le modèle avec les résultats obtenus par les spécialistes du domaine.

Nous avons utilisé des données réelles conçues dans un environnement réel et nous avons simulé l'apport humain dans la plupart des évaluations. Pour des raisons de confidentialité ou de non-disponibilité des acteurs réels, nous avons traité deux types d'exigences :

- Les documents pour lesquels nous n'avions pas accès aux acteurs réels : nous avons demandé la rétroaction de collègues spécialistes expérimentés dans l'analyse et la rédaction d'exigences. Ils ont révisé les documents et ils ont compris le contexte et le contenu. Ils ont pu agir en tant que spécialistes du domaine pour une rétroaction simulée;
- Les documents pour lesquels nous avions accès aux acteurs réels : nous avons utilisé les exigences rédigées pour un partenaire industriel qui nous a facilité l'accès aux ressources et aux experts. Nous avons ainsi consulté les vrais acteurs et spécialistes qui peuvent directement agir et effectuer une rétroaction humaine réelle dans leur propre domaine d'expertise.

6.3 Perspectives et futurs travaux

Nous présentons ici les perspectives potentielles pour les futurs travaux en analysant différents aspects et limitations de notre approche. Notre approche presente des limitations que nous pouvons resumer comme suit :

- Dépendance d'une rétroaction humaine qui requiert une expertise
- Qualité de la rédaction
- Jugement binaire des correspondances

Nous proposons l'application des thésaurus pour améliorer le modèle à l'aide des connaissances du domaine. Nous décrivons également l'application du modèle dans trois autres objectifs : nous proposons son utilisation pour vérifier la qualité des exigences, découvrir des types de liens entre les artéfacts ainsi que pour rétablir des séquences relatives aux processus d'affaires.

6.3.1 Thésaurus

Nous avons conçu notre approche sur la base du facteur humain. La rétroaction consiste à recourir aux connaissances des analystes et des experts pour évaluer les résultats. Cependant, la rétroaction humaine peut présenter des résultats différents ou incohérents selon le degré de connaissances des experts. Une piste de recherche consiste à appliquer un vocabulaire commun validé par les acteurs du domaine. Ce vocabulaire permettra de valider les résultats par la suite afin d'évaluer l'apport du thésaurus sur le traitement automatique. Le thésaurus est un moyen efficace pour remédier à l'ambiguïté et à la diversité des connaissances des acteurs humains. Cet outil sert à contrôler le vocabulaire que les acteurs et organisations définissent pour un domaine spécifique.
Le vocabulaire commun sert à préciser explicitement les relations entre les concepts (Aitchison, Bawden, & Gilchrist, 2003). Borg, Runeson, & Ardö (2014) ont exploré son utilisation pour la traçabilité entre les exigences pour fournir un

système de concepts et de termes acceptables par les experts du domaine. Le vocabulaire limite les termes que le modèle peut utiliser pour l'indexation et la recherche, les relations sémantiques et les relations entre les termes, telles que la synonymie et l'homonymie. L'utilisation d'un thésaurus permettra d'améliorer notre modèle en minimisant les erreurs d'ambiguïté, de cohérence et de compréhension des termes. En explorant l'ajout du vocabulaire à notre approche, le thésaurus jouera le rôle de regroupement des connaissances du domaine. Nous pourrons ainsi minimiser le rôle des acteurs humains. Les concepts du thésaurus permettront d'améliorer les résultats de la classification et de la précision de similarité pour établir les correspondances.

6.3.2 Qualité des documents

Arora, Sabetzadeh, Briand, & Zimmer (2015) ont affirmé que les modèles de documents représentent un outil efficace pour améliorer la précision des exigences rédigées dans un langage naturel. Les modèles permettent d'éviter les ambiguïtés qui peuvent surgir dans la rédaction des documents textuels. L'utilisation des modèles pour la rédaction des exigences fait partie des bonnes pratiques de l'industrie. Cependant, les analystes font face à un grand défi de validation pour approuver les documents. Selon Arora, Sabetzadeh, Briand, & Zimmer (2015), des chercheurs se sont intéressés à la validation des documents par rapport aux modèles prédéfinis dans les normes et les meilleures pratiques de l'industrie. La vérification manuelle de la conformité des exigences présente des défis qui grandissent quand la tâche de validation doit être répétée plusieurs fois pour des exigences fréquemment mises à jour.

Arora, Sabetzadeh, Briand, & Zimmer (2015) ont présenté une approche automatique qui utilise le découpage du texte en comparant les mots avec des modèles prédéfinis. Afin de vérifier la conformité des exigences avant d'établir les correspondances, nous proposons d'explorer la possibilité d'exécuter la classification ALD pour un lot qui contient les documents modèles et les documents rédigés. Nous évaluerons le résultat du regroupement par thème, les documents valides serons regroupés avec leurs modèles respectifs sous le même sujet. De plus, nous pourrons également explorer la validation des documents, avant d'établir les correspondances, en calculant les similarités sur un échantillon des documents regroupés. Les textes similaires permettront de déterminer le degré de ressemblance et de validité entre un document modèle et un document rédigé.

6.3.3 Type de liens

Marcus & Maletic (2003) ont affirmé que le processus de calcul de similarité entre deux documents doit prendre en considération un seuil minimal pour juger de l'existence de relations entre les documents. Ils ont considéré que le seuil heuristique est supérieur ou égal à 0,7. Cette valeur permet de déterminer l'existence de liens entre les documents, les phrases et les mots. Ainsi, nous proposons un autre aspect à analyser en explorant d'autres valeurs calculées à l'aide de la similarité sémantique. Si nous pouvons conclure qu'une correspondance existe entre deux artéfacts, une autre possibilité consisterait à découvrir la signification de tous les autres types de liens non couverts par notre approche.

L'objectif consiste à atteindre un jugement plus précis qu'une simple correspondance révélée par une valeur binaire. Le calcul de similarité supporte toutes les valeurs entre 0 et 1. L'idée proposée consiste à associer des explications à ces différentes valeurs. Une telle approche peut être appliquée pour évaluer différents types de liens, notamment des liens directs ou indirects, des liens forts, moyens ou faibles et des liens inexistants. Nous suggérons d'analyser davantage les valeurs calculées pour la similarité sémantique entre les artéfacts. Les spécialistes joueront un rôle important dans la réalisation de cette tâche en analysant et en expliquant les résultats. Citons des exemples de valeurs avec leur signification potentielle :

- une valeur 1 indique un lien très fort ou deux artéfacts identiques;
- une valeur entre 0,7 et 0,9 indique un lien fort;
- une valeur entre 0,5 et 0,7 indique un lien moyen ou une relation indirecte;
- une valeur entre 0,1 et 0,5 indique un lien faible;
- une valeur 0 indique l'absence de lien.

6.3.4 Identifier et rétablir les séquences d'un processus

Kowsari & al. (2017) ont visé la classification hiérarchique des documents, traitée comme un problème de classes multiples, en utilisant l'apprentissage profond hiérarchique HDLTex (*Hierarchical Deep Learning for Text Classification*). Cette méthode emploie un ensemble de modèles pour fournir une compréhension spécialisée pour chaque niveau de la hiérarchie.

Pan & al. (2018) ont présenté les modèles de détection de séquences (*seq2seq*) à partir du texte. Ils ont affirmé que ces modèles permettent d'encoder en entrée une séquence à l'aide d'un réseau de neurones. Chaque séquence se présente comme un vecteur de mots en appliquant un réseau de neurones récurrent. Le deuxième réseau de neurones permet de décoder le résultat comme une séquence étape par étape conditionnée par l'encodage initial. Les chercheurs ont adopté ce genre de modèle pour détecter les séquences dans le texte. L'approche consiste à comparer chaque étiquette générée avec une autre étiquette de la séquence initiale. Nous proposons d'effectuer des recherches sur le résultat de la classification en prenant en considération tous les artéfacts qui appartiennent au thème relatif aux processus d'affaires. L'idée consiste à appliquer le modèle (*seq2seq)* pour identifier les étapes d'un processus à partir du texte extrait des exigences.

Notre approche a permis d'identifier le texte qui regroupe les processus; nous proposons d'utiliser le résultat en appliquant un modèle qui découvre la hiérarchie et les séquences de processus à partir du texte classifié. L'extraction de cet enchainement permettra de déterminer le déroulement des tâches qui composent un processus. Nous pouvons extraire les tâches qui composent un processus spécifique en regroupant les textes classifiés et en analysant les différentes étapes possibles du processus. Un acteur humain pourra évaluer les résultats et apporter une rétroaction utile à la recherche.

ANNEXE A

EXIGENCE 1 - RÉSULTAT DE L'EXPLORATION DES THÈMES

```
--------------------------------------------------
Nombre de thèmes:  1
--------------------------------------------------
Perplexité        :  -6.103520671295117
Cohérence         :  0.4989947200870154
Les termes dominants par thème
--------------------------------------------------
Thème 1 item  system  library  report  pine  patron  would  count  ability  management
--------------------------------------------------
Nombre de thèmes:  2
--------------------------------------------------
Perplexité        :  -6.0647813528643555
Cohérence         :  0.4567522797651308
Les termes dominants par thème
--------------------------------------------------
Thème 1 item  system  patron  count  name  description  library  record  hold  use
Thème 2 pine  report  would  library  management  priority  req  source  process  system
--------------------------------------------------
Nombre de thèmes:  3
--------------------------------------------------
Perplexité        :  -6.110801932612332
Cohérence         :  0.4305663217616213
Les termes dominants par thème
--------------------------------------------------
Thème 1 item  count  system  name  patron  use  library  description  record  ability
Thème 2 pine  report  would  management  req  priority  source  library  process  system
Thème 3 library  system  staff  patron  check  description  transaction  template
display  item
--------------------------------------------------
Nombre de thèmes:  4
--------------------------------------------------
Perplexité        :  -6.177402682645807
Cohérence         :  0.39669857411065257
Les termes dominants par thème
--------------------------------------------------
Thème 1 name  count  item  patron  use  record  system  staff  description  tool
Thème 2 pine  report  would  req  priority  source  library  system  patron  list
Thème 3 staff  library  hold  system  check  renewal  description  display  user  manager
Thème 4 item  library  system  ability  process  management  record  last  provide
shall
--------------------------------------------------
Nombre de thèmes:  5
--------------------------------------------------
Perplexité        :  -6.208821193432635
Cohérence         :  0.4221458613432457
Les termes dominants par thème
--------------------------------------------------
```

```
Thème 1 name item use count record patron report system library financial
Thème 2 pine would report req priority source library patron list group
Thème 3 staff display template charge payment user item design include list
Thème 4 management process system library software tool shall record requirement
specification
Thème 5 item system ability description hold last library count check patron
--------------------------------------------------
Nombre de thèmes: 6
--------------------------------------------------
Perplexité        :  -6.249101667501587
Cohérence         :  0.4026812773428361
Les termes dominants par thème
--------------------------------------------------
Thème 1 use library report staff pine patron system count group item
Thème 2 pine would req priority report source library demand can list
Thème 3 staff library charge template payment user last fine software account
Thème 4 system process management include transaction library software requirement
shall renewal
Thème 5 item name system hold description ability report library check patron
Thème 6 count patron item circulation month tool record management circ status
--------------------------------------------------
Nombre de thèmes: 7
--------------------------------------------------
Perplexité        :  -6.287127294552944
Cohérence         :  0.39961675781672285
Les termes dominants par thème
--------------------------------------------------
Thème 1 report pine record group use library staff name system user
Thème 2 source library report system process patron management demand requirement
can
Thème 3 template display staff attribute open item fine requirement patron
system
Thème 4 management tool transaction patron record process category ability item
library
Thème 5 name item ability system report description hold check location record
Thème 6 would pine priority req count item patron month library circulation
Thème 7 list system item report library last staff run transit ability
--------------------------------------------------
Nombre de thèmes: 8
--------------------------------------------------
Perplexité        :  -6.304015650920883
Cohérence         :  0.39536715850378734
Les termes dominants par thème
--------------------------------------------------
Thème 1 report pine count use name library group record staff system
Thème 2 source system report requirement library patron pine demand can process
Thème 3 library staff item payment charge patron template display fine service
Thème 4 circulation library total ability circ count transaction include software
cat
Thème 5 ability report item name run description system book check branch
Thème 6 would pine req priority patron item count month system library
Thème 7 item system hold list library last transit count description staff
Thème 8 management tool report name record process patron category interface
ability
--------------------------------------------------
Nombre de thèmes: 9
--------------------------------------------------
Perplexité        :  -6.2701008395156235
Cohérence         :  0.4073671168239789
Les termes dominants par thème
--------------------------------------------------
Thème 1 use pine count report library work group cat name system
Thème 2 report patron count demand can circulation would circ edit add
Thème 3 staff payment item charge template library software display manager
check
```

```
Thème 4 library system process requirement management include software total
demographic loan
Thème 5 name item check hold description ability system location material branch
Thème 6 would pine priority req library service item provide survey patron
Thème 7 item system list last library description transit staff copy hold
Thème 8 management record tool process category month patron financial name
shall
Thème 9 source report system library process level management run ability
provide
```

