Réput lique Algérienne Democratique et Populaire
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique

Université Saad Dahlab Blida

N° D'ordre :.....

Faculté des sciences

Département d'informatique

Mémoire Présenté par :

Nom : Krebbaza prénom : Zineb Nom : Medjamia prénom : Anes

En vue d'obtenir le diplôme de master

Domaine : Mathématique et informatique

Filière : Informatique
Spécialité : Informatique
Option : Ingénierie de logiciel

Thème : Vers un meilleur résumé automatique multi-document

Soutenu le :

M. Président
M. Examinateur
M. Examinateur
M. Promoteur: Kourtiche Abdallah Hicham

Promotion 2015 / 2016

Remerciements

Avant tout nous remercions Dieu le tout puissant de nous avoir donné le courage et de nous avoir guidé vers le droit chemin, de nous avoir aidé tout au long de nos années d'études.

Nous adressons notre profond remerciement à M. Abdallah Hicham Kameche notre promoteur qui nous a aidé et orienté avec ses corrections.

Dédicaces

A mes chers parents qui m'ont guidé durant les moments les plus pénibles de ce long chemin, ma mère qui a été à mes côtés et m'a soutenu durant toute ma vie, et mon père qui a sacrifié toute sa vie afin de me voir devenir ce que je suis, merci mes parents je vous aime beaucoup.

A mes chères frères: Mohamed et Yassine pour leur soutien morale et leurs sacrifices le long de mes études. Avec toute ma tendresse.

A toute ma famille, mes oncles et leurs épouses, cousins et à leurs épouses Hafsa, Fella et Zahra, à toutes mes cousines en particulier Fatima et Meriem.

A mon binôme: Anes qui m'a accompagné tout au long de cette année.

A mes meilleures amies: Djazia, Wissem, Fella et Maria.

Enfin je dédie à tous mes amis que j'ai n'ai pas cités, à tous ceux qui me connaissent.

Je dédie ce modeste travail

Zineb

Dédicaces

Tout d'abord, Je remercie Allah de m'avoir accordé la patience, le courage et la volonté durant toutes mes années.

Je dédie ce modeste travail A mes parents mes frères ma petite sœur ma grande mère bien aimé et tous les membres de ma famille qui m'ont soutenu. A mes meilleurs amis Ayoub, Amine, Adlane, Yassine, Hassane, Toufik et Habib. A mes amis de notre section informatique de la promotion 2015/2016. A mes enseignants qui nous ont beaucoup appris dans ces deux dernières années. Que Dieu les garde.

Anes

Résumé

La forte augmentation de texte disponible en format numérique a fait ressortir la nécessité de concevoir et de développer des outils de résumé performants dans le but de repérer et extraire l'information pertinente sous une forme abrégée. Un résumé est un texte reformulé dans un espace plus réduit. Il doit exprimer avec un minimum de mots le contenu essentiel d'un document. Le but d'un système de résumé automatique est de produire une représentation condensée d'une source d'information, dans laquelle les informations « importantes » du contenu original sont préservées. Les sources d'information pouvant être résumées sont nombreuses et hétérogènes : documents vidéos, sonores ou textuels. Notre étude portera uniquement sur le résumé extractif multi-document.

Notre idée est de choisir les phrases qui peuvent représenter le texte d'entrée au totalité. Puisque le texte d'entrée se compose de plusieurs thèmes, une phrase du résumé est supposée représenter tous ou la majorité de ses thèmes, prenant en considération l'importance de chaque thème. Pour cela nous avons utilisé l'approche de Clustering pour regrouper les phrases similaires dans des clusters.

Mots clés: Résumé multi-document, résumé extractif, Clustering

Abstract

The increase in availability of text in digital format accentuates the need for design and development of efficient summarizer tools to track and extract relevant information in a shortened form. A summary is a text rephrased in a smaller space. It should express the essential content of a document with a minimum of words. The purpose of a summarization system is to produce a condensed representation of a source of information, in which the "significant" information of the original content is preserved. The sources of information that can be summarized are many and diverse: video documents, audio or text. Our study will focus only on the extractive multi-document summary.

Our idea is to select the sentences which can represent the whole input text. Because the input text is composed of many topics, a sentence of the summary is supposed to represent all or the majority of these topics, considering each topic's importance. For this we used the approach of Clustering to create similar sentences' clustes.

Keywords: Multi-document summarization, extractive summarization, Clustering

Table des matières

Listes des figures

Introduction générale

Avec l'émergence du World Wide Web (WWW) et des services en ligne, une quantité croissante d'informations devient disponible et accessible. L'explosion de l'information disponible a engendré un problème bien connu qui est la surcharge d'information. Nous ne disposons pas d'assez de temps et d'énergie pour toute lire. Une prise de décision doit alors se faire en connaissance du fait que l'intégralité de l'information disponible n'a pu être traitée. C'est pour résoudre cette problématique que le résumé de texte est indispensable.

Résumer consiste à condenser l'information la plus importante provenant d'un document (ou de plusieurs documents) afin d'en produire une version abrégée pour un utilisateur (ou plusieurs utilisateurs). Il existe de nombreuses applications du résumé présentes dans la vie de tous les jours avec lesquelles nous nous sommes familiarisés.

Notre projet a pour but d'extraire les parties les plus intéressantes d'un ensemble de documents afin de présenter à l'utilisateur les informations importantes qui lui semblent correspondre au mieux à ses besoins ainsi qu'au thème qui se trouvent dans le contenu des documents.

Pour cela nous examinons les différentes approches existantes de classification et de segmentation documentaires en mettant l'accent sur les travaux les plus récents.
Afin de réaliser notre projet nous avons divisé notre travail en quatre chapitres :

- Chapitre 1 : Résumé automatique de texte ; où nous allons discuter le résumé automatique de façon globale en citant ses différents types. Ensuite, nous allons présenter les étapes du résumé automatique en détail. Enfin, nous allons présenter quelques méthodes d'évaluation telles que la méthode ROUGE, Pyramides, etc.
- Chapitre 2 : Classification automatique de textes ; où nous allons définir la classification automatique de texte avec ses deux types. Nous allons présenter les techniques utilisées dans la classification non supervisée (Clustering) qui est l'approche de notre méthode cible, il s'agit de discussions sur les méthodes non supervisé en examinant pour chacune (ses point fort, ses faiblesses, son principe… etc.).
- Chapitre 3 : Conception ; nous allons présenter l'architecture générale de notre système de résumé multi-documents pour ensuite expliquer les différentes étapes qu'on a suivies.
- Chapitre 4 : Réalisation ; il est consacré à l'évaluation de nos deux approches utilisées, pour enfin présenter et discuter les résultats de nos expérimentations.

Chapitre 1

Résumé automatique de texte

1.1 Introduction

Le résumé automatique de documents n'est pas un nouveau domaine. Il a connu un fort renouveau ces dernières années et les recherches en ce domaine ont fortement évolué récemment. Dès les années 50, les chercheurs font leurs études sur le développement d'un outil permettant de résumer les documents, en utilisant les différentes méthodes pour améliorer ce résumé. Dans ce fait, plusieurs approches sont apparait pour le résumé automatique de documents.

Dans ce chapitre, nous allons présenter le résumé automatique de façon générale en donnant premièrement quelques définitions pour le résumé de manière classique ensuite, on revient sur le résumé automatique pour voir ses différents types et étapes. Enfin nous allons citer quelques applications du résumé automatique de textes.

1.2 Définitions

Avant de parler de résumé automatique, on peut peut-être déjà identifier ce qu'est le résumé de manière classique. Au départ, le résumé, c'est quand un résumeur professionnel - donc un humain - prend un texte et en dégage les idées essentielles pour en faire un texte plus court. Le résumé est donc un autre texte, plus court et censé dégager les idées saillantes qui étaient présentes dans le texte initial.

Le résumé automatique, c'est de faire faire par une machine la tâche auparavant faite par un résumeur humain. À partir d'un texte source qui est évidemment numérisé, il s'agit donc de produire un texte plus court. Voilà comment parler du résumé automatique.

Plusieurs définitions du résumé de texte peuvent être trouvées dans la littérature. Une définition concise a été donnée en 1979 par *l'American National Standards Institute* (ANSI) [1] : « Le résumé est la représentation abrégée et exacte du contenu d'un document, de préférence préparé par son auteur(s) pour être publié avec le document. Les résumés sont

utiles pour l'accès aux publications et dans les bases de données informatisées »Selon van Djik en 1980 **[2]** : « La fonction primaire des abstracts est d'indiquer et de prédire la structure et le contenu du texte ». D'après Horacio Saggion et Guy La palme **[3]**, fonctionnellement un résumé est définit comme : « Une version condensée d'un document source qui possède un genre et un propos spécifique pour donner au lecteur une idée exacte et concise du contenu de la source » .

Enfin selon l'article de Karen Spärck-Jones et Tetsuya Sakai apparu en 2001, le processus de génération du résumé automatique de documents peut être définit comme suit : « Un résumé est une transformation réductrice d'un texte source vers un résumé par extraction ou génération (du contenu) ».

1.3 Classification de résumé automatique

La classification du résumé automatique peut varier selon le critère choisi pour la faire. C'est pour cette raison, le résumé automatique n'appartient pas exclusivement à une classe donnée, et donc on peut trouver un résumé qui appartient à plusieurs classes.

Tout résumé peut être classifié par (au moins) trois classes majeures des facteurs de classification **[4]**. Selon **[5]** et **[4]**, on peut classifier ces facteurs en trois catégories : des facteurs sur document d'entrée, des facteurs sur l'objectif, et des facteurs sur le document de sortie.

1.3.1 Le document d'entrée

Les documents utilisés comme entrée d'un système de résumé automatique, peuvent prendre plusieurs formes. Ils peuvent aussi être spécialisés pour un domaine, comme ils peuvent appartenir à des domaines arbitraires. Si ces documents ont le même thème, on peut les prendre tous comme entrée, alors un système de résumé peut être mono-document ou multi-document. Donc, on peut déduire qu'il existe trois façons de classifier le résumé en fonction du document d'entrée : selon sa forme, selon sa spécialité, et selon l'unité.

1.3.1.1 La forme

La forme contient plusieurs aspects : la structure, l'échelle, le médium, et le genre. La structure comprend l'organisation explicite comme le marquage par des entêtes (ex. l'objectif, data, méthode, etc. Il est plus approprié ou non de préserver la structure de document dans le

résumé. En prenant les articles scientifiques comme un exemple, la structure commune dans ce type de document contient les composants : l'introduction, les travaux reliés, le travail réalisé, l'évaluation, et la conclusion. Si on ne veut pas préserver la structure de ce document, on peut simplement faire un résumé sur son contenu entier. Sinon, le résumé va avoir comme contenu, le résumé de chaque composant du document d'entrée. Le travail de **[6]** est un exemple des travaux qui prennent en considération la structure du document. Il utilise comme documents d'entrée, les papiers de recherche très structurées en examinant un corpus des articles pour en extraire les différents patterns des structures existantes.

Du fait qu'on peut résumer un paragraphe comme on peut résumer un livre, l'échelle est importante puisqu'elle contient l'implication pas seulement pour le degré de réduction de résumé, mais aussi pour l'ampleur possible de la transformation du contenu. Un des exemples sur la petite échelle est le travail de **[7]** dont l'objectif était de résumer les histoires courtes afin d'aider le lecteur à déterminer s'il est intéressé ou non par l'histoire, en fournissant une idée sur les paramètres de cette histoire (comme les caractères, heure, lieu, etc.) sans donner de détails.

Le médium est le support du document d'entrée. En regardant aux recherches sur le résumé automatique, on trouve que le support le plus utilisé est le texte, bien qu'il existe d'autres supports comme l'image, le son, la vidéo, etc. Ces supports contiennent des informations qu'on peut extraire et résumer de même principe que celui du document texte. Parmi les travaux qui concernent le résumé d'un document non textuelle, on peut citer : le résumé des enregistrements audio **[8]** et **[9]**, résumé des vidéos comme dans **[10]** et **[11]** qui ont comme but de résumer des vidéo concernant le football, ou des images comme dans le travail de **[12]** et **[13]**.

Différentes techniques de résumés peuvent être appliquées sur différents genres selon différents échelles, et pas sur d'autres. Ces genres peuvent âtre des articles de journaux, des pièces d'opinion, des romans, des histoires, des rapports, etc. Un exemple est celui de **[14]** qui prend comme entrée les documents de genre pièces d'opinion (en Anglais : *reviews*).

1.3.1.2 La spécialité

Le résumé automatique peut être général ou spécifique à un domaine précis. Lorsque les documents d'entrée ont tous le même domaine, il est plus approprié d'utiliser un système de résumé automatique spécialisé pour ce domaine. Donc, ce type de résumé peut diminuer le

degré d'ambiguïté des termes, les mots particuliers et usage de grammaire, et le formatage spécialisé. Un des exemples sur un résumé de domaine spécifique est le travail de **[15]** qui s'intéresse aux textes biomédicaux pour aider les médecins à lire les informations concernant les tests cliniques des patients.

D'autre part, un résumé de domaine général est dérivé d'un ou plusieurs documents d'entrée dans n'importe quel domaine. Un exemple de résumé de domaine général est donné par le travail de **[16]** en exploitant la diversité des concepts dans le document d'entrée.

1.3.1.3 L'unité

Selon le nombre de sources, on peut classifier le résumé automatique de documents en deux catégories : Mono-document et multi-documents. Un résumé mono-document est un résumé qui traite un seul document en entrée (même si le processus utilise des données d'autres documents). Il existe plusieurs recherches sur le résumé automatique mono-document, commençant par le travail de Luhn **[17]**. Parmi ces travaux, on peut prendre comme exemple : **[18]**, **[19]**, **[20]**, **[4]**, [21], etc.

D'autre part, un résumé multi-documents est un résumé qui couvre le contenu de plusieurs documents en entrée, et qu'il est utilisé seulement si ces documents sont reliés par un même thème. Il parait que ce type de résumé a été utilisé pour la première fois par **[22]**, en développant le système SUMMONS1. Un exemple sur le résumé multi-documents est le travail de **[23]** qui se considère comme le premier travail qui contient une évaluation d'une approche de résumé automatique multi-documents sur les textes en français.

1.3.2 Le but

Un système de résumé peut être générique où le résumé toujours procède de la même façon quelque-soit l'utilisateur, ou il peut être contrôlé par l'utilisateur. Il peut être destiné pour indiquer seulement une idée sur le document sans donner aucun contenu, comme il peut être destiné pour informer l'utilisateur sur le contenu du document, en fournissant une version courte. Il peut fournir un résumé détaillé ou juste fournir les nouvelles. Donc, selon le but de résumé, on obtient trois facteurs de classification : le public, la fonction, et la situation.

1.3.2.1 Le public

Selon le public, on peut classifier le résumé sur deux catégories : générique et orienté par requête (profil utilisé). Un résumé générique est un résumé qui fournit le point de vu de

l'auteur sur un ou plusieurs documents d'entrée, il ne tient pas compte de ce que l'utilisateur veut chercher. Il existe plusieurs exemples sur ce type de résumé, parmi eux on trouve **[24]**. L'utilisateur dans ce travail est supposé chercher sur le thème général de l'histoire, pour juger s'il veut la lire ou non. Si on utilise un résumé orienté par requête, donc l'utilisateur doit savoir préalablement ce qu'il cherche, et le but devient absurde. Un résumé orienté par requête (ou orienté utilisateur) préfère des thèmes spécifiques ou aspect(s) de document. Il peut être explicite, en sélectionnant les thèmes pertinents. Comme il peut être implicite, en négligeant les thèmes qui ne sont pas en relation avec les intérêts de l'utilisateur. Le travail de **[25]** contient une sorte de résumé par requête, lorsque l'utilisateur sélectionne un mot-clé un résumé s'affiche suivant le thème concernant ce mot-clé.

1.3.2.2 La fonction

Selon la fonction, le résumé peut être indicatif ou informatif. Le résumé indicatif fournit seulement une indication sur la matière du sujet principal ou le domaine de document(s) d'entrée sans introduire son contenu. La recherche des mots-clés et thème d'un document est considérée comme une sorte de résumé indicatif, le travail de **[25]** en est un exemple. Ce travail est destiné pour aider les utilisateurs à analyser une grande quantité de textes, en fournissant les mots-clés et les thèmes, présentés d'une manière graphique et distribués au long du temps.