ANNEXE B

EXIGENCE 2 – RÉSULTAT DE L'EXPLORATION DES THÈMES

```
--------------------------------------------------
Nombre de thèmes:  1
--------------------------------------------------
Perplexité        :  -6.002461253218163
Cohérence         :  0.40019751961226824
Les termes dominants par thème
--------------------------------------------------
Thème 1 user  file  response  system  action  virtual  click  feature  student  req
--------------------------------------------------
Nombre de thèmes:  2
--------------------------------------------------
Perplexité        :  -5.918921557283748
Cohérence         :  0.48698289226753405
Les termes dominants par thème
--------------------------------------------------
Thème 1 student  user  feature  file  document  use  professor  video  able  need
Thème 2 user  response  file  action  click  system  virtual  req  show  menu
--------------------------------------------------
Nombre de thèmes:  3
--------------------------------------------------
Perplexité        :  -5.89172274449067
Cohérence         :  0.42173634802794285
Les termes dominants par thème
--------------------------------------------------
Thème 1 student  document  user  feature  video  professor  able  file  audio  priority
Thème 2 user  response  file  action  click  show  menu  window  button  select
Thème 3 system  requirement  user  virtual  software  space  feature  exam  use
functional
--------------------------------------------------
Nombre de thèmes:  4
--------------------------------------------------
Perplexité        :  -5.853038542563339
Cohérence         :  0.3924647174097377
Les termes dominants par thème
--------------------------------------------------
Thème 1 student  document  feature  video  able  audio  user  priority  professor  file
Thème 2 user  response  action  file  click  show  menu  button  select  window
Thème 3 requirement  system  use  user  software  functional  exam  need  feature
section
Thème 4 user  req  space  virtual  system  must  class  limit  need  software
--------------------------------------------------
Nombre de thèmes:  5
--------------------------------------------------
Perplexité        :  -5.8850299528840475
Cohérence         :  0.3999562855641835
```

```
Les termes dominants par thème
--------------------------------------------------
Thème 1 document video audio student feature able priority description professor
user
Thème 2 user response action menu click req pop new show font
Thème 3 system user software use must need file feature may instant
Thème 4 user space req class virtual browser response safari limit sequence
Thème 5 user file action response click select virtual window button requirement
--------------------------------------------------
Nombre de thèmes:  6
--------------------------------------------------
Perplexité        :  -5.936465437640481
Cohérence         :  0.4403834929981267
Les termes dominants par thème
--------------------------------------------------
Thème 1 video feature audio student document able professor transfer lecture
file
Thème 2 user response show new feature interface menu profile font choose
Thème 3 system use need exam file feature instal user software student
Thème 4 user req must space virtual system limit class browser need
Thème 5 user virtual response test show log available requirement system action
Thème 6 user file action click button window select response upload menu
--------------------------------------------------
Nombre de thèmes:  7
--------------------------------------------------
Perplexité        :  -5.950922402057845
Cohérence         :  0.4448829263971617
Les termes dominants par thème
--------------------------------------------------
Thème 1 video audio student document feature able professor lecture transfer
file
Thème 2 user response feature show interface font new req menu choose
Thème 3 system use need exam feature student excel section microsoft current
Thème 4 user system class req must file response would sequence stimulus
Thème 5 user response show available log virtual test menu feature system
Thème 6 user file action click button window select response pop upload
Thème 7 requirement virtual user software space exam req description functional
priority
--------------------------------------------------
Nombre de thèmes:  8
--------------------------------------------------
Perplexité        :  -5.930055161186734
Cohérence         :  0.45577961003101297
Les termes dominants par thème
--------------------------------------------------
Thème 1 student feature document able professor video lecture audio project
classroom
Thème 2 user interface response video profile feature show font give menu
Thème 3 system user must req software instant pop enable use messaging
Thème 4 user space virtual req class browser response safari stimulus sequence
Thème 5 response show available user feature test menu system new log
Thème 6 user file action click button window select response upload menu
Thème 7 requirement user virtual description functional system action priority
log need
Thème 8 user virtual exam req use windows xp vista file need time mac_os
--------------------------------------------------
Nombre de thèmes:  9
--------------------------------------------------
Perplexité        :  -5.93756061278047
Cohérence         :  0.4802388585775455
Les termes dominants par thème
--------------------------------------------------
Thème 1 video student audio feature professor able lecture document give real
Thème 2 user show feature response interface new menu font choose different
Thème 3 system user use instal browser must excel web software file
```

```
Thème 4 space file would upload allow release section folder student document
Thème 5 test file response select window show user student professor display
Thème 6 user action file click button select window menu upload response
Thème 7 requirement virtual user software description action functional log
priority ed
Thème 8 virtual user exam space need req response window pop file
Thème 9 user system req response must limit feature available show log
```

ANNEXE C

EXIGENCE 3 – RÉSULTAT DE L'EXPLORATION DES THÈMES

```
--------------------------------------------------
Nombre de thèmes:  1
--------------------------------------------------
Perplexité        :  -5.86
Cohérence         :  0.28
Les termes dominants par thème
--------------------------------------------------
Thème 1 document  user  pdf  file  split  requirement  merge  software  page  select
--------------------------------------------------
Nombre de thèmes:  2
--------------------------------------------------
Perplexité        :  -5.78
Cohérence         :  0.31
Les termes dominants par thème
--------------------------------------------------
Thème 1 document  pdf  user  file  split  requirement  merge  page  software  select
Thème 2 prefix  pdfsam  user  description  project  number  java  license  output  use
--------------------------------------------------
Nombre de thèmes:  3
--------------------------------------------------
Perplexité        :  -5.76
Cohérence         :  0.38
Les termes dominants par thème
--------------------------------------------------
Thème 1 document  pdf  user  split  file  merge  requirement  software  page
specification
Thème 2 project  user  source  log  software  pdfsam  license  panel  http  gnu
Thème 3 use  prefix  file  environment  user  output  save  number  order  page
--------------------------------------------------
Nombre de thèmes:  4
--------------------------------------------------
Perplexité        :  -5.81
Cohérence         :  0.37
Les termes dominants par thème
--------------------------------------------------
Thème 1 document  file  user  pdf  page  system  feature  select  output  want
Thème 2 project  source  software  license  panel  program  http  gnu  gpl  free
Thème 3 environment  use  file  save  page  case  java  select  pdfsam  user
Thème 4 requirement  split  software  merge  pdf  specification  user  prefix
interface  number
--------------------------------------------------
Nombre de thèmes:  5
--------------------------------------------------
Perplexité        :  -5.84
Cohérence         :  0.39
```

```
Les termes dominants par thème
-----------------------------------------------------
Thème 1 file user pdf system feature document select output pdfsam page
Thème 2 panel gui plugin consist log mix follow part basic alternate
Thème 3 environment use java page save select description load file user
Thème 4 user prefix number application use interface option
stimulus response sequence level set
Thème 5 document split requirement pdf merge software specification page
functional interface
-----------------------------------------------------
Nombre de thèmes: 6
-----------------------------------------------------
Perplexité        :  -5.82
Cohérence         :  0.40
Les termes dominants par thème
-----------------------------------------------------
Thème 1 file user document output pdf select pdfsam want version project
Thème 2 description project software user source confirmation default open ask
overwrite
Thème 3 page rotate file select java document pdf user reverse order
Thème 4 user prefix number application pdfsam use stimulus_response_sequence
log level message
Thème 5 split pdf requirement document merge software specification page
functional basic
Thème 6 feature system environment panel plugin save load interface gui case
-----------------------------------------------------
Nombre de thèmes: 7
-----------------------------------------------------
Perplexité        :  -5.82
Cohérence         :  0.41
Les termes dominants par thème
-----------------------------------------------------
Thème 1 file user document pdf select output page want version rotate
Thème 2 panel part log project consist gui follow purpose pdfsam message
Thème 3 use environment case java user save load file select page
Thème 4 user prefix number use stimulus_response_sequence level order start
change move
Thème 5 requirement document pdf merge split software specification page
functional user
Thème 6 feature system interface plugin application user gui screenshot
setting pdfsam
Thème 7 split source description software page file program gpl license input
-----------------------------------------------------
Nombre de thèmes: 8
-----------------------------------------------------
Perplexité        :  -5.87
Cohérence         :  0.38
Les termes dominants par thème
-----------------------------------------------------
Thème 1 file user pdf document output select pdfsam version application want
Thème 2 gui panel consist part follow pdfsam project plugin description org
Thème 3 page document select rotate environment user use pdf reverse save
Thème 4 prefix user number use order stimulus_response_sequence move change
variable name
Thème 5 requirement pdf document split merge software specification user
interface single
Thème 6 feature system panel screenshot functional plugin user setting visual
reorder
Thème 7 split description page input file source open program freedom
software
Thème 8 interface license user file software pdfsam output use java gnu
-----------------------------------------------------
Nombre de thèmes: 9
-----------------------------------------------------
Perplexité        :  -5.86
```

```
Cohérence         :  0.39
Les termes dominants par thème
-------------------------------------------------
Thème 1 file  user  pdf  document  output  select  version  want  pdfsam  folder
Thème 2 panel  log  gui  message  part  pdfsam  plugin  alternate  follow  consist
Thème 3 use  case  java  appendix  save  pdfsam  platform  environment  jvm  machine
Thème 4 prefix  number  user  stimulus_response_sequence  order  use  move  name
variable  unique
Thème 5 requirement  pdf  document  merge  split  software  specification  user
functional  extract
Thème 6 feature  system  user  application  panel  command  option  setting  visually
environment
Thème 7 description  split  file  page  environment  input  program  alt  load  window
Thème 8 interface  software  user  work  environment  file  source  free  change
default
Thème 9 page  rotate  document  select  plugin  reverse  interface  user  pdfsam
reorder
```

ANNEXE D

EXIGENCE 4 – RÉSULTAT DE L'EXPLORATION DES THÈMES

```
--------------------------------------------------
Nombre de thèmes:  1
--------------------------------------------------
Perplexité        :  -5.80
Cohérence         :  0.40
Les termes dominants par thème
--------------------------------------------------
Thème 1 attribute  vessel  declaration  uk  element  product  fishing  ers  must  code
--------------------------------------------------
Nombre de thèmes:  2
--------------------------------------------------
Perplexité        :  -5.89
Cohérence         :  0.42
Les termes dominants par thème
--------------------------------------------------
Thème 1 attribute  vessel  uk  element  ers  must  product  fishing  code  declaration
Thème 2 declaration  product  attribute  fishing  sub  message  elss  ers  must
logbook
--------------------------------------------------
Nombre de thèmes:  3
--------------------------------------------------
Perplexité        :  -5.69
Cohérence         :  0.50
Les termes dominants par thème
--------------------------------------------------
Thème 1 attribute  declaration  element  vessel  code  uk  fishing  sub  cif  ers
Thème 2 product  feature  must  elss  message  provide  logbook  operation  datum
electronic
Thème 3 uk  vessel  system  ers  datum  transmission  fishery  administration  elss
electronic
--------------------------------------------------
Nombre de thèmes:  4
--------------------------------------------------
Perplexité        :  -5.71
Cohérence         :  0.51
Les termes dominants par thème
--------------------------------------------------
Thème 1 vessel  use  fishing  uk  master  definition  system  regulation  length
electronic
Thème 2 product  elss  provide  must  logbook  electronic  supplier  op  mean  require
Thème 3 datum  operation  transmission  system  message  send  fishery  ers  uk
administration
Thème 4 attribute  declaration  element  vessel  uk  fishing  code  ers  sub  cif
--------------------------------------------------
Nombre de thèmes:  5
--------------------------------------------------
Perplexité        :  -5.67
```

```
Cohérence         :  0.46
Les termes dominants par thème
-------------------------------------------------
Thème 1 cif  vessel  element  attribute  pos  detail  sub  spe  see  number
Thème 2 declaration  sub  start  catch  gear  land  area  op  operation  fishing
Thème 3 version  code  fishery  datum  transmission  system  port  ec  ers  list
Thème 4 product  logbook  electronic  uk  vessel  requirement  fishing  elss  must
specification
Thème 5 attribute  ers  support  element  fishery  feature  name  item  uk  status
-------------------------------------------------
Nombre de thèmes:  6
-------------------------------------------------
Perplexité        :  -5.61
Cohérence         :  0.50
Les termes dominants par thème
-------------------------------------------------
Thème 1 cif  vessel  item  ref  number  net  gear  gill  fishing  fleet
Thème 2 declaration  sub  start  land  area  gear  tag_indicate  fishing  operation
message
Thème 3 version  code  port  list  ec  fishery  find  vessel  ers  catch
Thème 4 must  format  define  message  fix  pattern  time  yyyy  value  conform
Thème 5 uk  fishery  administration  complete  commercial  datum  elss  copyright
system  transmission
Thème 6 attribute  element  product  electronic  logbook  support  ers  requirement
fishing  feature
-------------------------------------------------
Nombre de thèmes:  7
-------------------------------------------------
Perplexité        :  -5.60
Cohérence         :  0.47
Les termes dominants par thème
-------------------------------------------------
Thème 1 cif  vessel  item  ref  fishing  zone  gear  receive  registration  norway
Thème 2 declaration  sub  spe  gear  catch  area  transhipment  specie  position
relevant
Thème 3 code  version  port  ec  fishery  vessel  list  find  far  fishing
Thème 4 message  operation  start  datum  transmission  send  must  log  correction
elss
Thème 5 uk  fishery  administration  ers  complete  commercial  copyright  system  rlc
tra
Thème 6 product  electronic  logbook  requirement  uk  vessel  fishing  elss
specification  functional
Thème 7 attribute  element  name  support  define  status  description  format  note
ers
-------------------------------------------------
Nombre de thèmes:  8
-------------------------------------------------
Perplexité        :  -5.67
Cohérence         :  0.47
Les termes dominants par thème
-------------------------------------------------
Thème 1 element  spe  attribute  detail  sub  pos  see  gea  far  fishing
Thème 2 declaration  sub  gear  land  area  acknowledgement  specie  relevant  message
deployment
Thème 3 code  port  list  find  ec  vessel  iso_alpha_country  country  letter
fishery
Thème 4 send  message  must  correction  log  transmit  voyage  deletion  datum
return
Thème 5 datum  start  system  transmission  elss  message  capture  operation  email
require
Thème 6 product  electronic  logbook  requirement  uk  fishing  vessel  specification
functional  declaration
Thème 7 attribute  element  name  syntax  code  op  number  definition  ers  ret
Thème 8 ers  fishery  cif  vessel  uk  feature  format  item  define  support
-------------------------------------------------
```

```
Nombre de thèmes:  9
--------------------------------------------------
Perplexité        :  -5.58
Cohérence         :  0.52
Les termes dominants par thème
--------------------------------------------------
Thème 1 item  vessel  ref  zone  receive  fishing  master  code  registration  number
Thème 2 declaration  sub  spe  start  catch  gear  area  specie  relevant  immediately
Thème 3 code  port  ec  fishery  ers  find  list  far  vessel  lan
Thème 4 ers  message  must  elss  datum  transmission  send  transmit  correction  log
Thème 5 uk  fishery  administration  complete  commercial  copyright  system  number
record  operation
Thème 6 product  electronic  logbook  requirement  uk  fishing  vessel  specification
functional  declaration
Thème 7 attribute  element  name  syntax  code  pos  xml  da  gbrrn  copy
Thème 8 cif  support  status  feature  description  note  version  content  net
vessel
Thème 9 define  format  fix  pattern  yyyy  time  must  conform  value  utc_standard
```

ANNEXE E

EXIGENCE 5 – RÉSULTAT DE L'EXPLORATION DES THÈMES

```
--------------------------------------------------
Nombre de thèmes:  1
--------------------------------------------------
Perplexité        :  -5.63
Cohérence         :  0.41
Les termes dominants par thème
--------------------------------------------------
Thème 1 file  hat  display  user  node  shall  select  application  program  text
--------------------------------------------------
Nombre de thèmes:  2
--------------------------------------------------
Perplexité        :  -5.70
Cohérence         :  0.44
Les termes dominants par thème
--------------------------------------------------
Thème 1 file  hat  display  node  shall  user  gui  select  application  sdt
Thème 2 file  user  select  program  text  transformation  display  hat  application
shall
--------------------------------------------------
Nombre de thèmes:  3
--------------------------------------------------
Perplexité        :  -5.49
Cohérence         :  0.54
Les termes dominants par thème
--------------------------------------------------
Thème 1 display  node  user  select  shall  application  text  use  srsreq  file
Thème 2 software  system  requirement  roach  develop  description  team  definition
hat  behavioral
Thème 3 file  program  hat  transformation  language  output  target  parse  sml  gui
--------------------------------------------------
Nombre de thèmes:  4
--------------------------------------------------
Perplexité        :  -5.49
Cohérence         :  0.44
Les termes dominants par thème
--------------------------------------------------
Thème 1 search  tree  string  match  sub  sdt  table  appendix  display  window
Thème 2 software  use  requirement  case  specification  end  refer  step  scenario
continue
Thème 3 node  program  transformation  display  file  language  target  parse  sdt
graph
Thème 4 file  user  hat  shall  select  text  application  gui  srsreq  pretty
--------------------------------------------------
Nombre de thèmes:  5
--------------------------------------------------
Perplexité        :  -5.51
Cohérence         :  0.41
```

```
Les termes dominants par thème
-------------------------------------------------
Thème 1 search tree sdt display string window sub table node appendix
Thème 2 software requirement specification scenario describe behavioral
description develop step definition
Thème 3 program node transformation display language file target parse sdt
output
Thème 4 user hat shall file select application text use qui srsreq
Thème 5 file parser system time host os define tlp function grammar send
-------------------------------------------------
Nombre de thèmes: 6
-------------------------------------------------
Perplexité       : -5.43
Cohérence        : 0.47
Les termes dominants par thème
-------------------------------------------------
Thème 1 tree search table sub string appendix menu example give roach
Thème 2 use software requirement case specification end refer step application
continue
Thème 3 display node sdt text window graph select shall user correspond
Thème 4 user shall application select file srsreq hat qui system text
Thème 5 system scroll define window java platform operate function develop
core
Thème 6 file program hat transformation pretty language output target sml
print
-------------------------------------------------
Nombre de thèmes: 7
-------------------------------------------------
Perplexité       : -5.46
Cohérence        : 0.51
Les termes dominants par thème
-------------------------------------------------
Thème 1 tree search string sub sdt table match navigation criterion contain
Thème 2 use software requirement case specification end refer step continue
scenario
Thème 3 node display sdt select graph user shall expand descendant correspond
Thème 4 hat qui shall user search new current enter directory figure
Thème 5 text window display cursor system shall keyboard srsreq location
highlight
Thème 6 file program transformation hat pretty language target output print
parse
Thème 7 application user select file srsreq shall editor hat system save
-------------------------------------------------
Nombre de thèmes: 8
-------------------------------------------------
Perplexité       : -5.49
Cohérence        : 0.46
Les termes dominants par thème
-------------------------------------------------
Thème 1 display sdt node tree window string graph sub search correspond
Thème 2 use case end refer step application select alt user system
Thème 3 node program transformation target file parse language display select
sdt
Thème 4 user hat search shall cursor qui text directory current system
Thème 5 system tlp scroll window shall operate leaf develop platform java
Thème 6 hat file qui shall srsreq sml output name program user
Thème 7 select text application user file pretty print editor configuration
save
Thème 8 software requirement specification table roach communication relate
item team menu
-------------------------------------------------
Nombre de thèmes: 9
-------------------------------------------------
Perplexité       : -5.44
Cohérence        : 0.49
```

```
Les termes dominants par thème
-------------------------------------------------
Thème 1 search  tree  sdt  sub  display  window  appendix  string  criterion
navigation
Thème 2 software  use  requirement  case  specification  end  refer  step  continue
system
Thème 3 node  display  graph  sdt  select  expand  shall  scenario  type  descendant
Thème 4 user  application  select  hat  shall  gui  file  save  system  alt
Thème 5 cursor  system  window  keyboard  location  display  move  mouse  shall  key
Thème 6 file  srsreq  name  shall  gui  parser  associate  hat  time  editor
Thème 7 text  pretty  print  display  shall  output  editor  select  srsreq  sdt
Thème 8 search  table  string  match  text  menu  leave  right  roach  character
Thème 9 program  hat  transformation  file  language  target  sml  parse  gui  output
```

ANNEXE F

EXIGENCE 6 – RÉSULTAT DE L'EXPLORATION DES THÈMES

```
--------------------------------------------------
Nombre de thèmes:  1
--------------------------------------------------
Perplexité        :  -6.50
Cohérence         :  0.45
Les termes dominants par thème
--------------------------------------------------
Thème 1 vcd  evidence  datum  national  requirement  package  service  economic
operator  provide
--------------------------------------------------
Nombre de thèmes:  2
--------------------------------------------------
Perplexité        :  -6.57
Cohérence         :  0.44
Les termes dominants par thème
--------------------------------------------------
Thème 1 vcd  package  requirement  service  evidence  economic  national  datum
operator  system
Thème 2 vcd  datum  evidence  national  provide  operator  requirement  economic
criterion  system
--------------------------------------------------
Nombre de thèmes:  3
--------------------------------------------------
Perplexité        :  -6.48
Cohérence         :  0.38
Les termes dominants par thème
--------------------------------------------------
Thème 1 vcd  evidence  datum  national  requirement  package  service  economic
operator  provide
Thème 2 criterion  atomic  personal  final  sub  social  convict  fiscal  location
france
Thème 3 eprocurement  contract_notice  government  union  te  public  base
publication  high  engagement
--------------------------------------------------
Nombre de thèmes:  4
--------------------------------------------------
Perplexité        :  -6.42
Cohérence         :  0.52
Les termes dominants par thème
--------------------------------------------------
Thème 1 service  national  provider  vcd  attestation  public  level  legal  solution
stage
Thème 2 national  criterion  mapping  european  evidence  need  assessment  authority
tender  attestation
Thème 3 datum  package  vcd  system  evidence  specific  provide  operator  economic
context
```

```
Thème 4 requirement functional approval pende peppol vcd specification version
deliverable table
--------------------------------------------------
Nombre de thèmes: 5
--------------------------------------------------
Perplexité        :  -6.46
Cohérence         :  0.53
Les termes dominants par thème
--------------------------------------------------
Thème 1 service national provider vcd level legal stage attestation economic
solution
Thème 2 national criterion european mapping need evidence authority tender
assessment tool
Thème 3 datum package vcd system evidence specific operator economic provide
context
Thème 4 document implementation table figure follow pilot stage approach type
define
Thème 5 requirement functional approval pende peppol vcd specification version
deliverable certificate
--------------------------------------------------
Nombre de thèmes: 6
--------------------------------------------------
Perplexité        :  -6.47
Cohérence         :  0.52
Les termes dominants par thème
--------------------------------------------------
Thème 1 service economic operator provider national attestation level vcd
stage provide
Thème 2 national criterion mapping european need evidence authority assessment
tender tool
Thème 3 package datum vcd system evidence specific provide context stage
economic
Thème 4 approval pende version ec table approach necessary fulfilment document
stakeholder
Thème 5 requirement functional vcd peppol specification deliverable document
electronic legal certificate
Thème 6 datum identity system record equivalent evidence feature document
standard access management
--------------------------------------------------
Nombre de thèmes: 7
--------------------------------------------------
Perplexité        :  -6.49
Cohérence         :  0.50
Les termes dominants par thème
--------------------------------------------------
Thème 1 approval pende version service provider level legal vcd list national
Thème 2 national european criterion authority mapping need assessment tender
economic evidence
Thème 3 package vcd system datum economic operator provide evidence must
service
Thème 4 document implementation declaration table type legal evidence country
approach necessary
Thème 5 requirement vcd functional peppol specification deliverable stage
solution mapping pre
Thème 6 description electronic procurement document public certificate
information semantic procedure use
Thème 7 evidence datum attestation specific context stage provide translation
member state certificate
--------------------------------------------------
Nombre de thèmes: 8
--------------------------------------------------
Perplexité        :  -6.47
Cohérence         :  0.46
Les termes dominants par thème
--------------------------------------------------
```

```
Thème 1 approval pende version level support step maturity ec weakness
dependency
Thème 2 national european criterion mapping authority need assessment tender
evidence attestation
Thème 3 package datum vcd economic operator system evidence specific provide
service
Thème 4 legal requirement analysis stakehold framework table define
quality attribute organisational document
Thème 5 requirement functional vcd peppol specification deliverable feature
system authentication infrastructure
Thème 6 eprocurement directive european public article interface evidence
collection source feature describes
Thème 7 figure identity access_management evidence suitability account stage
reference actor criterion
Thème 8 service attestation document national stage vcd evidence use datum
electronic
--------------------------------------------------
Nombre de thèmes:  9
--------------------------------------------------
Perplexité        :  -6.50
Cohérence         :  0.51
Les termes dominants par thème
--------------------------------------------------
Thème 1 service provider national vcd level article european trust list
directive
Thème 2 national economic operator criterion authority european need assessment
contracting evidence
Thème 3 package datum vcd evidence specific provide context stage system
operator
Thème 4 type suitability stage table requirement define legal document
organisational approach
Thème 5 requirement functional vcd peppol specification deliverable
transportation feature infrastructure instance
Thème 6 system identity view access management infrastructure datum use
authentication functionality quality attribute
Thème 7 approval pende version figure mapping feature pre vision tool level
Thème 8 attestation stage evidence document certificate description mapping use
measurement information
Thème 9 datum electronic must document business infrastructure stakeholder
ensure transport exchange
```