Un résumé informatif reflet le contenu (ou une partie), et nous permet de décrire le contenu du document d'entrée. D'après **[26]**, "Un résumé informatif fournit une information sur tous les points importants du document en cherchant à couvrir tous les sujets traités parle document". Deux exemples sur le résumé informatif sont : **[23]** pour le résumé multi-documents et **[27]** pour le résumé mono-document.

1.3.2.3 La situation

Le résumé selon la situation peut être un résumé de fond (background) ou un résumé qui rapporte Seulement-les-nouvelles. Un résumé de fond suppose que la connaissance antérieure de lecteur sur le cadre général du contenu de document(s) d'entrée est très pauvre, et donc il inclut des matériels d'explication, comme la situation des places, le temps, et les acteurs. Le travail de **[24]** peut être considéré comme un résumé automatique de fond. Un résumé de type Seulement-les-nouvelles contient seulement les thèmes nouveaux ou principaux, en supposant que le lecteur a un fond suffisant pour les interpréter dans le contexte. Le travail de **[28]**

présente un système de résumé automatique qui détecte les nouveaux thèmes après une période de temps. Il est considéré comme un système de résumé multi documents.

1.3.3 Le document de sortie

Le résumé d'un document peut être obtenu de deux manières : la première est de prendre les phrases (ou des parties) qu'elles semblent plus probable de représenter ce document, la deuxième est de générer le résumé. Il peut refléter le document d'entrée sans introduire des jugements, comme il peut porter des jugements sur son contenu. Tandis que le format du résumé peut être fixe pour tous les documents d'entrée, comme il peut varier selon les types de ces documents.

Les trois critères de classification, en observant le document de sortie, sont : la dérivation, la partialité, et le format.

1.3.3.1Dérivation

Un résumé est soit une extraction ou une abstraction. Une extraction est une collection de passages (des mots jusqu'à des paragraphes) extraite d'un document d'entrée et produite textuellement comme un résumé. Les premières techniques d'extraction ont utilisé le calcul de score pour chaque phrase en fonction des critères tels que la fréquence de mots de phrase **[17]**, les phrases clés **[18]**...etc. Des méthodes d'extraction récentes utilisent d'autres techniques plus sophistiquées pour décider quelle phrase va être extraite. Ces techniques dépendent souvent sur l'apprentissage automatique sur l'analyse du langage naturel pour identifier les passages clés, ou sur les relations entre les mots plutôt que des sacs de mots **[29]**.

Une abstraction est un texte (document en général) nouvellement généré, produit de certaines représentations internes, et qui est un résultat de l'analyse de l'entrée. Le résumé par abstraction est difficile à concevoir, en le comparant avec le résumé par extraction, mais on peut le rendre moins difficile en utilisant quelques techniques. Parmi ces techniques, la conception d'une méthode de résumé destinée à un domaine spécifique des documents, cela rend le processus plus facile, comme il est indiqué par **[19]**. Pour éviter le problème de différenciation entre les concepts importants dans le texte entre autres, l'auteur utilise un domaine spécifique pour les documents d'entrée.

1.3.3.2 Partialité

Ce critère est principalement appliqué quand le matériel d'entrée est un sujet d'opinion ou pré-jugement. Un résumé selon la partialité peut être neutre ou évaluatif. Un résumé neutre reflète le contenu de document(s) d'entrée, soit partialement ou impartialement, sans introduire des critiques ou évaluations. La plupart des recherches dans le résumé automatique sont de ce type.

Un résumé évaluatif inclut quelques jugements propres au système, soit explicitement (en utilisant des déclarations d'opinion) ou implicitement (en incluant des matériels d'un pré jugement et l'omission des matériels avec un autre). Le travail de **[30]** représente un système de résumé automatique destiné pour les documents médicales, afin d'aider les médecins dans leurs décisions. Un autre exemple est celui présenté dans **[31]** qui traite les textes d'opinion par l'analyse de blogs, où sont exprimées à la fois des informations factuelles et des prises de position sur les faits considérés.

1.3.3.3 Le format

Un résumé, selon le format, peut être fixe ou flottant. Un résumé d'un format fixe est créé pour une utilisation, utilisateur (ou classe d'utilisateurs), ou situation spécifique. Comme tel, il peut se conformer aux conventions internes appropriées de soulignement, formatage, etc. Un résumé d'un format flottant est un résumé ayant un format varié ; il est créé et affiché avec des préférences variées, pour des différents utilisateurs, et des différents buts. Par exemple, pour un utilisateur donné, le système produit un résumé au format d'une table, et pour un autre utilisateur, pour le même document, il affiche un texte. **La figure 1.1** illustre les différents facteurs de classification.

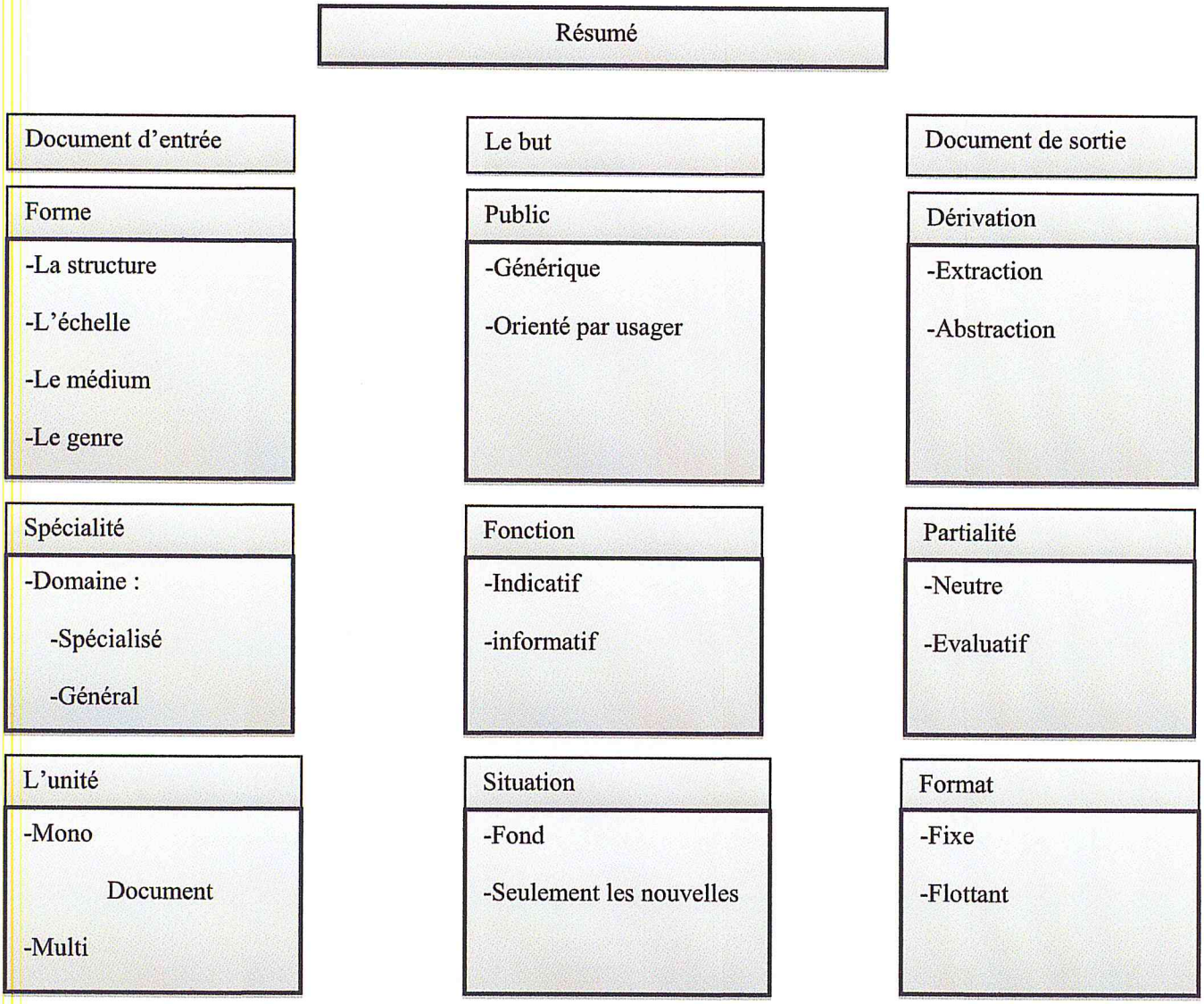


Figure 1.1 : Les différentes classes d'un résumé automatique

1.4 Les étapes de résumé automatique

Les documents en entrée du système de résumé automatique subissent un traitement préalable qui nous garde que l'information pertinente dont le système a besoin. Ainsi que les critères fondamentaux pour le calcul du score d'une unité du texte, en général la phrase. Les scores de ces critères vont être combinés pour avoir un seul score qui nous aide à juger la pertinence d'une unité, nous présentons ici les différentes étapes de résumé :

1.4.1 Pré-traitement :

Le Pré-traitement dans le résumé automatique de textes (ou la recherche d'information en général) est une tâche très importante, nous avons introduit les différentes étapes utilisées dans le pré-traitement, sont des techniques du TALN **[32]** qui servent à rendre le texte d'entrée plus facile à traiter et de le transformer à un standard qui est utilisé comme entrée au module de traitement.

1.4.1.1 Normalisation

La normalisation est la procédure qui transforme un document dans un format standard plus facile à manipuler. Par exemple, un document peut contenir les mots équivalents "Mr.", "Mr","mister", et "Mister", qui peuvent tous être normalisés en un seul mot.

Parmi les opérations effectuées dans cette étape :

- Suppression des caractères spéciaux et des chiffres.
- Remplacement de certains mots équivalents, comme les abréviations (voir l'exemple précédent).
- Dans certaines langues comme la langue Arabe, élimination des diacritiques.
- La suppression de certaines formes de style, comme celles des pages Web.

1.4.1.2 Segmentation

Dans le résumé par extraction, il est nécessaire de décider sur quelle granularité nos segments vont être extraits, c.-à-d. la taille de chaque unité extraite pour former le résumé. Bien que dans la plupart des cas, l'extraction se fait au niveau de phrase. Dans l'approche

statistique, les traitements sont exécutés sur les mots qui doivent être segmentés eux aussi, on appelle ça en anglais : Tokenization.

1.4.1.2.1 Segmentation du texte en phrases

Dans la plupart des langues écrites, les phrases sont délimitées par des marques de ponctuation comme le point, le point d'exclamation, le point d'interrogation. La phrase est définie par une clause qui commence par une lettre majuscule et se termine par un des trois marques de ponctuation précédentes **[33]**. Mais il existe des cas où cette définition ne peut pas être appliquée :

– Le point peut être utilisé dans les abréviations, par exemple "Mr.", "Dr.", "etc.", etc., qui se trouvent dans la plupart des cas au milieu d'une phrase.

– Les phrases peuvent être délimitées par plusieurs autres marques de ponctuation **[34].**

Il existe des cas où des phrases sont délimitées par des marques autres que les points, comme par exemple les virgules utilisées par les séquences d'actions.

– Dans des langues, comme le Thaï, il n'existe pas de ponctuation pour différencier les limites de phrases.

Plusieurs facteurs contextuels ont été proposés pour aider à la segmentation en phrases, comme **[34]** :

– Distinction de la casse : les mots commençant par une lettre majuscule donnent une information sur les limites de phrases. Les phrases commencent toujours par une lettre en majuscule.

– Parties du discours (Part of speech) : **[35]** ont prouvé que l'utilisation des parties de discours qui entourent une marque de ponctuation avec un algorithme d'apprentissage, peuvent aider à détecter les limites de phrases.

– Longueur du mot : La longueur de mots avant et après un point, est utilisée par **[36]**, comme un critère contextuel.

– Les suffixes : **[37]** utilisent l'analyse morphologique pour identifier les suffixes, et de ce fait filtrer les mots qui ne sont pas des abréviations. Ceci est utilisé pour identifier les mots qui ne figurent pas dans la liste des abréviations.

– Préfixes et suffixes : **[38]** utilisent les préfixes et les suffixes des mots entourant la marque de ponctuation, comme un critère contextuel.

– Les classes des abréviations : **[36]** et **[38]** divisent les abréviations sur des catégories comme les titres (qui ne peuvent pas être à la fin de phrase, comme Mr., Dr., etc.) et les indicatifs corporatifs (qui peuvent être à la fin de phrase, comme Corp., S.p.A., etc.).

– Les noms propres : **[39]** utilise la présence des noms propres à droite du point comme critère.

Pour détecter les limites de phrases, il existe deux approches : l'approche basée sur les règles manuelles, et l'approche par apprentissage. La première approche est la plus ancienne et la plus utilisée **[34]** parmi ces deux approches, pour déterminer les limites de phrases. Elle utilise une grammaire régulière définie manuellement, avec une liste des abréviations et des noms propres. Un des algorithmes basés sur les expressions régulières, est le système d'extraction de l'information Alembic développé par **[40]**. Il contient des différents modules qui tentent de classifier toute sorte de marque de ponctuation en identifiant le point dans les nombres, les expressions de date et de temps, et les abréviations. Il utilise une liste de 75 abréviations et une série de plus de 100 règles manuelles.

Cette approche est développée pour un corpus et une langue spécifique et se base sur les listes de mots spécifiques pour une langue. Ainsi, pour l'appliquer sur une autre langue il faut les redéfinir à nouveau. La première approche requiert un grand effort pour écrire les règles de détection les délimiteurs et la préparation de la liste des abréviations.

Les méthodes par apprentissage tentent de définir ces règles de manière automatique, ceci permettra de varier les langues, les applications, et les genres. Le premier travail utilisant cette approche pour la segmentation en phrases, est celui de **[36]**. Sa méthode utilise les arbres de régression **[41]** pour classifier les points selon des critères contextuels concernant les mots qui précèdent et qui suivent les points. Ces critères contextuels comportent : la longueur du mot, la ponctuation après le point, les classes d'abréviation, la casse du mot, et la probabilité que le mot ayant place au début ou à la fin de la phrase. Les algorithmes basant sur l'approche par apprentissage, sont nécessaires pour fournir des traitements robustes sur une variété de textes et de langues.

1.4.1.2.2 Segmentation du texte en mots (Tokenization)

La segmentation de mots est une étape nécessaire pour exécuter les traitements statistiques. Dans la plupart de systèmes d'écriture, les mots sont séparés par des espaces. Un simple algorithme de segmentation de mots considère tous les caractères consécutifs précédés et suivis par un espace, comme un mot. Cette convention est sujet de plusieurs problèmes :

– Les marques de ponctuation sont considérées comme un terme séparé. Mais dans le cas du point, on peut trouver qu'il peut être attaché à un autre terme, comme les abréviations qui se terminent toujours par un point.

– Les marques de citation (“”‘’) sont utilisées pour spécifier le début et la fin d'une citation. les guillemets de citation et l'apostrophe ont le même caractère, et donc on ne peut pas déterminer immédiatement si la marque est un guillemet ou une apostrophe.

–L'apostrophe est une source d'ambiguïté dans la segmentation de mots. Dans l'anglais, l'apostrophe peut être utilisée avec un "*s*" dans la contraction du verbe "*is*" (*she's,* etc.). Dans l'anglais, la contractions "*I'm*" est segmenté comme "*I am*". Pour le français, on peut citer un ensemble de contractions, y compris : la contraction des articles (l'homme, c'était), et autres formes (n'y, qu'ils, d'ailleurs).

– Certaines langues contiennent des mots composés, soit en attachant un mot à l'autre, soit par trait d'union. Dans l'allemand, il est commun d'utiliser la composition des mots : nom-nom (*Lebensversicherung*, assurance vie), adverbe-nom (*Nichtraucher,* non-fumeur), et préposition-nom (*Nachkriegszeit*, période d'après-guerre). Autres langues utilisent le trait d'union, Comme l'anglais (ex: *end-of-file*, *classification-based*) et le français (ex: *va-t-il*, *c'est-à-dire*). Il existe des langues non-segmentées (tous les mots sont attachés) comme le chinois, le japonais, et le thaï. Les techniques pour segmenter les mots pour ces langues sont différentes à celles utilisées pour les langues avec espace. Il existe trois approches pour la segmentation :
Statistique, lexicale basée sur des règles, et une autre approche hybride entre ces deux.

L'approche statistique utilise des données comme l'information mutuelle entre les caractères, compilée d'un corpus d'apprentissage, pour déterminer quels sont les caractères les plus probables pour former un mot. L'approche lexicale utilise des règles encodées manuellement sur la langue, comme l'information syntaxique et sémantique, les structures communes de phrases, et les règles morphologiques, pour définir la segmentation. Le système d'écriture chinois est constitué de milliers de caractères appelés *Hanzi*, avec des mots d'un ou plusieurs caractères **[34]**. Il existe une multitude d'algorithmes pour la segmentation de textes chinois **[42, 43, 44, 45, 46]**.