ANNEXE G

EXIGENCE 7 – RÉSULTAT DE L'EXPLORATION DES THÈMES

```
--------------------------------------------------
Nombre de thèmes:  1
--------------------------------------------------
Perplexité        :  -5.91
Cohérence         :  0.41
Les termes dominants par thème
--------------------------------------------------
Thème 1 system  shall  home  digitalhome  requirement  user  use  digital  device
sensor
--------------------------------------------------
Nombre de thèmes:  2
--------------------------------------------------
Perplexité        :  -5.89
Cohérence         :  0.35
Les termes dominants par thème
--------------------------------------------------
Thème 1 home  digital  requirement  ieee  project  shall  system  srs  inc  use
Thème 2 system  shall  digitalhome  home  user  device  sensor  security  use
requirement
--------------------------------------------------
Nombre de thèmes:  3
--------------------------------------------------
Perplexité        :  -5.92
Cohérence         :  0.40
Les termes dominants par thème
--------------------------------------------------
Thème 1 ieee  requirement  shall  temperature  software  system  user  engineer
standard  humidistat
Thème 2 shall  system  home  digitalhome  device  user  sensor  use  controller
appliance
Thème 3 system  home  requirement  digital  project  security  srs  digitalhome  time
shall
--------------------------------------------------
Nombre de thèmes:  4
--------------------------------------------------
Perplexité        :  -5.94
Cohérence         :  0.41
Les termes dominants par thème
--------------------------------------------------
Thème 1 ieee  shall  home  control  user  temperature  system  software  inc  humidity
Thème 2 shall  system  digitalhome  home  device  sensor  controller  user  appliance
provide
Thème 3 home  digital  project  srs  system  time  plan  period  set  use
Thème 4 system  requirement  security  shall  digitalhome  alarm  user  use
humidistat  thermostat
--------------------------------------------------
Nombre de thèmes:  5
```

```
--------------------------------------------------
Perplexité        :  -5.96
Cohérence         :  0.41
Les termes dominants par thème
--------------------------------------------------
Thème 1 ieee  shall  temperature  control  home  user  system  engineer  humidity
thermostat
Thème 2 shall  sensor  device  home  system  digitalhome  controller  power  user
switch
Thème 3 home  digital  project  srs  system  time  plan  use  default  set
Thème 4 requirement  system  security  shall  description  user  alarm  use
construction  cycle
Thème 5 digitalhome  system  shall  version  homeowner  requirement  product  digital
specification  document
--------------------------------------------------
Nombre de thèmes:  6
--------------------------------------------------
Perplexité        :  -6.01
Cohérence         :  0.40
Les termes dominants par thème
--------------------------------------------------
Thème 1 ieee  home  shall  user  temperature  control  humidity  standard  inc
engineer
Thème 2 shall  system  sensor  device  home  digitalhome  controller  able  switch
communication
Thème 3 home  project  digital  srs  time  plan  set  system  construction  default
Thème 4 requirement  security  system  user  description  breach  set  homeowner
functional  contact
Thème 5 digitalhome  system  version  shall  requirement  digital  provide  use
homeowner  document
Thème 6 system  shall  thermostat  humidistat  temperature  air  space  humidity  web
conditioning
--------------------------------------------------
Nombre de thèmes:  7
--------------------------------------------------
Perplexité        :  -6.00
Cohérence         :  0.40
Les termes dominants par thème
--------------------------------------------------
Thème 1 shall  temperature  user  set  control  able  thermostat  device  level
humidity
Thème 2 shall  device  home  system  user  digitalhome  sensor  controller  appliance
able
Thème 3 home  project  digital  srs  time  plan  period  system  use  default
Thème 4 requirement  system  security  user  change  safety  functional  set  general
failure
Thème 5 ieee  system  digital  inc  version  digitalhome  homeowner  shall
requirement  software
Thème 6 system  web  day  humidistat  familiar  setting  schedule  thermostat
technician  temperature
Thème 7 system  shall  security  sensor  digitalhome  home  contact  light
temperature  humidity
--------------------------------------------------
Nombre de thèmes:  8
--------------------------------------------------
Perplexité        :  -6.04
Cohérence         :  0.37
Les termes dominants par thème
--------------------------------------------------
Thème 1 requirement  thermostat  temperature  home  control  user  shall  able  set
level
Thème 2 shall  device  home  system  user  sensor  digitalhome  use  switch
controller
Thème 3 home  project  digital  srs  time  period  use  set  system  plan
```

```
Thème 4 requirement system security user change safety set documentation dh
information
Thème 5 ieee homeowner inc software digitalhome development standard
specification system version
Thème 6 humidistat humidity thermostat temperature shall space sensor system
web day
Thème 7 shall system security digitalhome home user light temperature alarm
humidity
Thème 8 system shall datum requirement plan version backup provide setting
digital
--------------------------------------------------
Nombre de thèmes: 9
--------------------------------------------------
Perplexité       : -6.07
Cohérence        : 0.42
Les termes dominants par thème
--------------------------------------------------
Thème 1 level information testing project temperature home control system
shall different
Thème 2 shall appliance sensor home controller system able power switch
management
Thème 3 home digital srs project device system server web case model
Thème 4 requirement system security day change set humidistat thermostat
include general
Thème 5 ieee version software digitalhome requirement inc standard homeowner
user product
Thème 6 sensor contact shall system alarm power switch controller web
familiar
Thème 7 system shall home digitalhome control use specification device user
unit
Thème 8 system shall datum provide backup internet service digitalhome digital
technician
Thème 9 shall time temperature user plan set humidity thermostat period space
```

ANNEXE H

EXIGENCE 8 – RÉSULTAT DE L'EXPLORATION DES THÈMES

```
--------------------------------------------------
Nombre de thèmes:  1
--------------------------------------------------
Perplexité        :  -5.844991013327651
Cohérence         :  0.3968219078415257
Les termes dominants par thème
--------------------------------------------------
Thème 1 customer  user  process  account  change  service  method  payment  use  new
--------------------------------------------------
Nombre de thèmes:  2
--------------------------------------------------
Perplexité        :  -5.920607606783408
Cohérence         :  0.4223877670578078
Les termes dominants par thème
--------------------------------------------------
Thème 1 customer  user  change  process  account  method  service  new  use  payment
Thème 2 customer  process  payment  account  user  service  use  method  change  would
--------------------------------------------------
Nombre de thèmes:  3
--------------------------------------------------
Perplexité        :  -5.927870817756175
Cohérence         :  0.3985677336176405
Les termes dominants par thème
--------------------------------------------------
Thème 1 change  customer  process  user  service  method  account  new  address  provision
Thème 2 erp  microsoft  nav  implement  real  financial  intelligence  scorecard  kpis
still
Thème 3 account  customer  payment  user  method  use  process  service  create  would
--------------------------------------------------
Nombre de thèmes:  4
--------------------------------------------------
Perplexité        :  -5.6960192613743965
Cohérence         :  0.4498553651514641
Les termes dominants par thème
--------------------------------------------------
Thème 1 process  change  service  user  new  address  method  provision  quote  would
Thème 2 customer  care  email  add  account  check  credit  send  task  use
Thème 3 payment  account  user  method  store  customer  interface  pay  option  present
Thème 4 customer  receive  iristel  account  bill  listing  order  contact  reseller
line
--------------------------------------------------
Nombre de thèmes:  5
--------------------------------------------------
Perplexité        :  -5.668373824412164
Cohérence         :  0.4845581966873671
Les termes dominants par thème
--------------------------------------------------
```

```
Thème 1 change customer service provision care status description product url
method
Thème 2 process address new follow migration exist service port lnp task
Thème 3 payment user method store account quote use option service class
Thème 4 customer account email add crm credit check use request create
Thème 5 would current account department process user reseller sale input
customer
--------------------------------------------------
Nombre de thèmes: 6
--------------------------------------------------
Perplexité       : -5.67795991003773
Cohérence        : 0.45168712004155437
Les termes dominants par thème
--------------------------------------------------
Thème 1 change customer care provision process would current description method
class
Thème 2 process address new exist follow migration status lnp port task
Thème 3 payment method account user store pay class wizard use option
Thème 4 customer add account create crm order email use go new
Thème 5 user department account reseller customer interface general select url
wireless
Thème 6 service user quote sale customer variable determine use method response
--------------------------------------------------
Nombre de thèmes: 7
--------------------------------------------------
Perplexité       : -5.682512568746152
Cohérence        : 0.43555326947234263
Les termes dominants par thème
--------------------------------------------------
Thème 1 change service user bill broadband information activation wizard system
line
Thème 2 process new address exist follow migration port create attach table
Thème 3 payment account user method store option pay present lead invoice
Thème 4 customer care email account add request crm send create task
Thème 5 user department input product port reseller account list url request
Thème 6 provision service use process method would status current description
quote
Thème 7 service customer management iristel portal general business click account
term
--------------------------------------------------
Nombre de thèmes: 8
--------------------------------------------------
Perplexité       : -5.672322374247788
Cohérence        : 0.45484723162869756
Les termes dominants par thème
--------------------------------------------------
Thème 1 change provision process status would description current api system
bill
Thème 2 item class activate method credit check catalogue specific use product
Thème 3 payment method account store user pay use option lead class
Thème 4 customer account add create crm order new use contact listing
Thème 5 user account department service input method interface list require
select
Thème 6 service email sale send user update quote term use phone
Thème 7 customer care product general display email selection iristel user
click
Thème 8 process address new exist follow migration quote variable determine
store
--------------------------------------------------
Nombre de thèmes: 9
--------------------------------------------------
Perplexité       : -5.631215122086581
Cohérence        : 0.4280419984571858
Les termes dominants par thème
--------------------------------------------------
```

```
Thème 1 change provision process status would description current api system
activation
Thème 2 process address new follow migration exist relay browse name manage
Thème 3 account payment user method pay store option invoice present use
Thème 4 customer account order crm bill listing create contact store credit
Thème 5 department account reseller user password create email interface lead
view
Thème 6 service sale use user add payment send update request customer
Thème 7 customer care email service val residential general credit receive
check
Thème 8 quote method account use variable user store trouble_ticket response
service
Thème 9 user product payment class method input item catalogue require option
```