1.4.1.3 Radicalisation

Dans tout texte, le même mot pouvait se produire dans des différentes variantes morphologiques. Dans la plupart des cas, ces formes lexicales possèdent des interprétations sémantiques similaires, et peuvent être considérées comme équivalentes pour les systèmes de gestion d'information. Afin que ces systèmes puisent utiliser ces formes comme un seul concept, on utilise souvent un algorithme pour trouver la racine du mot, appelé radicalisateur (Stemmer). Le rôle de cette tâche consiste à supprimer suffixes et préfixes afin que nous puissions obtenir le radical d'un mot.

La radicalisation tente de réduire un mot vers son radical. L'effet n'est pas seulement celui de réduire les différentes variantes d'un terme vers une seule forme représentative, mais aussi de réduire la taille du vocabulaire utilisée par le système pour stocker les représentations. Dans la plupart des cas, la petite taille du dictionnaire nous permet de préserver l'espace du stockage et le temps de traitement, ainsi de rendre le document moins bruyant, plus compact, et plus souple **[47]**. Le résultat d'une radicalisation peut être un mot qui n'a aucun sens, mais qui est commun entre les mots ayant le même sens. Le rendement d'une radicalisation dépend sur la racine résultat ; il est bon si les différents mots avec le même sens de base ont la même racine, et si les mots qui n'ont pas le même sens sont séparés. Selon ces conditions, on peut avoir deux types de problèmes **[47]** :

– Sur-radicalisation : se passe lorsque deux mots sont donnés la même racine, mais en réalité ils ne l'ont pas.

– Sous-radicalisation : se passe lorsque les mots qui doivent avoir la même forme de base, ne l'ont pas. Par exemple, les deux mots "running" et "ran" doivent avoir la même racine "run", mais le système nous donne "run" et "ran" en ordre.

Dans la radicalisation, il existe deux grandes approches **[32]** : radicalisation linguistique/avec-dictionnaire et radicalisation de style Porter **[48]**. La première méthode donne un meilleur résultat, mais exige un cout d'implémentation et de traitement très élevé pour une petite couverture. La deuxième méthode est meilleure de point de vue implémentation et coût de traitement pour une performance acceptable. Des radicalisateurs ont été développés pour plusieurs langues, ayant compris Malais **[49]**, Latin **[50]**, Indonésien **[51]**, Suédois **[52]**, Néerlandais **[53]**, Allemand **[54]**, Français **[55]**, Slovène **[56]**, Turc **[57]**, et Arabe **[58]**.

1.4.1.4 Suppression des mots vides

La suppression de mots vides est une technique commune pour répondre au fait que plusieurs mots dans le document ne contribuent pas particulièrement dans la description de son contenu, et ne font qu'ajouter de bruit. La plupart des systèmes de recherche d'information et de résumé automatique, suppriment les mots vides avant de traiter les documents. Ceci va augmenter la performance du système, mais éventuellement peut causer des problèmes pour traiter certains cas comme les suivants (les mots vides sont en gras) :

– To be or not to be

– Will and Grace

Pour la première phrase, en supprimant les mots vides, on n'aura aucun mot à traiter, puisque tous les mots sont des mots vides, et donc cette phrase ne participera jamais dans le traitement.

La deuxième phrase contient le mot "*Will*" qui est un nom, mais le système le considère comme le verbe "*will*" présent dans la liste des mots vides pour l'anglais.

1.5 Critères de sélection statistiques

Une fois qu'on procède au pré-traitement du texte source, on aura les informations nécessaires pour effectuer le traitement. Dans cette partie nous détaillons les critères de sélection des unités textuelles utilisés par les systèmes de résumé. Ces unités dépendent du modèle de langue choisi, elles peuvent être des phrases, des N-grammes ou n'importe quel segment du texte. Ces critères ne sont pas exclusifs d'une méthode bien déterminée mais sont applicables à tous les types de résumés extractifs qu'ils soient mono-document, multi-document ou dynamiques.

1.5.1 Critères liés au contenu du texte

Cet ensemble de critères s'intéressent au contenu du texte et aux informations qu'il apporte. Le contenu est analysé soit par des approches de surface, comme le calcul des fréquences des mots, soit par des approches sémantiques qui exploitent les sens des mots et leurs relations sémantiques, comme avec l'annotation en rôles sémantiques. Nous citons, dans ce qui suit, les critères les plus utilisés.

1.5.1.1 Fréquence des mots

Ce critère a été introduit initialement par Luhn **[17]**, c'est le critère le plus utilisé pour calculer le score de chaque phrase. L'idée est que les mots les plus fréquents sont les plus liés au sujet du texte. Même les méthodes reposant sur l'analyse sémantique des mots utilisent la fréquence d'occurrence comme première étape pour déterminer les thèmes principaux abordés par le texte. Le point fort de ce critère est qu'il est totalement indépendant de la langue. Il correspond au nombre de fois que la racine d'un mot *m*, apparait dans un document *d*. Pour calculer la fréquence d'un mot m, on calcule le nombre d'apparition de toutes ces formes dans le document en question, comme par exemple : "différencier", "différent", "différenciation", etc. Souvent, le nombre de mots est divisé par le nombre total de mots dans le document, ceci afin de ne pas favoriser les documents longs ;
Ceci peut être représenté par l'équation suivante :

$$tf(m,d) = \frac{|m|_d}{\sum_{i=1}^{|d|} |m_i|_d}$$

Où : $tf(m, d)$ est la fréquence du mot *m* dans le document *d*. $|m|_d$ est le nombre d'occurrences de *m* dans le document *d*. $|d|$ est le nombre de mots différents dans le document *d*.

En utilisant la fréquence de mots, on peut rapidement détecter qu'il y a un problème pour les documents du même domaine. Dans un domaine donnée, l'informatique par exemple, certains mots sont fréquemment utilisés (exp. ordinateur, informatique, etc.) et donc ces mots vont avoir des grandes fréquences malgré qu'ils ne représentent pas le thème du document. Pour contourner ce problème, Salton **[59]** définit un autre critère appelé $tf * idf$. Le critère $tf * idf$ exprime qu'un mot est plus important lorsqu'il est plus fréquent dans le document analysé et peu fréquent dans le corpus de documents analysés. La propriété id f s'appelle "*la fréquence inverse de documents*" (en Anglais *: inverse documents frequency*). La logique derrière cette propriété est que le mot est plus important lorsqu'il est moins probable de se trouver dans les documents du corpus traité (concernant certain domaine) d'où le mot inverse :

$$Idf(m) = log \frac{|D|}{|\{d : m \in d\}| + 1}$$

Où :

– $|D|$ est le nombre des documents dans le corpus.

– $|\{d: m \in d\}|$ est le nombre des documents contenants le mot m.

Donc, le facteur $tf * idf$ va être écrit comme dans l'équation suivante :

$$tf * idf(m, d) = tf(m, d) * idf(m)$$

1.5.1.2 Reconnaissance d'entités nommées /Annotation en rôles sémantiques

La reconnaissance des entités nommées dans un texte améliore le filtrage des informations pertinentes **[60]**. Elle sert aussi à répondre à des requêtes standards (OÙ, QUI, QUAND, etc.) dans le résumé guidé **[61]**. L'entité la plus fréquente est identifiée, c'est l'entité principale. Par la suite, les phrases contenant cette entité sont sélectionnées. Enfin, seules les phrases où l'entité principale possède un rôle sémantique fondamental (non auxiliaire) sont gardées pour le résumé.

1.5.1.3 Similarité entre les phrases

La similarité des textes est une notion très importante en TAL. Plusieurs mesures de similarité textuelle ont été établies **[62]**. Dans le domaine du résumé automatique, elle est d'abord exploitée pour l'élimination de la redondance mais aussi plus indirectement pour la sélection de phrases pertinentes. Certaines méthodes de résumé s'appuient uniquement sur ce critère. Tel est le cas de l'algorithme de résumé mono-document *TextRank* **[63]**. Ce critère est par ailleurs particulièrement important dans le cas multi-document.

1.5.2 Critères liés à la forme et à la structure du texte

La structure du texte est très importante dans le jugement de la pertinence d'une phrase. En effet, lors de la rédaction d'un texte, l'ordre des phrases n'est pas arbitraire. De plus, les styles de rédaction diffèrent d'un domaine à l'autre. Nous expliquons dans cette partie les critères les plus importants.

1.5.2.1 Position dans le texte :

La position d'une phrase dans un document peut dans certains cas se révéler un bon indicateur de son importance. Celle-ci peut faire référence à la position de mots dans la phrase **[17]**, où bien à la position de la phrase par rapport à une unité (document, section, paragraphe, etc.) **[64, 18, 65, 66]**. Ce critère dépend de la nature du document et de son genre. Les phrases

se trouvant au début sont généralement plus informatives et décrivent le sujet principal du document. De plus, les phrases situées au début de chaque paragraphe tendent à apporter plus d'informations pertinentes **[65, 67, 68]**. Dans un article de presse, la première phrase sert de point d'appui à l'ensemble du document, et est considérée comme la plus importante. Dans un article scientifique, ce sont les phrases de la conclusion qui sont les plus représentatives du document (avec les phrases du résumé si celui-ci existe).

Dans **[66]**, les auteurs définissent rois méthodes pour le calcul du score de la phrase en utilisant sa position (voir les équations 1.1, 1.2, et 1.3). La première méthode donne 1 pour les *N* premières phrases, et 0 pour les autres.

$$Score_{pos(Pi)=} \begin{cases} 1 & si\ (i<N) \\ 0 & sinon \end{cases} \qquad (1.1)$$

La deuxième méthode suppose que les phrases ont une importance inversement proportionnée à leurs positions.

$$Score_{pos(Pi)} = \frac{1}{i} \qquad (1.2)$$

La troisième méthode suppose que les phrases dans le début et dans la fin du document sont les plus importantes.

$$Score_{pos(Pi)} = \max\left(\frac{1}{i}, \frac{1}{n-i+1}\right) \qquad (1.3)$$

Selon les auteurs, la deuxième méthode donne un résultat meilleur que ceux des autres méthodes.

Dans **[69]**, les auteurs utilisent la position des phrases dans le paragraphe (et pas dans la totalité de texte). Ils supposent que les premières phrases du paragraphe sont les plus importantes, en prenant cinq phrases comme la position maximale (Voir l'équation 1.4).

$$Score_{pos(Pi)=} \begin{cases} 6-i & si\ (i\leq 5) \\ 0 & sinon \end{cases} \qquad (1.4)$$

Où : *i* est la position de phrase dans le paragraphe.

1.5.2.2 Mots du titre et des sous-titres :

Dans **[18]**, l'auteur se base sur l'hypothèse que le titre véhicule le thème du document. En plus, lorsque l'auteur partitionne le document en sections, il les résume en leurs choisissant les entêtes (titres de sections) appropriées. En suivant cette hypothèse, les phrases importantes sont celles contenant des mots du titre ou des sous-titres. Bien évidemment, les mots du titre ont plus de poids que ceux des sous-titres. Dans le travail de **[70]**, les auteurs propose d'utiliser le titre comme une requête pour toutes les phrases du document ; ensuite la similarité cosinus est calculée entre chaque phrase et le titre

En se basant sur la fréquence de mot, **[71]** proposent une méthode qui prend en considération les mots appartenant au titre. En effet, lorsqu'un mot appartient au titre, sa fréquence va être multipliée par un nombre A>1. Ainsi, le score d'une phrase p_i : $Score_{titre}\ (p_i)$ est donné par l'équation suivante :

$$Score_{titre}\ (p_i) = \sum_{\{m\} \in p_i} \alpha(m) * tf(m)$$

Où :

$$\alpha(w) = \begin{cases} A > 1 & \text{si } w \in \text{Titre} \\ 1 & \text{sinon} \end{cases}$$

Dans **[66]**, deux méthodes sont utilisées pour calculer le score des phrases en se basant sur les mots du titre. Pour chaque phrase p_i et sachant le titre T, le score de cette phrase est donné par $Score_{titre}\ (p_i)$. La première méthode utilise le $tf * idf$ des mots de titre, comme il est indiqué dans (l'équation 1.5) :

$$Score_{titre}\ (p_i) = \frac{\sum_{m \in T \cap P_i} \frac{tf(m)}{tf(m)+1} idf(m)}{\sum_{m \in T} \frac{tf(m)}{tf(m)+1} idf(m)} \qquad (1.5)$$

La deuxième méthode utilise les entités nommées et la fréquence de mots tf. Pour une entité nommée e, l'équation 1.6 est utilisée pour calculer le score d'une phrase p_i en se basant sur le titre.

$$Score_{titre}\ (p_i) = \frac{\sum_{e \in T \cap P_i} \frac{tf(e)}{tf(e)+1}}{\sum_{e \in T} \frac{tf(e)}{tf(e)+1}} \qquad (1.6)$$

Selon les auteurs, la deuxième méthode donne un meilleur résultat.

1.5.2.3 Longueur de phrases :

Ce critère est utilisé pour pénaliser les phrases qui sont trop petites, vu que ces phrases ne seront pas incluses dans le résumé **[20]**. Le score pour ce critère est la longueur normalisée de phrase ; qui est la proportion entre le nombre de mots dans la phrase et le nombre de mots dans la phrase la plus longue dans le document **[72, 66]**.

Dans le travail de **[66]**, deux méthodes sont utilisées. La première donne un score égal à la longueur de phrase divisée par une longueur L_{max} prédéfinie, si la longueur de phrase est supérieure à L_{max} on lui attribue un score égal à 1 (voir l'équation *1.7*)

$$Score_{long}(p_i) = \begin{cases} \frac{L_i}{L_{max}} & si\ (L_i \leq L_{max}) \\ 1 & sinon \end{cases} \qquad (1.7)$$

La deuxième méthode donne un score négatif afin de pénaliser les phrases qui sont plus courtes que L_{min} (voir l'équation 1.8).

$$Score_{long}(p_i) = \begin{cases} 0 & \text{si } (L_i \geq L_{min}) \\ \frac{L_i - L_{min}}{L_{min}} & Sinon \end{cases} \qquad (1.8)$$

Selon les auteurs, la deuxième méthode donne un meilleur résultat.

Une autre formule pour calculer le score d'une phrase pi dans un document d en utilisant sa longueur, est utilisée dans **[69]** (voir l'équation 1.9).

$$Score_{pos}(P_i) = \frac{|P_i| * |\{P : P \in d\}|}{|d|} \qquad (1.9)$$

Où : $|P_i|$ et $|d|$ sont les nombres de mots dans la phrase P_i et le document d respectivement. $|\{P : P \in d\}|$ est le nombre de nombre de phrases dans le document d.

1.5.2.4 Mots indicatifs :

Les méthodes utilisant les mots indicatifs (*cue words*), sont basées sur l'hypothèse que la pertinence d'une phrase est affectée par la présence de mots comme "*significatif* ", "*impossible*", et "*Difficilement*" **[18]**. Ces méthodes utilisent des mots sauvegardés dans un dictionnaire extrait à partir d'un corpus. Le dictionnaire de mots clés contient trois sous-dictionnaires : Les mots Bonus, qui sont pertinents positivement (important, précisément, particulièrement, etc.) ; Les mots Stigmates, qui sont plutôt pénalisant (obscurément, peut-être, etc.) ; et les mots *Nul*, qui ne sont pas indicatifs. Le score total pour une phrase P_i basé sur ce critère, est la somme des poids *cue(m)* pour ses mots constituants *m*. Ceci est indiqué dans l'équation suivante, où *cue(m)* est le poids du mot *m* par rapport au dictionnaire de mots clés.

$$Score_{cue}(p_i) = \sum_{m \in p_i} cue(m)$$

Sachant que :

$$cue(m) = \begin{cases} b>0 \quad si\ (w \in Bonus) \\ \delta < 0\ si\ (w \in Stigma) \\ 0 \qquad sinon \end{cases}$$

1.5.2.5 Expressions Saillantes :

Les expressions saillantes sont des structures, lorsqu'elles se produisent dans une phrase, expriment explicitement que cette phrase contient une information importante à propos du sujet ou un "message" du document **[73]**. Par exemple "Le but *principal de cet article est de vérifier* ...", "*Dans cet article, une méthode est décrite pour* ...", etc.