ANNEXE I

EXIGENCE 9 – RÉSULTAT DE L'EXPLORATION DES THÈMES

```
--------------------------------------------------
Nombre de thèmes:  1
--------------------------------------------------
Perplexité        :  -6.295530984069275
Cohérence         :  0.36144807209347374
Les termes dominants par thème
--------------------------------------------------
Thème 1 customer  number  call  order  function  account  rout  profile  may  request
--------------------------------------------------
Nombre de thèmes:  2
--------------------------------------------------
Perplexité        :  -6.298225950582632
Cohérence         :  0.40289770233750466
Les termes dominants par thème
--------------------------------------------------
Thème 1 wholesale  call  customer  time  ordernumber  capacity  invoice  email  iristel
type str
Thème 2 customer  number  order  rout  account  profile  function  request  may  call
--------------------------------------------------
Nombre de thèmes:  3
--------------------------------------------------
Perplexité        :  -6.370641298694857
Cohérence         :  0.42841410332712515
Les termes dominants par thème
--------------------------------------------------
Thème 1 customer  call  iristel  variable  take  report  group  specific  continue
capacity
Thème 2 customer  number  order  rout  function  profile  request  account  must  array
Thème 3 call  wholesale  number  list  customer  ip  ordernumber  time  iristel  may
--------------------------------------------------
Nombre de thèmes:  4
--------------------------------------------------
Perplexité        :  -6.383329036131479
Cohérence         :  0.4062147336145285
Les termes dominants par thème
--------------------------------------------------
Thème 1 customer  call  variable  receive  take  capacity  invoice  type  lead  continue
Thème 2 customer  number  order  function  rout  request  profile  array  account  return
Thème 3 call  wholesale  number  list  customer  ip  time  iristel  ordernumber  may
Thème 4 customer  service  account  product  rep  sale  bill  call  iristel  amount
--------------------------------------------------
Nombre de thèmes:  5
--------------------------------------------------
Perplexité        :  -6.4565748277454
Cohérence         :  0.4305716911348826
```

```
Les termes dominants par thème
--------------------------------------------------
Thème 1 lnp wholesale customer account call continue ponnumber invoice charge
desire
Thème 2 number customer rout profile order function request return array account
Thème 3 wholesale number customer iristel time invoice call ordernumber may
account
Thème 4 customer service rep product call sale iristel email may inbound
Thème 5 call customer list update ip function order reject pend iristel
--------------------------------------------------
Nombre de thèmes:  6
--------------------------------------------------
Perplexité        :  -6.455346430733107
Cohérence         :  0.3882667872601462
Les termes dominants par thème
--------------------------------------------------
Thème 1 wholesale ordernumber type_str takes_one_variable function lnp customer
variable ponnumber take
Thème 2 number customer order function rout request profile return array status
Thème 3 number list invoice time iristel customer technical call may dial
Thème 4 call product customer may process group service specific setup record
Thème 5 call customer traffic update ip reject capacity pend iristel account
Thème 6 account customer rep sale report dashboard need information bill deposit
--------------------------------------------------
Nombre de thèmes:  7
--------------------------------------------------
Perplexité        :  -6.5094277544446975
Cohérence         :  0.407740088313483
Les termes dominants par thème
--------------------------------------------------
Thème 1 call variable record take customer well datetime startdate enddate
option
Thème 2 customer number rout profile channel order did note able may
Thème 3 wholesale number customer lnp service invoice ordernumber iristel dial
call
Thème 4 service product process customer may iristel inbound available format
lead
Thème 5 list customer ip call iristel pend traffic reject account service
Thème 6 account rep customer sale information email need general quote lead
Thème 7 function number request call order return array status must customer
--------------------------------------------------
Nombre de thèmes:  8
--------------------------------------------------
Perplexité        :  -6.541845821130133
Cohérence         :  0.4195538657486307
Les termes dominants par thème
--------------------------------------------------
Thème 1 wholesale lnp service customer testing iristel inbound ponnumber
management provision
Thème 2 customer account service lead create information number did channel
request
Thème 3 number wholesale invoice ordernumber phone ip dial customer iristel
account
Thème 4 product service process setup dashboard available may catalogue include
specific
Thème 5 list account iristel call user display email customer use quote
Thème 6 account rep sale number customer new report need issue api
Thème 7 number customer call profile rout function order array request may
Thème 8 time update function pend customer reject status return order request
--------------------------------------------------
Nombre de thèmes:  9
--------------------------------------------------
Perplexité        :  -6.553302756635503
Cohérence         :  0.4147253998912998
Les termes dominants par thème
```

```
--------------------------------------------------
Thème 1 invoice wholesale management timezone test platform portal comment flow
activation
Thème 2 customer rout profile did number service order able may channel
Thème 3 wholesale ordernumber type_str number lnp takes_one_variable variable
iristel function customer
Thème 4 product process group may available service include lead setup structure
Thème 5 list ip customer call pend account update email iristel reject
Thème 6 rep sale account customer report email need version history dashboard
Thème 7 number function call request array order return customer must note
Thème 8 customer number account option end report order did create service
Thème 9 time account customer iristel call record user provide information
support
```

ANNEXE J

EXIGENCE 10 – RÉSULTAT DE L'EXPLORATION DES THÈMES

```
--------------------------------------------------
Nombre de thèmes:  1
--------------------------------------------------
Perplexité        :  -5.760703454765456
Cohérence         :  0.3143892268941272
Les termes dominants par thème
--------------------------------------------------
Thème 1 listing  file  crm  list  account  name  partner  new  input  upload
--------------------------------------------------
Nombre de thèmes:  2
--------------------------------------------------
Perplexité        :  -5.746047658247888
Cohérence         :  0.3477642578986079
Les termes dominants par thème
--------------------------------------------------
Thème 1 file  crm  partner  name  list  account  input  upload  listing  code
Thème 2 listing  espresso  request  service  var  wholesale  inventory  new  mail
management
--------------------------------------------------
Nombre de thèmes:  3
--------------------------------------------------
Perplexité        :  -5.763758418570854
Cohérence         :  0.3398026457870084
Les termes dominants par thème
--------------------------------------------------
Thème 1 file  listing  name  input  account  list  upload  crm  partner  download
Thème 2 listing  request  service  mail  new  case  create  task  type  espresso
Thème 3 crm  npa  exchange  nxx  code  partner  new  var  listing  customer
--------------------------------------------------
Nombre de thèmes:  4
--------------------------------------------------
Perplexité        :  -5.85436875770767
Cohérence         :  0.39137031056870253
Les termes dominants par thème
--------------------------------------------------
Thème 1 name  account  list  partner  file  accept  crm  listing  submission  input
Thème 2 listing  request  espresso  var  service  inventory  create  case  task
management
Thème 3 crm  new  validation  customer  listing  filter  notification  entry  user
partner
Thème 4 file  exchange  code  input  csv  download  upload  template  nxx  sample
--------------------------------------------------
Nombre de thèmes:  5
--------------------------------------------------
Perplexité        :  -5.898623046792141
Cohérence         :  0.42539551047589796
Les termes dominants par thème
```

```
--------------------------------------------------
Thème 1 name account partner accept list submission reject crm customer code
Thème 2 listing request create case mail task service fix would order
Thème 3 crm entry user new filter listing street change npa notification
Thème 4 exchange code npa file nxx template csv format download table
Thème 5 listing file input crm espresso validation tab var download inventory
--------------------------------------------------
Nombre de thèmes:  6
--------------------------------------------------
Perplexité        :  -5.904871129579491
Cohérence         :  0.4247222802349068
Les termes dominants par thème
--------------------------------------------------
Thème 1 account crm partner accept reject name list submission upload listing
Thème 2 listing request service espresso type create activation task case fix
Thème 3 entry line user fill crm auto filter unique customer file
Thème 4 npa exchange nxx code format csv new table file detail
Thème 5 file listing input validation var list management inventory download
ftp
Thème 6 download mail name listing new order notification content business list
--------------------------------------------------
Nombre de thèmes:  7
--------------------------------------------------
Perplexité        :  -5.940041937717307
Cohérence         :  0.42162704677935386
Les termes dominants par thème
--------------------------------------------------
Thème 1 name account crm partner file upload accept validation listing reject
Thème 2 listing type task create request service fix activation case operation
Thème 3 entry customer partner user filter detail crm datum report accept
Thème 4 npa exchange code nxx format file csv table new list
Thème 5 listing file var list send inventory management download input ftp
Thème 6 new listing espresso mail template crm download order notification file
Thème 7 listing number display phone change edid report would sub configuration
--------------------------------------------------
Nombre de thèmes:  8
--------------------------------------------------
Perplexité        :  -5.923878707228488
Cohérence         :  0.44301155167826156
Les termes dominants par thème
--------------------------------------------------
Thème 1 name crm partner account accept reject upload submission file entry
Thème 2 listing request espresso type activation service fix task create account
Thème 3 customer partner entry accept name crm report status wholesaler reject
Thème 4 file format nxx npa csv sample code detail error list
Thème 5 listing file input var inventory ftp management active tab download
Thème 6 new listing mail download crm notification order name follow list
Thème 7 fill file auto number field spid detail phone create user
Thème 8 exchange npa code unique line nxx account crm report filter
--------------------------------------------------
Nombre de thèmes:  9
--------------------------------------------------
Perplexité        :  -5.935574128385861
Cohérence         :  0.431810881725432
Les termes dominants par thème
--------------------------------------------------
Thème 1 account name upload submission code request crm code date partner must
Thème 2 listing task service create activation request fix case activate queue
Thème 3 customer filter entry crm user line listing reject new contain
Thème 4 download csv file template format subject summary sample wholesale
address
Thème 5 listing partner list accept var active telus_national inventory customer
send
Thème 6 listing espresso file new upload mail tab input order notification
Thème 7 fill detail auto field file spid edid input number listing
```

Theme 8	adjustment unique entry listing ftp value view report table criterion
Theme 9	name crm npa exchange validation code nxx file input new

RÉFÉRENCES

Abadi, A., Nisenson, M., & Simionovici, Y. (2008). A traceability technique for specifications. Paper presented at the 16th IEEE International Conference on Program Comprehension (ICPC 2008).

Abebe, S. L., & Tonella, P. (2010). Natural language parsing of program element names for concept extraction. Paper presented at the 18th International Conference on Program Comprehension (ICPC 2010).

Afreen, H., & Bajwa, I. S. (2011). Generating UML class models from SBVR software requirements specifications. Paper presented at the 23rd Benelux Conference on Artificial Intelligence (BNAIC 2011).

Agirre, E., Banea, C., Cardie, C., Cer, D., Diab, M., Gonzalez-Agirre, A., ... Rigau, G. (2015). Semeval-2015 task 2: Semantic textual similarity, english, spanish and pilot on interpretability. Paper presented at the 9th international workshop on semantic evaluation (SemEval 2015).

Aitchison, J., Bawden, D., & Gilchrist, A. (2003). *Thesaurus construction and use: a practical manual*. Chicago: Fitzroy Dearden.

Aizenbud-Reshef, N., Nolan, B. T., Rubin, J., & Shaham-Gafni, Y. (2006). Model traceability. *IBM Systems Journal, 45*(3), 515-526.

Al-Safadi, L. A. (2009). Natural language processing for conceptual modeling. *International Journal of Digital Content Technology and its Applications, 3*(3), 47-59.

Aletras, N., & Stevenson, M. (2013). Evaluating topic coherence using distributional semantics. Paper presented at the 10th International Conference on Computational Semantics (IWCS 2013).

Ali, N., Cai, H., Hamou-Lhadj, A., & Hassine, J. (2019). Exploiting Parts-of-Speech for effective automated requirements traceability. *Information and Software Technology, 106,* 126-141.

Ali, N., Gueheneuc, Y. G., & Antoniol, G. (2011). Requirements traceability for object oriented systems by partitioning source code. Paper presented at the 18th Working Conference on Reverse Engineering (WCRE 2011).

Alshahrani, S., & Kapetanios, E. (2016). Are Deep Learning Approaches Suitable for Natural Language Processing?. Paper presented at the International Conference on Applications of Natural Language to Information (NLDB 2016).

Antoniol, G., Canfora, G., Casazza, G., De Lucia, A., & Merlo, E. (2002). Recovering traceability links between code and documentation. *IEEE transactions on software engineering, 28*(10), 970-983.

Antoniol, G., Canfora, G., De Lucia, A., & Merlo, E. (1999). Recovering code to documentation links in OO systems. Paper presented at the 6th Working Conference on Reverse Engineering (WCRE 1999).