D'après Paice, l'identification des expressions saillantes n'est pas une tâche simple (une comparaison de chaînes de caractères ne suffit pas), à cause des raisons suivantes :

– On ne peut pas lister toutes les expressions saillantes à cause des variations qui peuvent exister. Par exemple, les expressions suivantes ont la même structure : "*This article is concerned with* ...", "*Our paper deals with* ...", "*The present report concerns* ..." et "*The following discussion is about* ...". Donc, la solution est d'utiliser des modèles contenant des mots appelés les paradigmes (prototypes, modèles).

– En utilisant les modèles, il parait qu'il existe des mots qui ne figurent pas dans ces modèles malgré qu'ils fassent partie d'une expression indicative. La solution est d'utiliser des sauts limités entre les paradigmes d'un modèle donné.

– Il existe des mots qui sont optionnels, mais qui ajoute un poids lorsqu'on les utilise, comme le mot "*here*" dans l'expression "*The purpose here is to ...*". On peut définir de multiples chemins dans le modèle.

– Enfin, les mots peuvent avoir plusieurs variations, ce problème peut être résolu en utilisant leurs racines dans les modèles.

La figure 1.2 représente un modèle pour détecter les expressions saillantes, où les mots sont les paradigmes. Les limites de sauts entre les paradigmes sont présentées comme [3]. Le gain de poids de l'expression est présenté comme +2. Les paradigmes suivis par (?) sont optionnels.

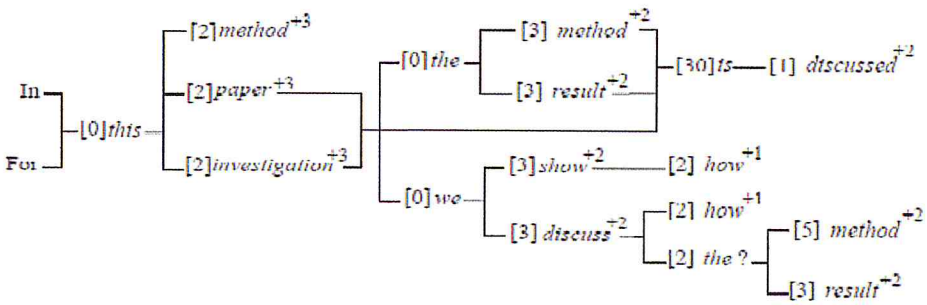


Figure 1.2 : Un des modèles pour détecter les expressions saillantes

Nous avons cité les critères de sélection les plus utilisés pour le résumé automatique. Le choix de bons critères n'est pas suffisant pour obtenir un bon résumé. Il faut savoir comment les utiliser et quel degré d'importance accordé à chacun pour produire un résumé satisfaisant. La section suivante s'intéresse à cette problématique.

1.6 Exploitation des critères :

Il est très rare qu'un système de résumé automatique utilise un seul critère pour sélectionner les phrases du texte source. Plusieurs critères sont combinés. Les méthodes d'intégration sont assez nombreuses. Nous décrivons dans cette partie les différentes méthodes pour combiner les critères et les utiliser pour sélectionner les phrases du résumé.

1.6.1 Méthodes d'analyse de surface :

Dans ce type de méthode, les différentes phrases du ou des documents à résumer se voient attribuer un score qui peut être fondé sur un des critères ou une combinaison des critères précédents, établis par Edmundson **[18].** Parmi les méthodes utilisant uniquement la fréquence des mots afin d'attribuer un score à une phrase et extraire les mieux classées, on peut citer :

[17] Celui-ci construit à partir des documents à résumer un paquet de mots importants dans ces documents. Seuls les mots dont la fréquence est située dans un certain intervalle sont considérés importants et intégrés dans ce paquet (**figure 1.3).** Plus de mots appartenant à ce paquet sont sélectionnées. Cette méthode comporte un défaut majeur : un terme est ajouté au paquet des termes importants si sa fréquence est située dans un intervalle donné. L'intervalle optimal dépend des documents à résumer et Luhn ne propose pas de solution pour faire varier cet intervalle selon les documents en entrée du système.

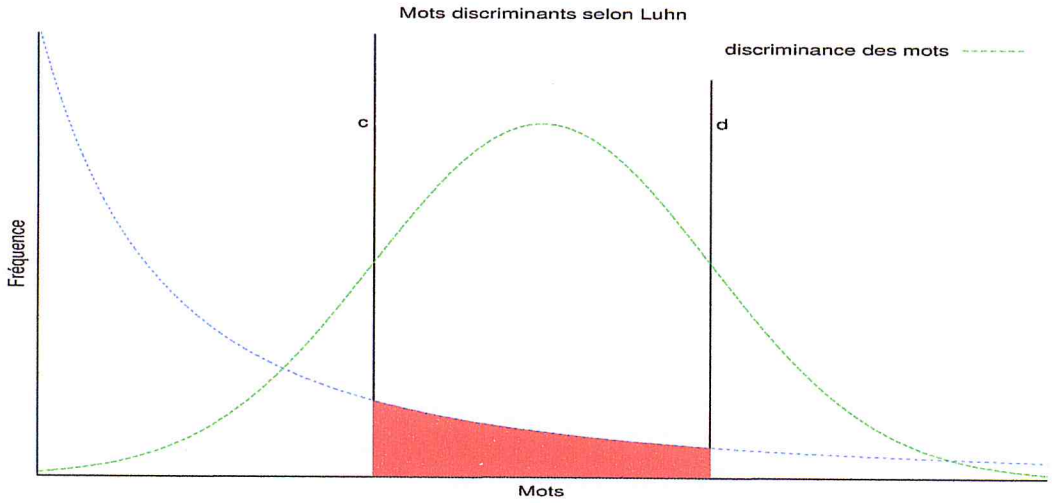


Figure 1.3 : Illustration de la discriminance des mots selon leur fréquence d'après Luhn

Les avancées en analyse statistique appliquée au texte ont permis de pallier en partie ce problème :

Salton propose dans **[74]** une métrique, appelée *tf.idf* (*term frequency, inverse document frequency*), qui permet de déterminer l'importance d'un terme dans un ensemble de documents sans utilisation de ressources exogènes. Cette mesure est fondée sur l'observation que les termes les plus discriminants/importants sont ceux qui apparaissent le plus dans un faible nombre de documents du corpus. Le *tf.idf* d'un terme dans un document est alors la fréquence de ce terme dans le document considéré multiplié par l'inverse de la fréquence des

documents dans lesquels le terme apparaît. Il existe plusieurs façons de calculer le *tf.idf*, la plus courante est détaillée en **Figure 1.4**

Calcul du tf d'un terme t_i dans le document d_j :

$$\mathrm{tf}_{t_i,d_j} = \frac{|t_i|_j}{\sum_{k=0}^{n} |t_k|_{d_j}}$$

Calcul de l'idf d'un terme t_i au sein d'un corpus D de documents d_j :

$$\mathrm{idf}_{t_i} = \log \frac{|D|}{|d_j : t_i \in d_j|}$$

Calcul du tf.idf du terme t_i pour un document d_j :

$$\mathrm{tf.idf}_{t_i,d_j} = tf_{t_i,d_j} \times idf_{t_i}$$

Figure 1.4 : Calcul du *tf.idf*

Cette métrique a été appliquée de différentes manières dans le cadre du résumé automatique :

– **[75]** évaluent plusieurs façons de calculer le *tf.idf* au travers d'une application de résumé automatique. Les phrases obtiennent entre autres scores un score égal à la somme des *tf.idf* des mots qui les composent.

– **[76]** utilisent la moyenne des *tf.idf* des mots d'une phrase.

Ces deux méthodes posent le problème de la taille des phrases. Dans le premier cas, les phrases les plus longues seront favorisées ; dans le deuxième cas, les phrases courtes éventuellement peu informatives mais composées uniquement de mots importants se verront avantagées.

– **[77]** extraient les phrases qui maximisent une combinaison de treize critères différents, dont la similarité au titre du document, la somme de l'entropie des différents termes de la phrase, et d'autres mesures de surface sur les phrases et les termes qui les composent.

1.6.2 Méthodes par apprentissage :

Les systèmes de résumé automatique par apprentissage combinent différentes caractéristiques pondérées. L'objectif de ces systèmes est alors de trouver la combinaison de poids pour chacune de ces caractéristiques qui permette de maximiser la qualité des résumés.

[78] ajoutent deux caractéristiques aux phrases : un score de centralité (l'intersection entre les mots-clés de la phrase et les mots-clés de l'ensemble des documents divisé par l'union de ces deux ensembles) et la similarité au titre du document dont la phrase est issue. La meilleure

combinaison des paramètres est approximée en utilisant un algorithme génétique. **[79]** utilise une descente de gradient pour optimiser les poids des paramètres.

[80] Les réseaux de neurones artificiels (RNA) ont aussi été utilisés pour l'apprentissage supervisé, Chaque phrase du texte est modélisée par un vecteur de n composantes, chacune correspondant à un critère de sélection. Le RNA est composé de n neurones d'entrées (un neurone par critère), une couche cachée de p neurones et un neurone de sortie. Ce dernier indique si la phrase en entrée doit être incluse dans le résumé. L'apprentissage permet d'adapter le poids des liaisons entre les couches en fonction d'un ensemble d'exemples. À l'issue de cet apprentissage, les liaisons de très faible poids sont éliminées, de même que les neurones isolés.

1.6.3 Méthodes fondées sur la programmation linéaire en nombres entiers :

[81] fut le premier à modéliser le problème du résumé automatique multi-document par le biais de la Programmation Linéaire en Nombres Entiers (PLNE). Plusieurs chercheurs utilisent cette méthode pour combiner différents critères de sélection **[82, 83]**. Le principe consiste à maximiser une fonction objective favorisant les unités textuelles satisfaisant les critères de sélection et pénalisant la redondance entre les unités sélectionnées. Le calcul est effectué sous un ensemble de contraintes à fixer selon le système de résumé. La contrainte triviale est la taille souhaitée du résumé.

Le modèle proposé par McDonald attribue des poids aux phrases du texte source. La PLNE permet de choisir les phrases ayant les poids maximaux et une redondance minimale avec les phrases déjà sélectionnées sans dépasser la taille maximale du résumé.

1.6.4 Méthodes à base de graphe :

En représentant un texte sous la forme d'un graphe de phrases, il devient possible d'appliquer un certain nombre d'algorithmes génériques. Les méthodes de résumé automatique à base de graphe ont récemment bénéficié des avancées de la recherche dans l'étude des réseaux sociaux. En effet, si l'on considère un corpus comme un réseau social où les différents acteurs du réseau sont les phrases du corpus, extraire les phrases centrales revient à extraire d'un réseau social les nœuds les plus influents, donc ceux qui entretiennent le plus de liens avec les autres. Ce type d'analyse permet de rendre compte de la centralité d'une phrase en tirant parti du contenu global des documents, et non plus uniquement de la requête et de listes de termes, forcément limitatifs.

L'algorithme *LexRank*, développé par Radev **[84]**, se fonde sur cette observation. Le but du système est d'extraire les phrases considérées comme « centrales » dans un corpus, en se fondant sur la notion de « prestige » des réseaux sociaux. Un graphe du corpus est établi, chaque nœud étant une phrase et chaque arête recevant comme poids la similarité des deux phrases qu'elle lie. Cette similarité est calculée à l'aide de la mesure *cosinus*. Cette mesure établit une similarité entre deux vecteurs en calculant le cosinus de l'angle de ces deux vecteurs. Pour calculer la similarité entre phrases, il faut établir un vecteur pour chaque phrase. Le vecteur représentant une phrase est généralement rempli avec les *tf.idf* des constituants de cette dernière.

PageRank **[85]** est un algorithme de classement utilisé par le moteur de recherche de Google. Il représente les pages Web par les sommets d'un graphe et les liens par les arcs. Il attribue récursivement à chaque nœud un score dépendant de la structure de tout le graphe.

TextRank est un algorithme pour le résumé automatique mono-document fondé sur les graphes **[63]**. Le texte est représenté par un graphe où les sommets sont les phrases du texte. Les arcs dans *TextRank* représentent leurs similarités. Tandis que *LexRank* et *MEAD* ont été conçus pour le résumé multi-documents. Le résumé multi-documents comporte un problème supplémentaire : étant donné la multiplication des sources, le risque de sélectionner des phrases redondantes est accru. Pour résoudre ce problème, les phrases trop similaires à une phrase déjà classée sont éliminées. La métrique *CSIS*, *« Cross sentence informational subsumption »* **[86]** est utilisée à cet effet.

1.7 Evaluation des résumés automatiques :

L'évaluation des résumés automatiques est une problématique importante à laquelle les travaux de recherche n'ont répondu que partiellement. Avec le développement du domaine et l'abondance des travaux proposés, des campagnes d'évaluation annuelles (DUC, TAC, TREC) ont été organisées afin de comparer les systèmes de résumé. Les premières évaluations reposaient sur le jugement des lecteurs concernant la qualité linguistique et le contenu du résumé, soit en estimant la similarité des résumés candidats avec un résumé manuel (évaluation objective), soit en jugeant la qualité du résumé sans se référer à un modèle (évaluation subjective). La dernière variante correspond à la mesure *Responsiveness* utilisée jusqu'à aujourd'hui pour évaluer le résumé de point de vue du contenu et de la qualité linguistique. Ces méthodes nécessitent un fort investissement en temps et en effort, ce qui pose problème pour le développement des systèmes de résumé. C'est pourquoi des métriques

standards, avec une mise en œuvre automatique, ont été proposées pour rendre plus facile la comparaison des différentes approches. Les méthodes répondant à cette problématique s'intéressent plus à l'évaluation du contenu sélectionné qu'à la qualité linguistique ou grammaticale. Par ailleurs, l'automatisation n'est que partielle.

En effet, pour juger un résumé, celui-ci est comparé à un résumé manuel (idéal, modèle ou de référence). Ces systèmes dépendent donc de la disponibilité des résumés manuels. Trois métriques sont généralement utilisées pour quantifier la comparaison :

Précision : Elle traduit à quel point les données sélectionnées sont pertinentes. Concrètement, il s'agit du rapport du nombre d'unités textuelles communes entre le résumé candidat et les résumés de référence sur le nombre de toutes les unités textuelles du résumé candidat.

Rappel : Il reflète à quel degré le résumé candidat rappelle (évoque) des données pertinentes qu'il est sensé inclure. Il désigne le rapport des unités textuelles communes aux résumés candidat et de référence sur le nombre de toutes les unités textuelles du résumé de référence.

F-mesure : C'est la moyenne harmonique de la précision et du rappel. D'après les résultats d'évaluation des systèmes de résumé, le rappel est généralement plus difficile à obtenir que la précision.

Dans ce qui suit, nous présentons les deux méthodes d'évaluation semi-automatique les plus utilisées : ROUGE et PYRAMID. Leur succès est en particulier lié à leurs fortes corrélations avec les jugements humains :

1.7.1 ROUGE :

Les mesures ROUGE, pour *Recall-Oriented Understudy for Gisty Evaluation*, ont été introduites par Lin dans **[87]**.

ROUGE évalue les résumés en les comparants à des résumés modèles. Cette comparaison est automatique et ne nécessite pas un pré-traitement particulier. Elle est déduite à partir du recouvrement entre les N-grammes des deux textes. Il n'existe pas un unique résumé de référence, et il est donc essentiel de comparer les résumés automatiques à plusieurs résumés de référence établis manuellement afin d'obtenir des mesures plus précises de la qualité des résumés. Ces mesures nécessitent donc la rédaction de résumés de référence par un ou plusieurs experts au préalable de la mesure de qualité du résumé. Cette méthode a montré une forte corrélation avec les jugements humains **[87]**. La corrélation de Pearson des scores ROUGE-2 avec les jugements humains, pour le résumé multi-document, varie entre 0,85 et

0,94 en utilisant 3 résumés de référence et en éliminant les mots vides. Cette corrélation augmente avec le nombre de résumés modèles. Il existe plusieurs variantes de ROUGE exploitant des modèles autres que les N-grammes, comme la plus longue sous-séquence commune ou les bi-grammes distants. La dernière implémentation de ROUGE permet de calculer en plus la précision et la f-mesure. Jusqu'à présent ROUGE est l'outil d'évaluation le plus utilisé. Il en existe plusieurs variantes, que nous présentons ci-dessous :

1.7.1.1 ROUGE-n:

Les métriques ROUGE-*n* sont fondées sur la comparaison simple de n-grammes. Une liste des n-grammes est établie pour chacun des résumés de référence et des résumés cibles. Les n-grammes sont composés de n mots consécutifs. Par exemple, pour le texte« ROUGE est une métrique d'évaluation », la liste de n-grammes créée par ROUGE-2 sera « ROUGE est », « est une », « métrique d' », « d' évaluation ». Une fois la liste des n-grammes établie, le score ROUGE est calculé selon la formule en **figure 1.5**.

$$\frac{\sum_{r \in CRef} \sum_{n-grammes \in RC} card_{match}(n-grammes)}{\sum_{r \in CRef} \sum_{n-grammes \in RC} card(n-grammes)}$$

- $CRef$ est la collection des résumés de références,
- RC le résumé cible (le résumé à évaluer).

Figure 1.5: Formule de calcul du score ROUGE-n : la formule revient à diviser le nombre de n-grammes communs entre le résumé à évaluer et les résumés de référence par le nombre total de n-grammes des résumés de référence.

Cette mesure présente un défaut majeur : l'ordre des mots n'influe pas toujours sur le résultat. Ainsi, l'exemple suivant, inspiré de la présentation par Lin de ROUGE au Workshop« *Text Summarization Branches Out* », montre à quel point les scores ROUGE-*n* peuvent être dé corrélés de la réalité :

– Phrase 1 (référence) : Dr Jekyll tua Hide– Phrase 2 : Dr Jekyll tue Hide– Phrase 3 : Hide tue Dr Jekyll

Les scores ROUGE-*n* des phrases 2 et 3 seront similaires. En effet, les deux phrases sont les mêmes n-grammes en commun avec la phrase 1, à savoir (« Dr Jekyll », « Hide »).

1.7.1.2 ROUGE-L:

Afin de pallier en partie un défaut de ROUGE-*n*, à savoir le fait que l'ordre des mots dans les phrases n'est pas pris en compte, **[88]** a introduit une autre mesure, ROUGE-L. Etant donné deux séquences S1 (référence) et S2, ROUGE-L est définie comme étant la plus longue sous-séquence commune (LCS) à S1 et S2 divisée par le nombre total d'éléments de S1. Ainsi, les scores des phrases 2 et 3 seront :

– phrase2 = (« Dr Jekyll Hide ») = 3/4

– phrase3 = (« Dr Jekyll ») = 2/4

Cependant, cette mesure n'est pas totalement satisfaisante. En effet, la phrase « Dr Jekyll a été tué par Hide. » obtiendrait avec cette mesure le même score que la « Dr Jekyll tue Hide. ».

1.7.1.3 ROUGE-SUn:

ROUGE-SU, pour *skip-bigram and unigram ROUGE* prend en compte des bi-grammes à trou ainsi que les unigrammes des résumés de référence et du résumé cible. Les bigrammes à trou (*skip-bigram*) sont définis dans **[88]** comme n'importe quelle paire de mots dans l'ordre de la phrase, séparés par une distance maximale arbitraire (le *n*). Ainsi, avec ROUGE-SU4, la phrase 1 comprend6 bi-grammes et 4 unigrammes :

« Dr Jekyll », « Dr tua », « Dr Hide », « Jekyll tua », « Jekyll Hide », « tua Hide » et les 4 unigrammes qui composent la phrase. Les phrases 2 et 3 ont ainsi pour score respectivement : 6/10 et 4/10.

Cette mesure permet de rendre compte efficacement des relations de dépendance éloignées dans le texte. La campagne TAC 2008 l'a d'ailleurs confirmé puisque parmi les mesures ROUGE, c'est ROUGE-SU4 qui est la plus corrélée aux jugements humains.

1.7.2 PYRAMID :

Cette méthode permet de comparer un résumé candidat à un ensemble de résumés de référence **[89]**. Étant donné qu'un résumé idéal n'existe pas et que les styles de rédaction diffèrent d'une personne à l'autre, l'utilisation d'un seul résumé de référence ne satisfait pas la condition d'équité entre les résumés candidats. Pour relaxer cette contrainte, les campagnes d'évaluation présentent au moins 4 résumés modèles. Le principe de la méthode PYRAMID consiste à annoter les résumés de référence afin d'identifier les unités appelées SCUs (*Summary Content Units*). Un SCU est un ensemble d'unités textuelles des résumés de référence exprimant la même information. Il lui est assigné un poids égal au nombre de

résumés de référence qui l'instancient. Ces SCUs peuvent être organisés en pyramide où chaque couche regroupe les SCUs de même poids. Pour évaluer un résumé, ce dernier est annoté afin de repérer les SCUs candidats qu'il contient. Par la suite, chaque SCU candidat hérite du poids du SCU le plus similaire dans la pyramide. Le score PYRAMID du résumé est finalement le rapport de la somme des poids de tous ses SCUs candidats sur la somme des poids d'un résumé idéal ayant le même nombre de SCUs.

L'inconvénient de cette méthode est qu'elle nécessite une étape d'annotation des résumés. Le calcul du score PYRAMID a été automatisé en utilisant la sémantique distributionnelle **[89].** Malheureusement, l'annotation des résumés modèles reste difficile à automatiser.

1.8 Conclusion :

Dans ce chapitre, nous avons présenté un état de l'art sur le résumé automatique basé sur l'approche statistique. Dans un premier temps, nous avons présenté quelques définitions pour le résumé, afin de comprendre ce domaine. Un résumé automatique peut appartient à plusieurs classes ou types, puisque il existe plusieurs critères de classification. Donc, nous avons vu qu'on peut diviser ces critères selon le document d'entrée, le but, et le document de sortie.

Le domaine de résumé automatique est très vaste, nous avons essayé de couvrir l'essentiel. Nous avons discuté quelques méthodes utilisées pour améliorer l'approche statistique en utilisant les différents critères de sélection pour le calcul des scores de phrases. Enfin nous avons présenté les différentes méthodes d'évaluation de résumé automatique de textes.

Dans le chapitre suivant, nous allons définir la classification automatique de texte avec ses deux types (catégorisation et clustering), l'idée générale est d'améliorer l'approche statistique en introduisant les différentes méthodes de classification.

Chapitre 2

Classification automatique de textes

2.1 Introduction

Actuellement, la classification de textes est un domaine de recherche très actif et l'automatisation de cette opération est devenue un enjeu pour la communauté scientifique, les travaux évoluent considérablement depuis une vingtaine d'années et plusieurs modèles ont vu le jour comme le filtrage (classification supervisée bi-classe), le routage (classification supervisée multi-classe) ou le classement ordonnée (classement des textes par ordre de pertinence pour chaque catégorie).

Avec ces modèles, des méthodologies de tests et des outils d'évaluation ont été mises en place. Les algorithmes de classification fonctionnent correctement mais déterminer les avantages des uns par rapport aux autres reste souvent délicat ou même améliorer les performances de la même méthode en intégrant d'autres paradigmes reste toujours un domaine de recherche très prometteur.

Le principe d'une classification automatique de textes est d'utiliser un modèle afin de regrouper les phrases similaires, c'est-à-dire thématiquement proches suivant certains critères. On distingue dans le domaine de la classification automatique deux types d'approches : la classification supervisée et la classification non supervisée, dans ce chapitre nous détaillerons les méthodes du deuxième type « clustering » qui sont des techniques statistiques largement utilisées dans la Fouille de Textes.

2.2 Les systèmes de classification

L'objectif de la CT (Classification Textuelle) est de classer de façon automatique les phrases dans des catégories qui ont été définies soit préalablement par un expert, il s'agit alors de classification supervisée ou catégorisation, soit de façon automatique, il s'agit alors de classification non supervisée ou encore Clustering.

Dans ce qui suit, nous allons essayer de distinguer entre les différentes variantes de classification de textes.

2.2.1 Catégorisation (Supervisé) :

La *catégorisation* de textes correspond à la procédure d'affectation d'une ou de plusieurs catégories ou classes prédéfinies à un texte. Elle correspond à la *classification supervisée* pour l'apprentissage automatique et à la *discrimination* en statistiques alors que la recherche d'informations utilise des termes plus proches de l'application concernée : *filtrage* ou *routage.*

Cette problématique a par ailleurs dernièrement trouvé de nouvelles applications dans les domaines du traitement du langage tels que : l'affectation de sujets en recherche d'information, l'aide de l'utilisateur pour l'indexation de documents **[90]**, la veille technologique, le filtrage personnalisé des documents intéressant un internaute connaissant ses préférences de sujets (catégories), le routage de textes (tels que le courrier) et l'amélioration de la recherche sur le web **[91]**, et enfin l'organisation des sources textuelles de plus en plus nombreuses, en particulier des pages web. Aujourd'hui, cette problématique utilise largement des méthodes issues de l'apprentissage automatique et beaucoup d'algorithmes d'apprentissage supervisé lui ont été appliqués (Naïve bayes, K-plus proches voisins, arbres de décision, machines à vecteurs support, réseaux de neurones, etc.).

2.2.2 Clustering (Non supervisé)

Toutefois quand l'ensemble des catégories n'est pas donné au départ, et qu'il s'agit de le créer en regroupant les textes en classes qui possèdent un certain degré de cohérence interne, on est dans un contexte de classification non supervisée pour l'apprentissage automatique.

La classification non supervisée (Le Clustering) aussi connu sous nom (Segmentation) est un regroupement en classes homogènes consistent à représenter un nuage des points d'un espace quelconque en un ensemble de groupes appelé Cluster qui possèdent les propriétés suivantes :

- Homogénéité dans les groupes, i.e. les données appartenant à un même cluster doivent être les plus similaires possibles.
- Hétérogénéité entre groupe, i.e. les données appartenant à différents clusters doivent être les plus dissemblables possibles.

Le Clustering consiste donc, à diviser les objets (dans notre cas des textes) en groupes sans connaître à priori leurs classes d'appartenance.

Notre travail va être concentré sur le Clustering (la classification non supervisée).

2.2.2.1 Techniques de classification non supervisée

Toutes les techniques de clustering que nous allons étudier font référence à la notion de mesure de similarité entre deux données. Les performances de ces techniques diminuent considérablement non pas en qualité des résultats mais en rapidité des calculs, dans ce contexte, il existe plusieurs variantes de distance, et divers mesures de similarités entre classes qui peuvent être utilisé.

Dans ce qui suit, on va évoquer les différents choix possibles pour la distance, pour ensuite présenter les mesures de similarités les plus utilisées dans les domaines de la recherche d'information et la classification.

2.2.2.1.1 Mesures de similarités et formules pour le calcul distance

Plusieurs méthodes de classification s'appuient sur le principe des mesures de distance. Selon **[92]**, «tout système ayant pour but d'analyser ou d'organiser automatiquement un ensemble de données ou de connaissances doit utiliser, sous une forme ou une autre, un opérateur de similarité dont le but est d'établir les ressemblances ou les relations qui existent entre les informations manipulées ». Donc la similarité est une partie importante dans la définition d'une méthode de Clustering.

Une bonne méthode de clustering est une méthode qui maximise la ressemblance entre les données à l'intérieur de chaque cluster, et minimise la ressemblance entre les données des différents clusters. C'est pourquoi les résultats d'une technique de clustering dépendent fortement de la mesure de similarité choisie par son concepteur, qui doit la choisir avec prudence. En effet la mesure de similarité repose sur le calcul de la distance entre deux données, sachant que chaque donnée est composée d'un ensemble d'attributs numériques et/ou textuels.

i. Attributs numériques

Soit x_i et x_j deux données déférentes dont on veut calculer la distance. Cette distance est composée de la distance entre les valeurs des attributs numériques. Pour mesurer cette distance, plusieurs formules existent :

- La distance Euclidienne :

$$D_n(x_i, x_j) = \sqrt{\sum_{k=1}^{n_n}(x_{ik} - x_{jk})^2} \qquad (2.1)$$

– La distance City blocs :

$$D_n(x_i, x_j) = \sum_{k=1}^{n_n} |x_{ik} - x_{jk}| \qquad (2.2)$$

– La distance de Minkowksi :

$$D_{np}(x_i, x_j) = (\sum_{k=1}^{n_n} (x_{ik} - x_{jk})^2)^2 \qquad (2.3)$$

ii. Attributs textuels

- **Définition :**

Une distance est une fonction de E × E, où E est un espace vectoriel. Cette fonction est caractérisée par les propriétés suivantes :

$D(x, y) \geq 0$ $\qquad$ $D(x, y) = 0 \Longleftrightarrow x = y$

$D(x, y) = D(x, y)$ $\qquad$ $D(x, y) \leq D(x, z) + D(z, y)$

x, y, z sont des éléments de l'espace E. **[93]**, ces éléments sont des textes.

- **Variantes de distance**

Une formule générale est connue pour mesurer les distances dans les espaces vectoriels c'est la grandeur de Minkowski :

$$D_k(x, y) = \sqrt{(\sum_i |x_i - y_i|^k)^{1/k}}$$

A partir de cette formule très générale, plusieurs distances connues en pratique sont déduites :

- Comme la distance euclidienne, dans le cas où k = 2, exprimée par :

$$D_e(x, y) = \sqrt{\sum_i (x_i - y_i)^2}$$

- Où la distance de Manhattan, dans le cas où k = 1, formulée par :

$$D_m(x, y) = |x_1 - y_1| + |x_2 - y_2| + \cdots + |x_n - y_n|$$

- Où aussi la distance de Chebyshev, dans le cas où k = ∞, définie par :

$$D_c(x, y) = \max\{|x_1 - y_1|, |x_2 - y_2|, \ldots, |x_n - y_n|\}$$

Les résultats fournis par un classifieur peuvent varier suivant l'utilisation de telle ou telle distance.

- **Mesure de similarité**

- Cosinus : Dans les modèles vectorielles, la mesure du cosinus est la plus utilisée pour définir la similarité entre deux éléments (comme dans notre cas les textes) qu'on veut comparer. Peut alors être définie par le cosinus de l'angle séparant les vecteurs des deux éléments [74].

Par rapport à un simple produit scalaire, cette mesure présente l'avantage de normaliser les scores de chaque objet en fonction de sa taille, elle même pondérée par le poids des termes. Le résultat renvoyé est facilement exploitable ensuite car c'est une valeur située entre 0 et 1. La valeur 1 indiquant une similarité maximum (les deux objets sont identiques) et 0 une similarité nulle (les deux objets n'ont absolument rien en commun). Cette mesure est égale au produit scalaire divisé par les deux vecteurs sont qui sont déjà normalisés.

$$Cosinus(i,j) = \frac{\sum_{w \in i \cap j} TFIDF_{w,i} \times TFIDF_{w,j}}{\sqrt{(\sum_{w \in i} TFIDF_{w,i}{}^2)} \times \sqrt{(\sum_{w \in j} TFIDF_{w,j}{}^2)}}$$

Avec : w un terme, i et j : les deux objets (textes ou phrases) à comparer.

$TFIDF_{w,i}$: Le poids du terme w dans i et $TFIDF_{w,j}$: le poids du terme w dans j.

Ce qui peut se traduire de la façon suivante :

« ***Plus on a de termes communs et plus ces termes communs ont des pondérations fortes, plus la similarité sera proche de 1, donc forte et vice versa.»***

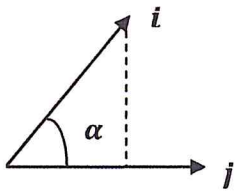


Figure 2.6 : La mesure de similarité Cosinus

i et j sont respectivement les vecteurs représentant les textes

2.2.2.1.2 Méthodes de Clustering

Il existe plusieurs algorithmes de classification non supervisée : ils peuvent être regroupés en trois grandes catégories; les approches hiérarchiques, les approches de clustering par partitionnement et les approches basées sur la densité des objets dans leur espace de représentation. Cette taxonomie est présentée dans le tableau suivant **[94]** :

Approches de clustering	**Algorithmes**
Approche par partitionnement	K-Means, Fuzzy C-Means, IsoData, Fast Global K-Means, K-Means++, CLARANS, etc.
Approche hiérarchique	Classification hiérarchique ascendante CAH, Classification hiérarchique descendante, CURE, BIRCH, l'algorithme de Ward, DIANA, etc.
Approche de clustering basé sur la densité	Denclust, Mean-scihft, Density-based spatial clustering of applications with noise (DBSCAN), etc.

Figure 2.7 : Taxonomie des méthodes de clustering

Dans la littérature certains auteurs considèrent les méthodes à centre mobile et les méthodes basées sur la densité comme des méthodes par partitionnement **[95]** tandis que d'autres considèrent les méthodes de partitionnement et les méthodes basées sur la densité comme deux catégories différentes **[96].** Les méthodes non supervisées de classification peuvent être dures ou floues ; les méthodes dures attribuent à chaque objet une seule étiquette, tandis que dans une classification floue, un objet peut appartenir simultanément à plusieurs classes. Les méthodes floues peuvent être facilement converties dans des méthodes dures.