Antoniol, G., Caprile, B., Potrich, A., & Tonella, P. (2000). Design-code traceability for object-oriented systems. *Annals of Software Engineering, 9*(1-2), 35-58.

Antoniol, G., Caprile, B., Potrich, A., & Tonella, P. (2001). Design-code traceability recovery: selecting the basic linkage properties. *Science of Computer Programming, 40*(2-3), 213-234.

Arellano, A., Carney, E., & Austin, M. A. (2015). Natural language processing of textual requirements. Paper presented at the 10th International Conference on Systems (ICONS 2015).

Arora, C., Sabetzadeh, M., Briand, L., & Zimmer, F. (2015a). Automated checking of conformance to requirements templates using natural language processing. *IEEE transactions on Software Engineering, 41*(10), 944-968.

Arora, C., Sabetzadeh, M., Briand, L., & Zimmer, F. (2016). Extracting domain models from natural-language requirements: approach and industrial evaluation. Paper presented at the 19th International Conference on Model Driven Engineering Languages and Systems (MODELS 2016).

Arora, C., Sabetzadeh, M., Goknil, A., Briand, L. C., & Zimmer, F. (2015b). NARCIA: an automated tool for change impact analysis in natural language requirements. Paper presented at the 10th Joint Meeting on Foundations of Software Engineering (ESEC/FSE 2015).

Asuncion, H. U., Asuncion, A. U., & Taylor, R. N. (2010). Software traceability with topic modeling. Paper presented at the 32nd International Conference on Software Engineering (ISCE 2010).

Aysolmaz, B., Leopold, H., Reijers, H. A., & Demirörs, O. (2018). A semi-automated approach for generating natural language requirements documents based on business process models. *Information and Software Technology, 93*, 14-29.

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

Bajwa, I. S., Samad, A., & Mumtaz, S. (2009). Object oriented software modeling using NLP based knowledge extraction. *European Journal of Scientific Research, 35*(01), 22-33.

Bakar, N. H., Kasirun, Z. M., & Salleh, N. (2015). Feature extraction approaches from natural language requirements for reuse in software product lines: A systematic literature review. *Journal of Systems and Software*, *106*, 132-149.

Bella, E. E., Gervais, M. P., Bendraou, R., Wouters, L., & Koudri, A. (2018). Semi-supervised Approach for Recovering Traceability Links in Complex Systems. Paper presented at the 23rd International Conference on Engineering of Complex Computer Systems (ICECCS 2018).

Bengio, Y., Ducharme, R., Vincent, P., & Jauvin, C. (2003). A neural probabilistic language model. *Journal of machine learning research*, *3*(2), 1137-1155.

Berry, D. M., Ferrari, A., & Gnesi, S. (2017). Assessing tools for defect detection in natural language requirements: Recall vs precision. Retrieved from https://pdfs.semanticscholar.org/bb70/310c2fad1648cdc31e9799733a52f6711311.pdf

Blei, D. M., & Lafferty, J. D. (2007). A correlated topic model of science. *The Annals of Applied Statistics*, *1*(1), 17-35.

Blei, D. M., Ng, A. Y., & Jordan, M. I. (2003). Latent dirichlet allocation. *Journal of machine Learning research*, *3*(Jan), 993-1022.

Borg, M., & Runeson, P. (2013). IR in software traceability: From a bird's eye view. Paper presented at the ACM/IEEE International Symposium on Empirical Software Engineering and Measurement (ESEM 2013).

Borg, M., Runeson, P., & Ardö, A. (2014). Recovering from a decade: a systematic mapping of information retrieval approaches to software traceability. *Empirical Software Engineering*, *19*(6), 1565-1616

Bozyi, F., & Deniz, K. (2016). AutoClass: Automatic Text to OOP Concept Identification Model. *International Journal of Computer Applications*, *150*(10).

Btoush, E. S., & Hammad, M. M. (2015). Generating ER diagrams from requirement specifications based on natural language processing. *International Journal of Database Theory and Application*, *8*(2), 61-70.

Cabral, G., & Sampaio, A. (2008). Automated formal specification generation and refinement from requirement documents. *Journal of the Brazilian Computer Society*, *14*(1), 87-106.

Capobianco, G., De Lucia, A., Oliveto, R., Panichella, A., & Panichella, S. (2009). Traceability recovery using numerical analysis. Paper presented at the 16th Working Conference on Reverse Engineering (WCRE 2009).

Capobianco, G., Lucia, A. D., Oliveto, R., Panichella, A., & Panichella, S. (2013). Improving IR-based traceability recovery via noun-based indexing of software artifacts. *Journal of Software: Evolution and Process*, *25*(7), 743-762.

Carlson, N., & Laplante, P. (2014). The NASA automated requirements measurement tool: a reconstruction. *Innovations in Systems and Software Engineering*, *10*(2), 77- 1.

Carvalho, G., Falcão, D., Barros, F., Sampaio, A., Mota, A., Motta, L., & Blackburn, M. (2014). NAT2TESTSCR: Test case generation from natural language requirements based on SCR specifications. *Science of Computer Programming*, *95*, 275-297.

Casamayor, A., Godoy, D., & Campo, M. (2010). Identification of non-functional requirements in textual specifications: A semi-supervised learning approach. *Information and Software Technology*, *52*(4), 436-445.

Casamayor, A., Godoy, D., & Campo, M. (2012a). Functional grouping of natural language requirements for assistance in architectural software design. *Knowledge-Based Systems*, *30*, 78-86.

Casamayor, A., Godoy, D., & Campo, M. (2012b). Mining textual requirements to assist architectural software design: a state of the art review. *Artificial Intelligence Review*, *38*(3), 173-191.

Chang, J., & Blei, D. M. (2010). Hierarchical relational models for document networks. *The Annals of Applied Statistics*, *4*(1), 124-150.

Chang, J., Gerrish, S., Wang, C., Boyd-Graber, J. L., & Blei, D. M. (2009). Reading tea leaves: How humans interpret topic models. Paper presented at the 23rd conference on neural information processing systems (NIPS 2009).

Charikar, M., Chekuri, C., Feder, T., & Motwani, R. (2004). Incremental clustering and dynamic information retrieval. *SIAM Journal on Computing*, *33*(6), 1417-1440.

Charlet, D., & Damnati, G. (2017). Simbow at SEMEVAL-2017 task 3: Soft-cosine semantic similarity between questions for community question answering. Paper presented at the 11th International Workshop on Semantic Evaluation (SemEval-2017).

Chen, L., & Zeng, Y. (2010). Automatic generation of UML diagrams from product requirements described by natural language. Paper presented at the International Design Engineering Technical Conferences and Computers and Information in Engineering Conference (ASME 2009).

Chen, X., & Grundy, J. (2011). Improving automated documentation to code traceability by combining retrieval techniques. Paper presented at the 26th IEEE/ACM International Conference on Automated Software Engineering (ASE 2011).

Chen, X., Hosking, J., & Grundy, J. (2011). A combination approach for enhancing automated traceability:(NIER track). Paper presented at the 33rd International Conference on Software Engineering (ICSE 2011).

Chowdhury, A., & McCabe, M. C. (1998). Improving information retrieval systems using part of speech tagging. TR-98-48, *Institute for Systems Research, Univ. of Maryland*, 1993.

Cleland-Huang, J., Gotel, O., & Zisman, A. (2012). *Software and systems traceability*. Heidelberg: Springer.

Cleland-Huang, J., Settimi, R., Duan, C., & Zou, X. (2005). Utilizing supporting evidence to improve dynamic requirements traceability. Paper presented at the 13th IEEE international conference on Requirements Engineering (RE 2005).

De Almeida Bordignon, A. C., Thom, L. H., Silva, T. S., Dani, V. S., Fantinato, M., & Ferreira, R. C. B. (2018). Natural Language Processing in Business Process Identification and Modeling: A Systematic Literature Review. Paper presented at the XIV Brazilian Symposium on Information Systems (SBIS 2018).

De Lucia, A., Marcus, A., Oliveto, R., & Poshyvanyk, D. (2012). Information retrieval methods for automated traceability recovery. *Software and systems traceability*. London: Springer.

De Santiago Junior, V. A., & Vijaykumar, N. L. (2012). Generating model-based test cases from natural language requirements for space application software. *Software Quality Journal*, *20*(1), 77-143.

Deerwester, S., Dumais, S. T., Furnas, G. W., Landauer, T. K., & Harshman, R. (1990). Indexing by latent semantic analysis. *Journal of the American society for information science, 41*(6), 391-407.

Dekhtyar, A., & Hayes, J. H. (2018). Automating Requirements Traceability: Two Decades of Learning from KDD. Paper presented at the 1st International Workshop on Learning from other Disciplines for Requirements Engineering (D4RE 2018).

Diaz, D., Bavota, G., Marcus, A., Oliveto, R., Takahashi, S., & De Lucia, A. (2013). Using code ownership to improve ir-based traceability link recovery. Paper presented at the 21st International Conference on Program Comprehension (ICPC 2013).

Dit, B., Guerrouj, L., Poshyvanyk, D., & Antoniol, G. (2011). Can better identifier splitting techniques help feature location?. Paper presented at the 19th International Conference on Program Comprehension (ICPC 2011).

Duan, C., & Cleland-Huang, J. (2007). Clustering support for automated tracing. Paper presented at the 22nd IEEE/ACM international conference on Automated software engineering (ASE 2007).

Egyed, A., & Grünbacher, P. (2005). Supporting software understanding with automated requirements traceability. *International Journal of Software Engineering and Knowledge Engineering*, *15*(05), 783-810.

Etzkorn, L. H., Bowen, L. L., & Davis, C. G. (1999). An approach to program understanding by natural language understanding. *Natural Language Engineering*, *5*(3), 219-236.

Fabbrini, F., Fusani, M., Gnesi, S., & Lami, G. (2001). An automatic quality evaluation for natural language requirements. Paper presented at the 7th International Workshop on Requirements Engineering: Foundation for Software Quality (REFSQ 2001).

Falessi, D., Cantone, G. & Canfora, G. (2013). Empirical principles and an industrial case study in retrieving equivalent requirements via natural language processing techniques. *IEEE Transactions on Software Engineering*, *39*(1), 18-44.

Fauzi, N. A. A., Hassan, R., & Shah, Z. A. (2017). Extraction of Non-Functional Requirement in Natural Language Document Using Keyword with Usability Quality Aspect. *International Journal of Software Engineering and Technology*, *2*(2).

Fellbaum, C. (1998). A semantic network of english: the mother of all WordNets. In *EuroWordNet: A multilingual database with lexical semantic networks* (pp. 137-148). Princeton, NJ :Springer.

Ferrari, A., Spagnolo, G. O., & Gnesi, S. (2017). Towards a Dataset for Natural Language Requirements Processing. Paper presented at the 23rd Working Conference on Requirements Engineering: Foundation of Software Quality (REFSQ 2017).

Ferreira, D., & Da Silva, A. R. (2008). Wiki supported collaborative requirements engineering. Paper presented at the 4th International Symposium on Wikis (WikiSym 2008).

De Almeida Ferreira, D., & Da Silva, A. R. (2012). RSLingo: An information extraction approach toward formal requirements specifications. Paper presented at the 2nd International Workshop on Model-Driven Requirements Engineering (MoDRE 2012).

Fliedl, G., Kop, C., Mayr, H. C., Salbrechter, A., Vöhringer, J., Weber, G., & Winkler, C. (2007). Deriving static and dynamic concepts from software requirements using sophisticated tagging. *Data & Knowledge Engineering*, *61*(3), 433-448.

Friedrich, F., Mendling, J., & Puhlmann, F. (2011). Process model generation from natural language text. Paper presented at the 23rd International Conference on Advanced Information Systems Engineering (CAiSE 2011).

Ganitkevitch, J., Van Durme, B., & Callison-Burch, C. (2013). PPDB: The paraphrase database. Paper presented at the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL 2013).

Gehrmann, S., Dernoncourt, F., Li, Y., Carlson, E. T., Wu, J. T., Welt, J., ... Celi, L. A. (2018). Comparing deep learning and concept extraction based methods for patient phenotyping from clinical narratives. *PloS one*, *13*(2).

Gervasi, V., & Zowghi, D. (2014). Supporting traceability through affinity mining. Paper presented at the 22nd International Requirements Engineering Conference (RE 2014).