Une autre distinction à faire concerne le type de résultat obtenu. Suivant les méthodes, les clusters obtenus peuvent être des ensembles durs ou flous. Certains objets peuvent ne pas être classés, et certains clusters peuvent se recouvrir. De plus, le résultat n'est pas forcément plat, et peut se présenter sous la forme d'une hiérarchie.

Il existe une trentaine de méthodes et leurs variations incluent dans différentes familles. Nous présentons les principales familles de méthodes de regroupement des données en clusters. Cette taxonomie est inspirée des articles d'état de l'art dans le domaine **[95]**. Les méthodes peuvent être séparées en quatre groupes :

- Les méthodes basées sur une distance : Ces méthodes se basent sur la notion de distance entre objets du jeu de données, en posant que si deux objets sont proches suivant cette distance, ils doivent être regroupés ensemble dans un même cluster. Les algorithmes Kmeans **[97]** et fuzzy-c-means **[98]** sont les algorithmes les plus connus de cette famille d'algorithme. Ces méthodes permettent de trouver des formes de clusters convexes et sont très utilisées notamment à cause de leur coût algorithmique faible.
- Les méthodes basées sur une grille : leur processus consiste à regrouper les cellules denses les plus proches. Ces méthodes ont été proposées pour réduire l'explosion combinatoire des méthodes à base de densité qui fait suite à l'augmentation du nombre d'objets. L'algorithme bang **[99]** effectue ce regroupement de manière hiérarchique, en partant de la grille et en fusionnant successivement les cellules denses voisines dont la différence de densité ne dépasse pas un certain seuil. L'algorithme clique **[100]** est une méthode très populaire basées sur les grilles.
- Les méthodes probabilistes : Ces méthodes supposent que les données suivent une certaine loi de probabilité. L'objectif est d'estimer les paramètres de cette loi et de définir un modèle de mélange de lois pour représenter les différents clusters. Ces méthodes font l'hypothèse qu'à chaque cluster Ci est associée une loi de probabilité P(x, θi) de paramètres θi qui permet de déterminer la probabilité d'appartenance de x à Ci. Si on note πi la proportion de la $i^{ème}$ loi dans le mélange, les paramètres du modèle sont $\phi = (\pi_1,..., \pi_K, \theta_1, ..., \theta_K)$ et la fonction de densité est :

  $$P(x, \phi) = \sum_{i=1}^{k} \pi_i P(x, \theta_i)$$

  Les méthodes de clustering probabilistes cherchent à approximer les paramètres ϕ du modèle. La probabilité d'appartenance aux clusters peut être interprétée comme un degré d'appartenance à ce cluster. On pourra notamment citer l'algorithme EM **[101]**. A noter qu'en général, les lois considérées sont supposées gaussiennes.
- Les méthodes hiérarchiques: Ces méthodes construisent une hiérarchie de clusters. Chaque nœud contient ses clusters enfants, et les nœuds frères partitionnent les objets contenus dans leurs parents. Ce type d'approche permet d'explorer les données à différents niveaux de granularité. Elles sont décomposées en deux types d'approches, les

approches ascendantes où l'algorithme part d'un grand nombre de clusters et ceux-ci sont ensuite fusionnés jusqu'à n'obtenir plus qu'un unique groupe contenant tous les objets du jeu de données et les approches descendantes qui partent, de l'ensemble des données, et le divisent en clusters qui sont ensuite divisés récursivement.

2.2.2.1.3 Les algorithmes de Clustering

Dans ce qu'il suit nous présentons quelques algorithmes de Clustering, voilà quelques exemples :

a- K-means

b- Fuzzy C-means

c- Mixture of Gaussians (Expectation maximisation)

....

a- K-means

L'algorithme k-means mis au point par McQueen en 1967**[97]**, un des plus simples algorithmes d'apprentissage non supervisé, appelée algorithme des centres mobiles **[102], [103]**, il attribue chaque point dans un cluster dont le centre (centroïde) est le plus proche. Le centre est la moyenne de tous les points dans le cluster, ses coordonnées sont la moyenne arithmétique pour chaque dimension séparément de tous les points dans le cluster c'est-à-dire chaque cluster est représentée par son centre de gravité.

i. Principe

L'idée principale est de définir les k centroïdes arbitraires c1, c2,..., ck (k le nombre de clusters fixé a priori, chaque ci représente le centre d'une classe), Ces centroïdes doivent être placés dans des emplacements différents. Donc, le meilleur choix est de les placer le plus possible éloignés les uns des autres. La prochaine étape est de prendre chaque point appartenant à l'ensemble de données et l'associer au plus proche centroïde. C'est à dire Chaque classe Si sera représentée par un ensemble d'individus les plus proches de son ci, Les nuées dynamiques sont une généralisation de ce principe, où chaque cluster est représenté par un noyau mais plus complexe qu'une moyenne.

Lorsqu'aucun point n'est en attente, la première étape est terminée et un groupage précoce est fait. À ce point nous avons besoin de recalculer les k nouveaux centroïdes mi des groupes issus de l'étape précédente qui vont remplacer les ci (mj est le centre de gravité de la classe

Sj, calculé en utilisant les nouvelles classes obtenues). Après, on réitère Le processus jusqu'à atteindre un état de stabilité où aucune amélioration n'est possible, nous pouvons constater que les k centroïdes changent leur localisation par étape jusqu'à plus de changements sont effectués. En d'autres termes les centroïdes ne bougent plus.

Algorithme :

Choisir k moyennes c_1, c_2,..., c_k initiales (par exp au hasard)

1. Répéter :

   Affectation de chaque point à son cluster le plus proche :

   $$S_i^{(t)} = \{x_i : \|x_j - m_i^{(t)}\| \leq \|x_j - m_{i*}^{(t)}\| \; for\; all\; i^* = 1, \ldots, k\} \qquad (2.4)$$

   Mettre à jour la moyenne de chaque cluster

   $$m_i^{(i+1)} = \frac{1}{|S_i^{(t)}|} \sum_{x_j \in S_i^{(t)}} x_j \qquad (2.5)$$

2. Jusqu'à : atteindre la convergence quand il n'y a plus de changement.

Fin.

ii. Discussion

Cette méthode est la plus populaire des méthodes de clustering, malgré cela, un de ses problèmes majeurs est qu'il tend à trouver des classes sphériques de même taille. En plus K-means est connu par sa complexité de « NP-difficile ». Cette approche est convergente et surtout avantageuse du point de vue calcul mais elle dépend essentiellement de la partition initiale (Des initialisations différentes peuvent mener à des clusters différents «problèmes de minima locaux ») cela risque d'obtenir une partition qui ne soit pas optimale pourtant qu'elle donne surement une partition meilleure que la partition initiale. De plus, la définition de la classe se fait à partir de son centre, qui pourrait ne pas être un individu de l'ensemble à classer, d'où le risque d'obtenir des classes vides.

b- Fuzzy C-means

i. Principe

Fuzzy C-means (FCM) est une méthode de clustering qui permet à un objet de données d'appartenir à deux ou plusieurs clusters. Cette méthode dérivée de l'algorithme C-means **[104]**, identique à l'algorithme k-means décrit précédemment, elle a été développée par Dunn

[105] en 1973 et améliorée par Bezdek **[106]** en 1981, est fréquemment utilisée dans la reconnaissance des formes. Il est basé sur la minimisation de la fonction objective suivante :

$$J_m = \sum_{i=1}^{N} \sum_{j=1}^{C} u_{ij}^{m} \|x_i - c_j\|^2 \qquad 1 \leq m < \infty \qquad (2.6)$$

Où m est un nombre réel (> 1), U_{ij} est le degré d'appartenance de xi dans le $j^{ème}$ Cluster, x_i est le $i^{ème}$ élément des données mesurées, c_j est le centre d'un cluster et $\| * \|$ est toute norme exprimant la similarité entre les données mesurées et le centre. Ce Partitionnement logique floue (fuzzy) est réalisé grâce à une optimisation itérative de la fonction objectif indiqué ci-dessus, avec la mise à jour de l'appartenance u_{ij} et les centres des clusters c_j.

On peut résumer la différence entre fuzzy C-means et k-means dans la fonction d'appartenance d'un nuage de points dans deux clusters dans l'exemple suivant :

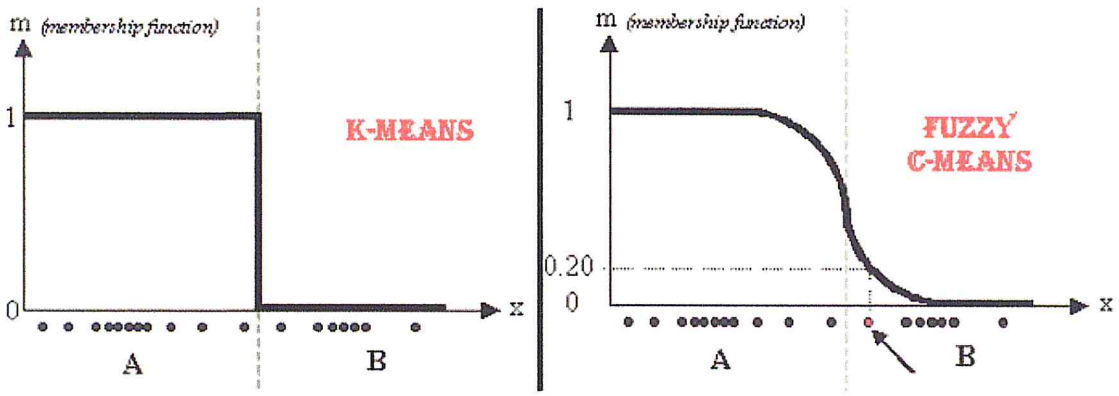


Figure 2.8 : Fonction d'appartenance dans k-means/Fuzzy C-means

Dans le cas de k-means un objet ne peut pas appartenir dans deux clusters Simultanément, ce qui explique la Discrimination binaire entre les clusters mais en FCM il est possible qu'un objet appartient à deux ou plusieurs clusters selon différents pourcentages c'est-à-dire que les données sont liées à chaque groupe par le biais d'une fonction d'appartenance, ce qui représente le comportement floue de cet algorithme. Pour le faire, nous devons simplement construire une matrice appropriée nommée U dont les facteurs sont des nombres entre 0 et 1, et représentent le degré d'appartenance entre les centres de données et des clusters.

$$U_{MC} = \begin{pmatrix} 0.8 & 0.2 \\ 0.3 & 0.7 \\ 0.6 & 0.4 \\ .. & .. \\ 0.9 & 0.1 \end{pmatrix}$$

Il est également important de noter que les initialisations différentes causent différentes évolutions de l'algorithme. En fait, il pourrait converger vers le même résultat, mais probablement avec un nombre différent d'itérations.

Algorithme

1. Initialiser U = [u_{ij}] matrice $U_{(0)}$
2. A la k-étape : calculer les centres $C_{(k)}$= [c_j] avec $U_{(k)}$

   $c_j = \sum_{i=1}^{N} u_{ij}^m . x_i \sum_{i=1}^{N} u_{ij}^m$

3. Mise à jour de $U_{(k)}$, $U_{(k+1)}$

   $u_i^{j=1} C_{k=1} \|x_i - c_j\| \|x_i - c_k\| \; 2m - 1$

4. Si $\|U_{(k+1)} - U_k\| < \varepsilon \quad (0 < \varepsilon < 1)$, alors *STOP*, sinon le retour à l'étape 2.

ii. Discussion

Une méthode que son caractère hybride (la notion de centre de gravité et la notion Floue) le rend simple, rapide. La FCM exige des paramètres d'entrées, et que la matrice de partition floue, doit être initialisée d'une manière appropriée. Ces paramètres sont choisis d'une façon arbitraire, ces paramètres ont une grande influence sur le résultat attendu. Ce qu'il nous oblige de faire une étude approprié sur les données en entrée et le regroupement que l'on souhaite obtenir. Ce type d'algorithme est fort utilisé en traitement d'images **[107]** afin d'identifier des zones similaires (contours, coins, région homogènes. . .).

c- Expectation-Maximisation (EM)

En français « L'algorithme d'espérance-maximisation », souvent abrégé par EM, Un des premiers articles sur EM a été écrit en 1958 **[108]** mais la référence pratique qui a formalisé EM et a fourni une preuve de convergence est le document de Dempster, Laird et Rubin en 1977 **[109]**, tandis que le livre de **[110]** est une autre référence populaire et très utile. Son

objectif est de trouver le maximum de vraisemblance des paramètres de modèles probabilistes (comme modèle gaussien).

i. Principe

Lorsque les seules données dont on dispose ne permettent pas l'estimation des paramètres, et/ou que l'expression de la vraisemblance est analytiquement impossible à maximiser, l'algorithme EM peut être une solution. De manière grossière et vague, il vise à fournir un estimateur lorsque cette impossibilité provient de la présence de données cachées ou manquantes ou plutôt, lorsque la connaissance de ces données rendrait possible l'estimation des paramètres.

L'algorithme EM tire son nom du fait qu'à chaque itération il opère deux étapes distinctes :

- la phase « Expectation », souvent désignée comme « l'étape E », procède comme son nom le laisse supposer à l'estimation des données inconnues, sachant les données observées et la valeur des paramètres déterminée à l'itération précédente ;
- la phase « Maximisation », ou « étape M », procède donc à la maximisation de la vraisemblance, rendue désormais possible en utilisant l'estimation des données inconnues effectuée à l'étape précédente, et met à jour la valeur du ou des paramètre(s) pour la prochaine itération. Et c'est ainsi qu'on itère l'algorithme en utilisant les paramètres trouvés à l'étape(M) pour évaluer à nouveau l'espérance.

On suppose que un échantillon X1,…Xn suit une distribution.

Exp: normale de paramètre $\Theta = (\mu, \Sigma)$

Pas de données manquantes :

$p(X|\Theta)$ la densité $\longrightarrow L(X1|\Theta) = \log(p(X|\Theta))$

Avec des données manquantes :

Variable modifiée $Z=(X, Y)$ où Y représente l'ensemble des données manquantes

$p(Z|\Theta) = p(Y|X, \Theta)\, p(X|\Theta) \longrightarrow L(X, Y|\Theta) = \log(p(Z|\Theta))$

But : Trouver la valeur espérée de la log-vraisemblance de l'ensemble complet de données Z par rapport aux données manquantes Y sachant les données observées X et les paramètres du modèle Θ

Problème : Manque de connaissance sur la variable Y, il faut donc utiliser les données et les paramètres d'une itération précédente X et $\Theta^{\wedge}(i-1)$

Valeur espérée : $Q(\Theta, \Theta^{\wedge}(i-1)) = Ey[L(X, Y|\Theta)|X, \Theta^{\wedge}(i-1)]$

ii. Discussion

EM est intéressant et très puissant (application multiple) mais pas de bonne qualité pour des séries avec une grande variance-covariance.

En effet L'algorithme EM est une méthode de clustering générale, ce qui veut dire qu'il ya des variétés de détails dans le cadre d'implémentation pour une situation donnée, en fait on compte plusieurs extensions (variantes) de cet algorithme chacune traite un domaine particulier **[111]** à fin de répondre aux problématiques qu'il rencontre EM, à titre d'exemple : GEM (Generalized EM) qui permet de simplifier le problème de l'étape maximisation ; l'algorithme CEM (Classification EM) permettant de prendre en compte l'aspect classification lors de l'estimation, ainsi que l'algorithme SEM (Stochastic EM) dont l'objectif est de réduire le risque de tomber dans un optimum local de vraisemblance. Contrairement aux approches traditionnelles de clustering cette méthode est adaptée pour traiter des grandes masses des données de grandes dimensions, pourtant elle marque quelques faiblesses en ce qui concerne l'influence des paramètres de départ (initiaux) sur les résultats attendus, ainsi que le temps de calcul…etc. alors que on peut remédier ces inconvénients en l'intégrant avec d'autres méthodes de classification (hiérarchique, k-means . . .).

2.3 Objectifs et intérêts

Les intérêts des méthodes de classification sont multiples, il peut s'agir d'améliorer les performances des moteurs de recherche documentaire ou aussi classer les documents en fonction de leurs références communes à d'autres documents pour faire apparaître les liens qui les unissent. Nous pouvons citer six applications typiques qui sont :

- Le classement automatique de différents communiqués de presse, ou messages sur des forums en différentes matières (« Les actualités de la région », « la bourse », « culture », etc..), (Exemple : Une boite propose un système de tri d'informations dans des flots de dépêches d'agence de presse AFP ou Reuters etc.. ou pages web. Chaque matin les nouvelles importantes sont faxées à différentes entreprises).
- Indexation automatique sur des catégories d'index de bibliothèques : aide à la classification thématique des différentes rédactions dans une bibliothèque.
- La gestion de bases documentaires (mémoire d'entreprise). Ce système peut être utilisé pour présenter l'information à l'utilisateur selon des catégories thématiques, ce qui facilite la navigation.

- Sauvegarde automatique de fichiers dans des répertoires.
- Les filtres internet en général, et en particulier les filtres anti-spam.
- Le classement automatique des emails, et particulièrement la redirection automatique de courriers des clients et fournisseurs en fonction de leur contenu vers les personnes compétentes dans une entreprise (Service commercial, livraison, service après-vente, approvisionnements, etc..).

2.4 Conclusion

Nous avons tenté dans ce chapitre préliminaire de définir la classification ainsi que les notions nécessaires pour l'entame de la suite de ce mémoire. Comme nous avons pu le voir dans ce chapitre qu'il y'a deux grandes approches en classification, nous avons intéressés qu'aux techniques de classification non supervisée (Clustering). Les méthodes de clustering comme toutes les autres méthodes de classification, ont leurs avantages, faiblesses (voir section : discussion).

Cependant, il n'y a pas que le type statistique, il y'en a d'autre type qui s'appuie sur la théorie de probabilité. Dans le chapitre suivant nous allons présenter une nouvelle méthode de conception qu'on va l'utiliser dans notre système de résumé automatique multi-documents, basée sur le modèle probabiliste et qui est très connu pour son efficacité. Une description détaillée sera accordée ultérieurement.

Chapitre 3

Conception

3.1 Introduction

Depuis quelques années déjà, les recherches se concentrent beaucoup plus sur le résumé multi-document que sur le résumé mono-document. Nous nous focalisons dans cette section sur le résumé multi-document statique, point central de notre attention dans ce travail.

Après avoir éclairci les différents systèmes de classification nous allons présenter en premier lieu notre système pour pouvoir spécifier exactement notre contribution par rapport à l'utilisation de classification rappelons que ce travail est focalisé principalement sur le Clustering, ensuite nous allons décrire notre approche d'une façon générale suivi par l'algorithme d'apprentissage et classification (Naïve Bayes) utilisé, ainsi qu'un algorithme FREACCA (Fuzzy Relational Eigenvector Centrality-based Clustering Algorithm) basé sur l'approche floue du Clustering (Fuzzy C-means) qui va être détaillé par la suite.

3.2 Problématique

Dans cette étude, on s'intéresse à la génération des résumés multi-documents. L'utilisateur choisis un corpus de documents, on essaye de générer automatiquement un résumé qui récapitule les points importants au thème qui se trouvent dans le contenu des documents.

Pour ce faire, on va proposer une solution de résumé automatique définie par l'architecture suivante :

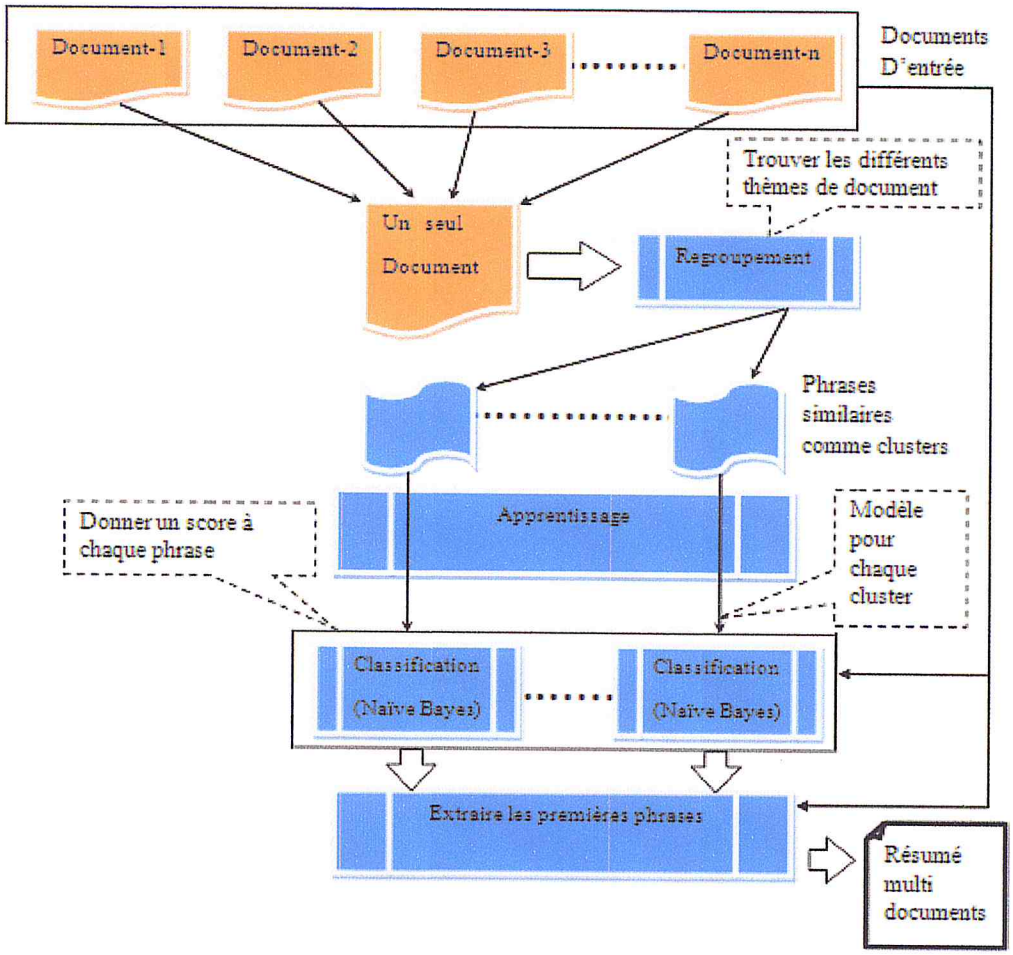


Figure 3.9 : Architecture de notre système de résumé multi-documents statique

3.3 Description des étapes de notre système

Pour gérer notre système de résumé multi-document nous fusionnons simplement tous les documents textes d'entrée dans le même thème et les considérer comme un seul document. Comme tous les systèmes de résumé automatique on a besoin d'une phase de pré-traitement, ainsi, une phase de post-traitement et une phase de traitement qui définit la différence entre un système de résumé automatique et un autre. Notre système ne fait pas l'exception par rapport aux autres systèmes, il comporte aussi ces trois modules classiques, pour cela un ensemble d'opérations préliminaires doivent être faites pour épurer le texte de tous les mots inutiles et conserver seulement ceux qui sont porteurs d'informations et utiles pour le processus de classification.

Afin de faciliter la compréhension de notre démarche on va spécifier les étapes que nous avons menées par rapport à l'architecture précédente.

3.3.1 Étape 1 : Extraction de texte

Le but de cette étape est de lire tous les documents introduit ou choisit par l'utilisateur afin d'extraire tous les contenus de ces derniers et les regrouper en un seul document ou d'une autre façon en un seul texte ensuite on les traiter comme un seul document (**Many In One Processing**).

Pour mieux voir les choses on a représenté notre méthode sur la figure ci-dessous (**figure 3.10**) :

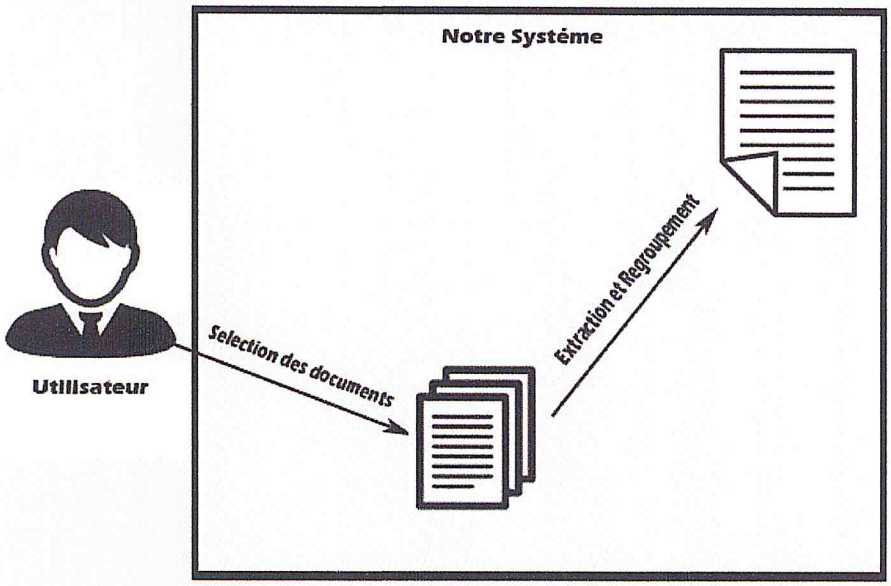


Figure 3.10: Méthode d'extraction de texte

3.3.2 Étape 2 : Pré-traitement

Ceci pour rendre le texte d'entrée conforme à la phase de traitement, c'est la partie dépendante de la langue, qui peut être trouvé dans de nombreux travaux de récupération d'informations (IR). Nous avons abordé les différentes méthodes de représentation des documents. Il est donc nécessaire d'effectuer, au préalable dans un espace de mots, une transformation permettant le passage de l'espace du caractère à un espace de mots. Toutes ces transformations et méthodes font partie de ce qu'on appelle le prétraitement. Plusieurs d'entre elles sont spécifiques à la langue des documents (on ne fait pas le même type de prétraitement pour des documents écrits en anglais qu'en français).

Dans notre système, nous sommes intéressés à quatre tâches de Prétraitement (voir **la figure 3.11**) :

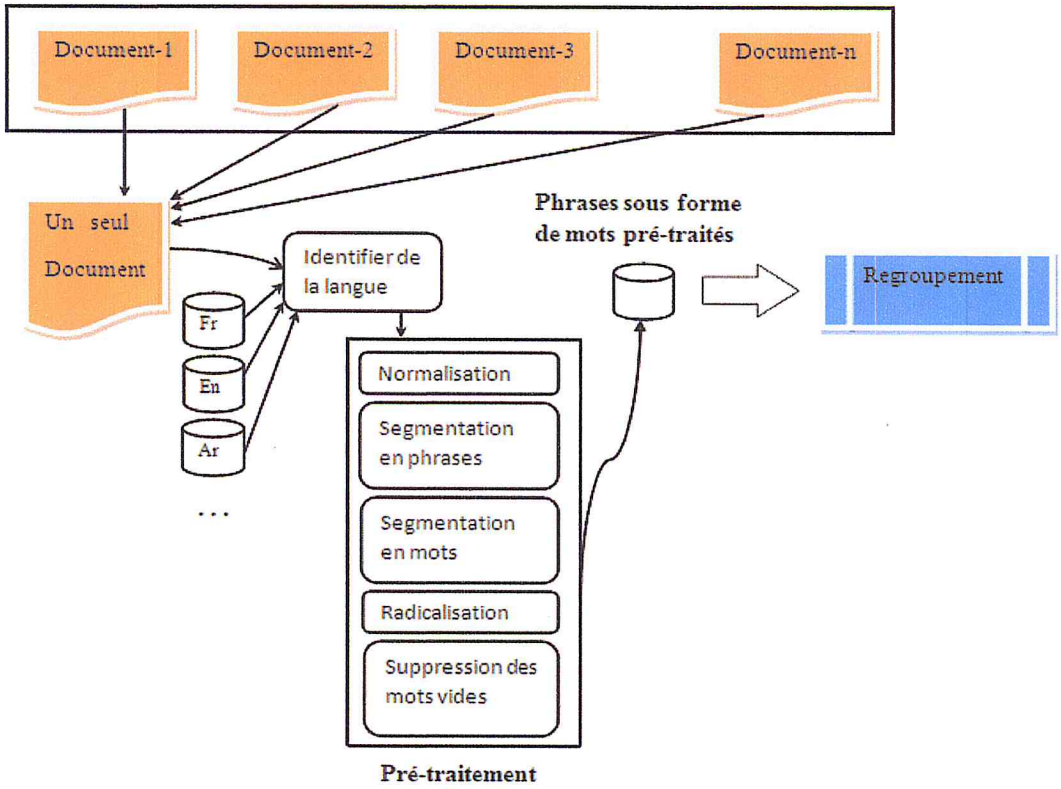


Figure 3.11 : Phase de Pré-traitement

1. **Normalisation :** Dans cette étape nous pouvons supprimer les caractères spéciaux et les chiffres.
2. **Segmentation :** c'est la reconnaissance des termes utilisés. Elle consiste à découper la séquence des caractères afin de regrouper les caractères formant un même mot. Elle inclut deux fonctions : segmentation de texte en phrases et segmentation de texte en mots (Tokenization).
3. **Radicalisation :** Le but de cette tâche est de supprimer les suffixes et préfixes alors nous pouvons obtenir le radicale d'un mot.
4. **Suppression des mots vides :** Nous avons l'utilisé pour enlever les mots vides, se sont les mots qui n'ajoutent aucune signification au texte. Nous avons utilisé des listes de mots disponibles sur le web pour chaque langue.

   Dans la figure ci-dessous (**figure 3.12**) on présente quelques outils pour la phase prétraitement.

Tâche	Outils	Langues
Segmentation des phrases	openNLP	Nl, En, De, It, Pt, Th
	JHazm	Fa
	Regex	Le reste
Segmentation des mots	openNlp	Nl, En, De, It, Pt, Th
	Lucene	Zh, Ja
	Regex	Le reste
Radicalisation	Shereen Khoja	Ar
	JHazm	Fa
	HebMorph	He
	Lucene	Bg, Cs, El, Hi, Id, Ja, No
	Snowball	Eu, Ca, Nl, En (Porter), Fi, Fr, De, Hu, It, Pt, Ro, Ru, Es, Sv, Tr
	/	Le reste

Figure 3.12 : Les outils utilisés pour prétraiter chaque langue

3.3.3 Étape 3 : Le traitement (Processing)

Cette étape est considérée comme l'étape principale de notre travail, puisque c'est là où on trouve les 3 phases tel que le regroupement ou aussi la classification (Clustering), la phase d'apprentissage et finalement la phase de calcule du score des phrases.

3.3.3.1 Le Regroupement (Clustering)

Chaque texte contient de nombreux thèmes où un thème est un ensemble de phrases ayant certaines relations entre eux. Cette relation est la similarité cosinus entre deux phrases, ici on assume que les phrases qui ont beaucoup termes en commun sont des phrases de même thème. Ainsi on regroupe les phrases similaires dans le même groupe (cluster), ensuite on les traite comme un seul texte. Le document créé est ensuite résumé en extrayant une ou plusieurs phrases du contenu qu'il contient. Pour cela nous utilisant un algorithme simple qui utilise une similarité cosinus et un seuil de clustering nommé *Threshold* pour regrouper les *n* phrases (voir **la figure 3.13**). L'équation 3.1 est utilisée pour calculer la similarité Cosinus entre deux phrases X et Y :

$$\cos(X,Y) = \frac{\sum_i x_i.y_i}{\sqrt{\sum_i (x_i)^2}.\sqrt{\sum_i (y_i)^2}} \qquad (3.1)$$

Où : xi est le mot numéro i dans la phrase X, et yi est le mot numéro i dans la phrase Y.