Gethers, M., Oliveto, R., Poshyvanyk, D., & De Lucia, A. (2011). On integrating orthogonal information retrieval methods to improve traceability recovery. Paper presented at the 27th IEEE International Conference on Software Maintenance (ICSM 2011).

Ghosh, S., Elenius, D., Li, W., Lincoln, P., Shankar, N., & Steiner, W. (2016). ARSENAL: automatic requirements specification extraction from natural language. Paper presented at the 8th International NASA Formal Methods Symposium (NFM 2016).

Gotel, O. C., & Finkelstein, C. W. (1994). An analysis of the requirements traceability problem. Paper presented at the IEEE International Conference on Requirements Engineering (RE 1994).

Gotel, O., Cleland-Huang, J., Hayes, J. H., Zisman, A., Egyed, A., Grünbacher, P., & Antoniol, G. (2012). The quest for ubiquity: A roadmap for software and systems traceability research. Paper presented at the 20th IEEE international requirements engineering conference (RE 2012).

Hayes, J. H., Dekhtyar, A., & Osborne, J. (2003). Improving requirements tracing via information retrieval. Paper presented at the 11th IEEE International Requirements Engineering Conference (RE 2003).

Hayes, J. H., Dekhtyar, A., & Sundaram, S. K. (2005). Improving after-the-fact tracing and mapping: Supporting software quality predictions. *IEEE software, 22*(6), 30-37.

Hayes, J. H., Dekhtyar, A., & Sundaram, S. K. (2006). Advancing candidate link generation for requirements tracing: The study of methods. *IEEE Transactions on Software Engineering, 32*(1), 4-19.

Hayes, J. H., Dekhtyar, A., Sundaram, S. K., Holbrook, E. A., Vadlamudi, S., & April, A. (2007). Requirements Tracing On target (RETRO): improving software maintenance through traceability recovery. *Innovations in Systems and Software Engineering, 3*(3), 193-202.

Herchi, H., & Abdessalem, W. B. (2012). From user requirements to UML class diagram. Retrieved from https://pdfs.semanticscholar.org/c48d/7a4efeaf31f5c2e104f5c71dead4400d0703.pdf

Hofmann, T. (2001). Unsupervised learning by probabilistic latent semantic analysis. *Machine learning*, *42*(1-2), 177-196.

Hughes, M., Li, I., Kotoulas, S., & Suzumura, T. (2017). Medical text classification using convolutional neural networks. *Stud Health Technol Inform*, *235*, 246-250.

Hussain, I., Kosseim, L., & Ormandjieva, O. (2008). Using linguistic knowledge to classify non-functional requirements in SRS documents. Paper presented at the 13th International Conference on Application of Natural Language to Information Systems (NLDB 2008).

Hussain, I., Ormandjieva, O., & Kosseim, L. (2009). Mining and Clustering Textual Requirements to Measure Functional Size of Software with COSMIC. Paper presented at the International Software Engineering Research and Practice (SERP 2009).

Ibrahim, M., & Ahmad, R. (2010, May). Class diagram extraction from textual requirements using natural language processing (NLP) techniques. Paper presented at the 2nd Second International Conference on Computer Research and Development (ICCRD 2010).

Ilieva, M. (2007). Use Case Paths Model Revealing Through Natural Language Requirements Analysis. Paper presented at the 11th International Conference on Artificial Intelligence and Law (ICAIL 2007).

Ingwersen, P., & Järvelin, K. (2006). *The turn: Integration of information seeking and retrieval in context*. Netherlands: Springer Science & Business Media.

International Institute of Business Analysis. IIBA. (2015). A Guide to the Business Analysis Body of Knowledge (Babok Guide). Retrieved from http://www.the-aba.com/administrator/components/com_event/uploads/59014e456ca677.92343092BABOK_Guide_v3_Member.pdf

International Standard Organization ISO/IEC/IEEE 24764. (2010). Systems and software engineering - Vocabulary. Retrieved from https://www.iso.org/standard/50518.html

International Standard Organization ISO/IEC/IEEE 29148. (2011). Systems and software engineering - Life cycle processes - Requirements engineering. Retrieved from https://edisciplinas.usp.br/pluginfile.php/1077344/mod_folder/content/0/iso-iec-ieee-29148-2011.pdf

Joulin, A., Grave, E., Bojanowski, P., & Mikolov, T. (2016). Bag of Tricks for Efficient Text Classification. Retrieved from https://www.aclweb.org/anthology/E17-2068.pdf

Kamarudin, N. J., MOHD SANI, N. F., & Atan, R. (2015). Automated Transformation Approach From User Requirement To Behavior Design. *Journal of Theoretical & Applied Information Technology*, *81*(1).

Kapetanios, E., Alshahrani, S., Angelopoulou, A., & Baldwin, M. (2018). What do we learn from word associations? Evaluating machine learning algorithms for the extraction of contextual word meaning in natural language processing. *Preprints 2018*, 2018050102. doi: 10.20944/preprints201805.0102.v2

Kim, M., Park, S., Sugumaran, V., & Yang, H. (2007). Managing requirements conflicts in software product lines: A goal and scenario based approach. *Data & Knowledge Engineering*, *61*(3), 417-432.

Kiros, R., Zhu, Y., Salakhutdinov, R. R., Zemel, R., Urtasun, R., Torralba, A., & Fidler, S. (2015). Skip-thought vectors. Paper presented at the 29th Conference on neural information processing systems (NIPS 2015).

Kiyavitskaya, N., Zeni, N., Mich, L., & Berry, D. M. (2008). Requirements for tools for ambiguity identification and measurement in natural language requirements specifications. *Requirements engineering*, *13*(3), 207-239.

Knight, K., Nenkova, A., & Rambow, O. (2016). Hierarchical Attention Networks for Different Types of Documents with Smaller Size of Datasets. Paper presented at the 15th Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (ICOIN 2018).

Ko, Y., Park, S., Seo, J., & Choi, S. (2007). Using classification techniques for informal requirements in the requirements analysis-supporting system. *Information and Software Technology*, *49*(11-12), 1128-1140.

Kof, L., & Penzenstadler, B. (2011). Faster from requirements documents to system models: Interactive semi-automatic translation. Paper presented at the 1st International Requirements Engineering Efficiency Workshop (REEW 2011).

Kop, C., Fliedl, G., & Mayr, H. C. (2010). From Natural Language Requirements to a Conceptual Model. Paper presented at the 1st International Workshop on Evolution Support for Model-Based Development and Testing (EMDT2010).

Koshy, S., & Padmajavalli, R. (2018). Evaluating Techniques for Pre-Processing of Unstructured Text For Text Classification. Retrieved from https://www.ijcseonline.org/pub_paper/26-IJCSE-04494.pdf

Kowsari, K., Brown, D. E., Heidarysafa, M., Meimandi, K. J., Gerber, M. S., & Barnes, L. E. (2017). Hdltex: Hierarchical deep learning for text classification. Paper presented at the 16th IEEE International Conference on Machine Learning and Applications (ICMLA 2017).

Kruchten, P. (2004). The rational unified process: an introduction. Addison-Wesley Professional. Boston, MA: Addison-Wesley Longman Publishing Co., Inc.

Kusner, M., Sun, Y., Kolkin, N., & Weinberger, K. (2015). From word embeddings to document distances. Paper presented at the 32nd International conference on machine learning (ICML 2015).

Landauer, T. K., Foltz, P. W., & Laham, D. (1998). An introduction to latent semantic analysis. *Discourse processes*, *25*(2-3), 259-284.

Landhäußer, M., Körner, S. J., & Tichy, W. F. (2014). From requirements to UML models and back: how automatic processing of text can support requirements engineering. *Software Quality Journal*, *22*(1), 121-149.

Lau, J.H., & Baldwin, T. (2016). An Empirical Evaluation of doc2vec with Practical Insights into Document Embedding Generation. Paper presented at the 1st Workshop on Representation Learning for NLP (Rep4LNP 2016).

Le, Q., & Mikolov, T. (2014). Distributed representations of sentences and documents. Paper presented at the 31st International conference on machine learning (ICML 2014).

Lenc, L., & Král, P. (2016). Deep neural networks for Czech multi-label document classification. Paper presented at the 17th International Conference on Intelligent Text Processing and Computational Linguistics (CICLing 2016).

Leopold, H., Eid-Sabbagh, R. H., Mendling, J., Azevedo, L. G., & Baiao, F. A. (2013). Detection of naming convention violations in process models for different languages. *Decision Support Systems, 56*, 310-325.

Leopold, H., Mendling, J., & Polyvyanyy, A. (2014). Supporting process model validation through natural language generation. *IEEE Transactions on Software Engineering, 40(8)*, 818-840.

Lewis, D. D. (1998). Naive (Bayes) at forty: The independence assumption in information retrieval. Paper presented at the 10th European conference on machine learning (ECML 1998).

Li, Y., Wei, B., Liu, Y., Yao, L., Chen, H., Yu, J., & Zhu, W. (2018). Incorporating knowledge into neural network for text representation. *Expert Systems with Applications, 96*, 103-114.

Lin, J., Lin, C. C., Cleland-Huang, J., Settimi, R., Amaya, J., Bedford, G., ... Zou, X. (2006, September). Poirot: A distributed tool supporting enterprise-wide automated traceability. Paper presented at the 14th IEEE International Requirements Engineering Conference (RE 2006).

Liu, D., Li, Y., & Thomas, M. A. (2017). A roadmap for natural language processing research in information systems. Paper presented at the 50th Hawaii International Conference on System Sciences (HICSS 2017).

Lucia, D. (2000). Information retrieval models for recovering traceability links between code and documentation. Paper presented at the 2000 International Conference on Software Maintenance (ISCM 2000).

Maalej, W., Kurtanović, Z., Nabil, H., & Stanik, C. (2016). On the automatic classification of app reviews. *Requirements Engineering, 21*(3), 311-331.

MacDonell, S. G., Min, K., & Connor, A. M. (2005). Autonomous requirements specification processing using natural language processing. Retrieved from https://core.ac.uk/download/pdf/56362163.pdf

Manning, C., Raghavan, P., & Schütze, H. (2010). Introduction to Information Retrieval. Retrieved from https://nlp.stanford.edu/IR-book/pdf/irbookonlinereading.pdf

Marcus, A., & Maletic, J. I. (2003). Recovering documentation-to-source-code traceability links using latent semantic indexing. Paper presented at the 25th international conference on software engineering (ICSE 2003).

Marcus, A., Maletic, J. I., & Sergeyev, A. (2005). Recovery of traceability links between software documentation and source code. *International Journal of Software Engineering and Knowledge Engineering*, *15*(05), 811-836.

McMillan, C., Poshyvanyk, D., & Revelle, M. (2009). Combining textual and structural analysis of software artifacts for traceability link recovery. Paper presented at the 5th International Workshop on Traceability in Emerging Forms of Software Engineering (TEFSE 2009).

Meziane, F., Athanasakis, N., & Ananiadou, S. (2008). Generating natural language specifications from UML class diagrams. *Requirements Engineering*, *13*(1), 1- 18.

Mikolov, T., Chen, K., Corrado, G.S., & Dean, J. (2013a). Efficient Estimation of Word Representations in Vector Space. Paper presented at the Workshop at the International Conference on Learning Representations (ICLR 2013).

Mikolov, T., Joulin, A., Chopra, S., Mathieu, M., & Ranzato, M. A. (2015). Learning longer memory in recurrent neural networks. Paper presented at the Workshop at the International Conference on Learning Representations (ICLR 2015).

Mikolov, T., Karafiát, M., Burget, L., Černocký, J., & Khudanpur, S. (2010). Recurrent neural network based language model. Paper presented at the 11th annual conference of the international speech communication association (INTERSPEECH 2010).

Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S., & Dean, J. (2013b). Distributed representations of words and phrases and their compositionality. Paper presented at the Advances in neural information processing systems (NIPS 2013).

Mimno, D., Wallach, H. M., Talley, E., Leenders, M. & McCallum, A. (2011). Optimizing semantic coherence in topic models. Paper presented at the conference on empirical methods in natural language processing (EMNLP 2011).

Montes, A., Pacheco, H., Estrada, H., & Pastor, O. (2008). Conceptual model generation from requirements model: A natural language processing approach. Paper presented at the 13th International Conference on Application of Natural Language to Information Systems (NLDB 2008).

More, P., & Phalnikar, R. (2012). Generating UML diagrams from natural language specifications. International Journal of Applied Information Systems, Foundation of Computer Science, 1(8), 19-23.

Morin, F., & Bengio, Y. (2005). Hierarchical probabilistic neural network language model. Rertrieved from https://www.iro.umontreal.ca/~lisa/pointeurs/hierarchical-nnlm-aistats05.pdf

Mou, L., Li, G., Zhang, L., Wang, T., & Jin, Z. (2016). Convolutional neural networks over tree structures for programming language processing. Paper presented at the 30th Association for the Advancement of Aritifical Intelligence Conference on Artificial Intelligence (AAAI 2016).

Newman, D., Lau, J. H., Grieser, K., & Baldwin, T. (2010). Automatic evaluation of topic coherence. Paper presented at the Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics (HLT 2010).

Nicolás, J., & Toval, A. (2009). On the generation of requirements specifications from software engineering models: A systematic literature review. *Information and Software Technology*, *51*(9), 1291-1307.

Nikora, A., Hayes, J., & Holbrook, E. (2010). Experiments in automated identification of ambiguous natural-language requirements. Paper presented at the 21st IEEE International Symposium on Software Reliability Engineering (ISSRE 2010).

Niyogi, M., & Pal, A. K. (2019). Discovering conversational topics and emotions associated with demonetization tweets in India. *Computational Intelligence: Theories, Applications and Future Directions*. doi:https://doi.org/10.1007/978-981-13-1132-1_17

Nowakowski, W., Smialek, M., Ambroziewicz, A., & Straszak, T. (2013). Requirements-level language and tools for capturing software system essence. *Comput. Sci. Inf. Syst.*, *10*(4), 1499- 524.

Oliveto, R., Gethers, M., Poshyvanyk, D., & De Lucia, A. (2010). On the equivalence of information retrieval methods for automated traceability link recovery. Paper presented at the 18th International Conference on Program Comprehension (ICPC 2010).

Ororbia II, A., Giles, C. L., & Reitter, D. (2015). Learning a deep hybrid model for semi-supervised text classification. Paper presented at the Conference on Empirical Methods in Natural Language Processing (EMNLP 2015).

Pan, B., Yang, Y., Li, H., Zhao, Z., Zhuang, Y., Cai, D., & He, X. (2018). Macnet: Transferring knowledge from machine comprehension to sequence-to-sequence models. Paper presented at the Advances in Neural Information Processing Systems (NIPS 2018).

Pennington, J., Socher, R., & Manning, C. (2014). Glove: Global vectors for word representation. Paper presented at the conference on empirical methods in natural language processing (EMNLP 2014).

Pierce, R. A. (1978). A requirements tracing tool. *ACM SIGMETRICS Performance Evaluation Review*, *7*(3-4), 53-60.

Pinheiro, F. A. (2004). Requirements traceability. *In Perspectives on software requirements* (pp. 91–113). Boston, MA: Springer.

Ponte, J. M., & Croft, W. B. (1998). A language modeling approach to information retrieval. Retrieved from https://www.iro.umontreal.ca/~nie/IFT6255/ponte-croft.pdf

Poshyvanyk, D., Gueheneuc, Y. G., Marcus, A., Antoniol, G., & Rajlich, V. (2007). Feature location using probabilistic ranking of methods based on execution scenarios and information retrieval. *IEEE Transactions on Software Engineering*, *33*(6), 420-432.

Ramesh, B. (1998). Factors influencing requirements traceability practice. *Communications of the ACM*, *41*(12), 37-44.

Redington, M., Crater, N., & Finch, S. (1998). Distributional information: A powerful cue for acquiring syntactic categories. *Cognitive science*, *22*(4), 425-469.

Rehurek, R., & Sojka, P. (2010). Software framework for topic modelling with large corpora. Paper presented at the Workshop on New Challenges for NLP Frameworks (LREC 2010).

Robeer, M., Lucassen, G., van der Werf, J. M. E., Dalpiaz, F., & Brinkkemper, S. (2016). Automated extraction of conceptual models from user stories via NLP. Paper presented at the 24th International Requirements Engineering Conference (RE 2016).

Robertson, S. E., & Jones, K. S. (1976). Relevance weighting of search terms. *Journal of the American Society for Information science*, *27*(3), 129-146.

Robertson, S., & Zaragoza, H. (2009). The probabilistic relevance framework: BM25 and beyond. *Foundations and Trends® in Information Retrieval*, *3*(4), 333-389.

Rosadini, B., Ferrari, A., Gori, G., Fantechi, A., Gnesi, S., Trotta, I., & Bacherini, S. (2017). Using NLP to detect requirements defects: An industrial experience in the railway domain. Paper presented at the 23rd International Working Conference on Requirements Engineering: Foundation for Software Quality (REFSQ 2017).

Ryan, K. (1993). The role of natural language in requirements engineering. Paper presented at the 1st IEEE International Symposium on Requirements Engineering (RE 1993).

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

Settimi, R., Cleland-Huang, J., Khadra, O. B., Mody, J., Lukasik, W., & DePalma, C. (2004). Supporting software evolution through dynamically retrieving traces to UML artifacts. Paper presented at the 7th International Workshop on Principles of Software Evolution (IWPSE 2004).

Shah, U. S., & Jinwala, D. C. (2015). Resolving ambiguities in natural language software requirements: a comprehensive survey. *ACM SIGSOFT Software Engineering Notes, 40*(5), 1-7.

Shokripour, R., Anvik, J., Kasirun, Z. M., & Zamani, S. (2015). A time-based approach to automatic bug report assignment. *Journal of Systems and Software, 102*, 109-122.

Sidorov, G., Gelbukh, A., Gómez-Adorno, H., & Pinto, D. (2014). Soft similarity and soft cosine measure: Similarity of features in vector space model. *Computación y Sistemas, 18*(3), 491-504.

Sievert, C., & Shirley, K. (2014). LDAvis: A method for visualizing and interpreting topics. Paper presented at the workshop on interactive language learning, visualization, and interfaces (ACL 2014).

Singhal, A. (2001). Modern information retrieval: A brief overview. *IEEE Data Engineering. Bull., 24*(4), 35-43.

Sinha, A., Paradkar, A., Kumanan, P., & Boguraev, B. (2008). An analysis engine for dependable elicitation on natural language use case description and its application to industrial use cases. Retrieved from https://pdfs.semanticscholar.org/e004/4a179749f5784ab126d2334bf65fb5c0c43b.pdf

Sinpang, J. S., Sulaiman, S., & Idris, N. (2017). Detecting Ambiguity in Requirements Analysis Using Mamdani Fuzzy Inference. *Journal of Telecommunication, Electronic and Computer Engineering (JTEC), 9*(3-4), 157-162.

Sohrabi, B., Vanani, I. R., & Shineh, M. B. (2018). Topic Modeling and Classification of Cyberspace Papers Using Text Mining. *Journal of Cyberspace Studies, 2*(1), 103-125.

Spanoudakis, G., & Zisman, A. (2005). Software traceability: a roadmap. *Handbook Of Software Engineering And Knowledge Engineering: Recent Advances*. doi: https://doi.org/10.1142/9789812775245_0014

Stevens, K., Kegelmeyer, P., Andrzejewski, D., & Buttler, D. (2012). Exploring topic coherence over many models and many topics. Paper presented at the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL 2012).

Sultan, M. A., Bethard, S., & Sumner, T. (2015). Dls@ cu: Sentence similarity from word alignment and semantic vector composition. Paper presented at the 9th International Workshop on Semantic Evaluation (SemEval 2015).

Sultanov, H., & Hayes, J. H. (2010). Application of swarm techniques to requirements engineering: Requirements tracing. Paper presented at the 18th IEEE International Requirements Engineering Conference (RE 2010).

Thayasivam, U., Verma, K., Kass, A., & Vasquez, R. G. (2011). Automatically Mapping Natural Language Requirements to Domain-Specific Process Models. Paper presented at the 23rd Innovative Applications of Artificial Intelligence Conference (IAAI 2011).

The Institute of Electrical and Electronics Engineers ANSI/IEEE. (1993). Guide to Software Requirements Specification. Retrieved from https://www.utdallas.edu/~chung/RE/IEEE830-1993.pdf

Tichy, W. F., & Koerner, S. J. (2010). Text to software: developing tools to close the gaps in software engineering. Paper presented at the FSE/SDP workshop on Future of software engineering research (FoSER 2010).

Tjong, S. F., & Berry, D. M. (2013). The design of SREE—a prototype potential ambiguity finder for requirements specifications and lessons learned. Paper presented at the 19th International Working Conference on Requirements Engineering: Foundation for Software Quality (REFSQ 2013).

Tjong, S. F., Hartley, M., & Berry, D. M. (2007). Extended Disambiguation Rules for Requirements Specifications. Paper presented at the 10th International Workshop on Requirements Engineering (RE 2007).

Tsumaki, T., & Morisawa, Y. (2000). A framework of requirements tracing using UML. Paper presented at the 7th Asia-Pacific Software Engineering Conference (APSEC 2000).

Turtle, H., & Croft, W. B. (1991). Evaluation of an inference network-based retrieval model. *ACM Transactions on Information Systems (TOIS)*, *9*(3), 187-222.

Velasco, J. L., Valencia-García, R., Fernández-Breis, J. T., & Toval, A. (2009). Modelling reusable security requirements based on an ontology framework. *Journal of Research and Practice in Information Technology*, *41*(2), 119.

Vlas, R. E., & Robinson, W. N. (2012). Two rule-based natural language strategies for requirements discovery and classification in open source software development projects. *Journal of Management Information Systems*, *28*(4), 11-38.

Vodrahalli, K. (2015). Deep Learning for NLP. Retrieved from http://3dvision.princeton.edu/courses/COS598/2015sp/slides/NLP/nlp_deep_learning.pdf

Wallach, H. M., Murray, I., Salakhutdinov, R. & Mimno, D. (2009). Evaluation methods for topic models. Paper presented at the 26th annual international conference on machine learning (ICML 2009).

Wei, X., & Croft, W. B. (2006). LDA-based document models for ad-hoc retrieval. Paper presented at the 29th annual international conference on Research and development in information retrieval (ACM SIGIR 2006).

Winkler, J. P., & Vogelsang, A. (2017). "What Does My Classifier Learn?" A Visual Approach to Understanding Natural Language Text Classifiers. Paper presented at the 23rd International Conference on Applications of Natural Language to Information Systems (NLDB 2018).

Winkler, S., & von Pilgrim, J. (2010). A survey of traceability in requirements engineering and model-driven development. *Software & Systems Modeling*, *9*(4), 529-565.

Yue, T., Ali, S., & Zhang, M. (2015). RTCM: a natural language based, automated, and practical test case generation framework. Paper presented at the International Symposium on Software Testing and Analysis (ISSTA 2015).

Zamani, S., Lee, S. P., Shokripour, R., & Anvik, J. (2014). A noun-based approach to feature location using time-aware term-weighting. *Information and Software Technology*, *56*(8), 991-1011.

Zeng, Y. (2008). Recursive object model (ROM)—Modelling of linguistic information in engineering design. *Computers in Industry, 59*(6), 612-625.

Zhai, C. (2007). A brief review of information retrieval models. Retrieved from http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.94.5325&rep=rep1&type=pdf

Zhai, C. (2008). Statistical language models for information retrieval. *Synthesis Lectures on Human Language Technologies, 1*(1), 1-141.

Zhang, H., Li, J., Zhu, L., Jeffery, R., Liu, Y., Wang, Q., & Li, M. (2014). Investigating dependencies in software requirements for change propagation analysis. *Information and Software Technology, 56*(1), 40-53.

Zhang, T., Huang, M., & Zhao, L. (2018). Learning structured representation for text classification via reinforcement learning. Paper presented at the 32nd AAAI Conference on Artificial Intelligence (AAAI 2018).

Zhong, H., Zhang, L., Xie, T., & Mei, H. (2011). Inferring specifications for resources from natural language API documentation. *Automated Software Engineering, 18*(3-4), 227-261.

Zou, X., Settimi, R., & Cleland-Huang, J. (2006). Phrasing in dynamic requirements trace retrieva. Paper presented at the 30th Annual International Computer Software and Applications Conference (COMPSAC 2006).