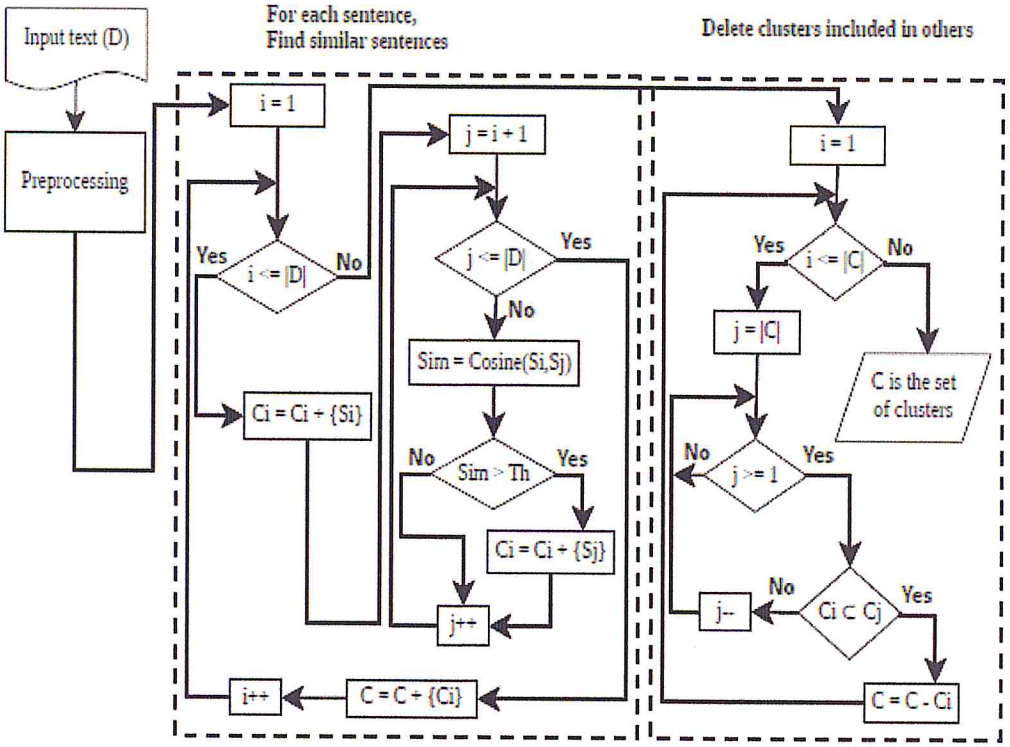


Figure 3.13 : Méthode de Clustering

Où :

- S : phrase.
- C : un ensemble de clusters.
- $|D|$: nombre de différents termes dans un document D, n : nombre de phrases dans ce document.

Notre algorithme de regroupement passe par les étapes suivantes :

- Pour chaque phrase, on calcule sa similarité avec toutes les autres phrases en utilisant l'équation (3.1).
- Ensuite, on utilise un seuil Th pour sélectionner les phrases les plus similaires.
- Chaque phrase et celles qui lui sont similaires vont construire un cluster.
- Nous exécutons les trois premières étapes jusqu'à ce qu'il ne reste aucune phrase.
- Après avoir terminé toutes les n phrases, nous obtenons n clusters (une phrase peut appartenir à plusieurs clusters).
- Pour chaque cluster, on vérifie s'il n'est pas inclut dans un autre cluster. Sinon, on l'efface.
- Enfin, on obtient un nombre total de clusters inférieur ou égal au nombre de phrases.

3.3.3.2 L'apprentissage

Puisque l'approche manuelle de classification de textes est coûteuse en temps de travail, peu générique, et relativement peu efficace, l'autre solution a été admise, qui consiste à faire apprendre automatiquement à l'ordinateur, sur la base d'un corpus de textes qui servent d'exemples, les paramètres de la fonction de classement. La notion d'apprentissage introduit le fait d'apprendre un ensemble de relations entre les critères caractérisant l'élément à classer et sa classe cible. Les algorithmes de classification avec apprentissage ont recours à un ensemble d'exemples afin d'apprendre ces relations.

Cette étape est super essentielle pour continuer notre traitement, on utilise des critères pour connaitre l'importance des phrases obtenues depuis la phase du regroupement. Pour cela on utilise l'équation mathématique suivante :

$$P_f(f = \emptyset | C_j) = \frac{|\emptyset \in C_j|}{\sum_{C_l \in C} |\emptyset' \in C_l|}$$

Où :

- f : Critère de sélection.
- $\emptyset$: Observation d'un critère f dans le cluster C.
- C : Ensemble des clusters.
- $|\emptyset \in C_j|$: Nombre d'apparition de $\emptyset$ dans le cluster C

On compte 4 critères de sélection :

a- Fréquence des termes (Unigram) (TFU) :

Ce critère est utilisé pour calculer la pertinence de la phrase selon ses termes. Chaque terme est considéré comme une catégorie.

b- Fréquence des termes (Bigram) (TFB) :

Ce critère est similaire à la fréquence des termes unigrammes mais au lieu d'un terme que nous utilisons deux termes consécutifs.

c- Position de la phrase (Pos) :

Nous voulons utiliser les positions de phrases dans le texte original comme un critère.

d- Longueur de la phrase (Rleng, Pleng) :

Un autre critère appliqué dans notre système est la longueur de phrase (nombre de mots), où les phrases dont la taille est petite sont toujours défavorisées et dont n'ont pas une grande chance contre celles de grande taille. Suivant la longueur d'une phrase, nous pouvons la mettre dans l'un des trois catégories : des phrases avec une longueur inférieure à 6 mots, ceux ayant une longueur de plus de 20 mots, et ceux ayant une longueur entre **[80]**.

Comme une position d'une phrase, trois catégories est un petit nombre. Par conséquent, nous avons utilisé chaque longueur comme une catégorie. Supposons que nous avons 4 phrases dont les longueurs sont: 5, 6, 5 et 7, alors nous aurons 3 catégories de longueurs: 5, 6 et 7. Dans notre travail, nous utilisons deux types de la longueur d'une phrase :

- **Longueur réelle (RLeng) :** qui est la longueur de la phrase sans enlever les stop-words.
- **Longueur pré-traitée (Pleng) :** qui est la longueur de la phrase après le pré-traitement.

3.3.3.3 Le Score des phrases (Scoring) en utilisant la classification

C'est la phase finale de l'étape traitement, dans cette phase on calcule le score des phrases grâce à les résultats obtenus depuis la phase précédente (l'apprentissage). Nous voulons trouver les phrases qui sont les plus probables pour représenter tous les thèmes (clusters) $C_j \in C$ en utilisant un ensemble de critères $f_k \in F$.

Pour calculer la probabilité qu'une phrase s_i appartient à une classe C_j sachant un ensemble de critères F (fréquence de mots, longueur de phrase, position de phrase, etc.), nous avons utilisé la classification de Naïve Bayes, le nom Naïve Bayes découle du fait que l'algorithme utilise le théorème de Bayes. Si nous supposons l'indépendance entre les différentes classes et les différents critères (une phrase peut avoir plusieurs classes), en utilisant le théorème de Bayes, on obtient l'équation suivante :

$$P(s_i \in \bigcap_j C_j \,|F) = \prod_j \prod_k P(f_k \,|s_i \in C_j)$$

Pour faciliter la combinaison des différents critères d'une phrase, en particulier ceux de la fréquence du terme, nous proposons d'utiliser un score plutôt qu'une probabilité. Une autre

raison est que la classification utilisée ici n'est qu'un outil de score (donner des scores à une phrase dans chaque cluster). Alors, l'équation précédente va être écrite comme suit :

$$Score(s_i, \bigcap_j C_j, F) = \prod_j \prod_k Score(s_i, C_j, f_k)$$

Enfin, les phrases vont être réorganisées selon leurs scores. La fonction $Score(s, C, f)$ est utilisée pour calculer le score de la phrase s dans la classe C, en utilisant le critère f qui peut apparaître plusieurs fois dans cette phrase.

Le score d'une phrase s dans une classe C en utilisant un critère f , peut être représenté comme la somme des probabilités des observations de f . Ensuite, on ajoute un à la somme pour éviter la multiplication par un score d'un critère égal à zéro.

$$Score(s_i, C_j, f_k) = 1 + \sum_{\emptyset \in s_i} P(f_k = \emptyset | s_i \in C_j)$$

Où $\emptyset$ est l'observation de critère f dans une phrase s_i . Par exemple, en supposant le critère f_k est la fréquence de mots, et nous avons la phrase : "*I am studying at home*". La phrase après l'étape de pré-traitement serait : s_1 = {"studi" (le radical de "study"), "home"}. Alors $\emptyset$ peut être "studi" ou "home", ou n'importe quel autre terme.

3.3.4 Etape 4 : Extraction du Résumé

C'est la dernière étape de notre travail, elle contient au fait 2 phases qui sont le ré-ordonnancement et l'extraction du résumé.

3.3.4.1 Le Ré-Ordonnancement (Post Processing)

Dans cette phase on utilise les résultats de la deuxième étape du résumé automatique (Traitement), grâce aux scores obtenus depuis la phase (Scoring) on fait un ré-ordonnancement sur les phrases obtenues.

La figure ci-dessous représente brièvement le fonctionnement de cette phase (**figure 3.14**) :

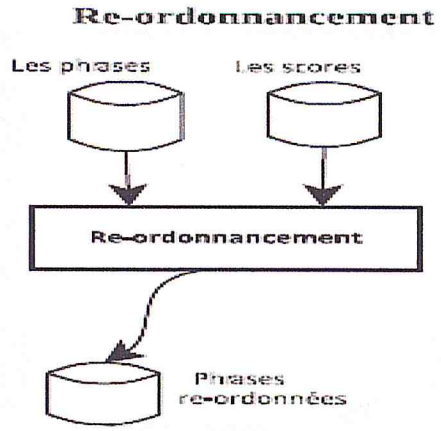


Figure 3.14 : Phase de Ré-Ordonnancement

3.3.4.2 Génération du Résumé :

C'est la deuxième phase de l'étape extraction du résumé et la dernière phase de notre travail, dans cette phrase on utilise un algorithme assez simple pour générer notre résumé.

La figure ci-dessous représente l'organigramme de notre algorithme (**figure 3.15**) :

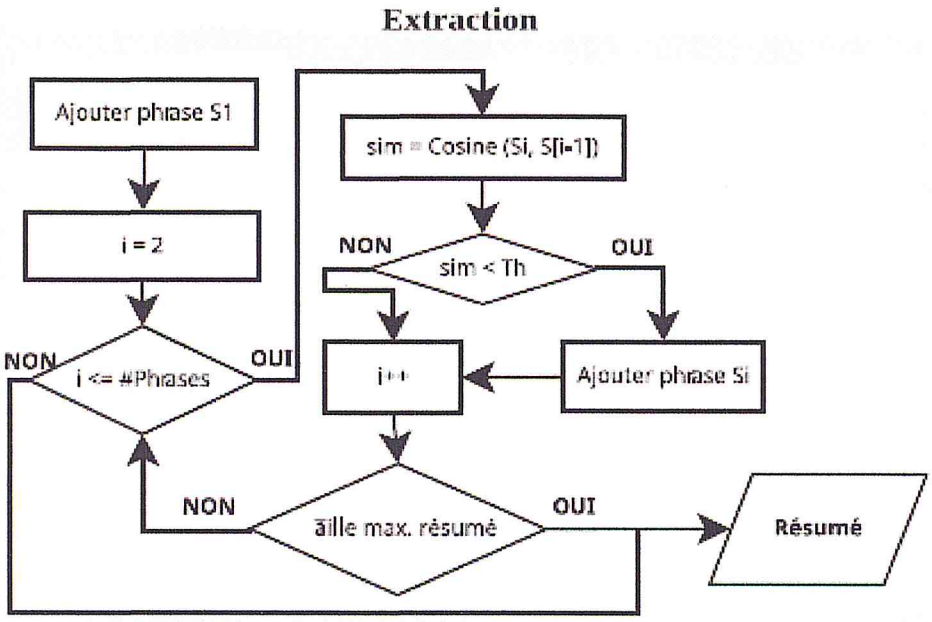


Figure 3.15 : Phase de Génération de résumé

Ces deux phases combinées constituent notre étape d'extraction du résumé pour en finir, la figure ci-dessous (**figure 3.16**) représente la combinaison des deux phases :

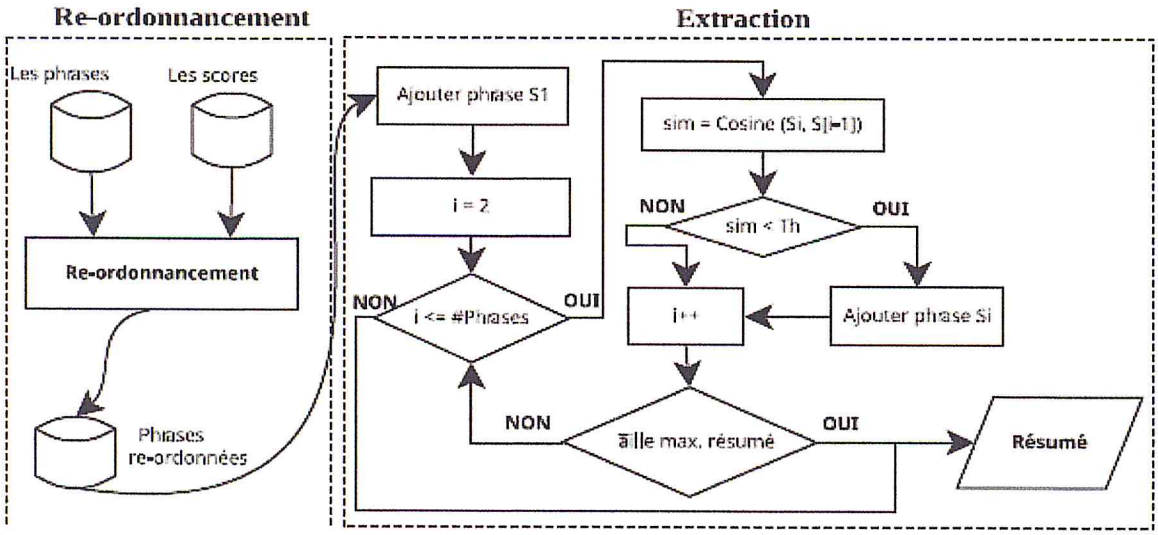


Figure 3.16 : Extraction du résumé

Les étapes suivantes décrivent l'algorithme d'extraction du résumé :

- Étant donné l'ensemble des phrases ordonnées par leurs scores, ajouter la première phrase au résumé.
- Pour chaque phrase candidate, on vérifie si le résumé a atteint sa limite. Si c'est le cas, on s'arrête ; Sinon, on continue.
- On calcule la similarité entre la dernière phrase ajoutée au résumé et la phrase candidate. Si la similarité est inférieure à certaine valeur, on l'ajoute au résumé. Sinon, on passe à la phrase suivante.

3.4 Description de notre deuxième approche proposée

Nous avons vu précédemment dans le chapitre 2 que plusieurs techniques existent pour le Clustering dont un algorithme de l'approche de Clustering flou (Fuzzy C-means), cette section présente notre méthode cible de classification non supervisée basée sur l'approche relationnelle du Clustering flou. Pour cela et pour mener à bien notre expérimentation nous avons choisi d'utiliser différemment par rapport à la méthode Naïve Bayes un algorithme qui s'appelle FRECCA (*Fuzzy Relational Eigenvector Centrality-based Clustering Algorithm*) proposé par Andrew Skabar and Khaled Abdalgader **[112]**.

3.4.1 Définition

FRECCA est basé sur une approche du modèle de mélange, où les données sont modélisées comme une combinaison des composants (clusters). L'approche du modèle de mélange utilise une fonction de vraisemblance. FRECCA travaille sur des données relationnelles ce qui veut dire, des données sous forme d'une matrice carré ayant une similarité entre les paires d'objets. Il utilise un graphe de centralité comme une vraisemblance (la centralité veut dire le sommet le plus important dans un graphe). Ici l'algorithme de PageRank **[113]** est utilisé pour calculer la centralité où il est appliqué à chaque cluster et la valeur de PageRank d'un objet est interprétée comme une vraisemblance et puis l'algorithme d'Expectation Maximisation (EM) **[114]** est utilisé pour calculer les valeurs d'appartenance au cluster et les coefficients de mélange, FRECCA utilise l'EM pour optimiser ces paramètres. Les valeurs finales d'appartenance au cluster sont obtenues en tant que sortie de l'algorithme. Afin de bien comprendre les algorithmes utilisés on donne quelques notions :

Un algorithme PageRank ou *PR* qui est à l'origine proposé pour le classement (*raking* en anglais) des pages web, il a été utilisé par le moteur de recherche Google pour classer les sites web. Il est utilisé pour mesurer l'importance des pages web, il détermine l'importance des sites web en comptant le nombre et la qualité des liens vers les pages web. Le web peut être vu comme un graphe dirigé G où les sommets sont les pages web et les arcs sont les hyperliens entre les pages, cela est résumé par l'équation suivante :

$$PR(V_i) = (1-d) + d \times \sum_{j \in In(V_i)} \frac{1}{|Out(V_j)|} PR(V_j) \qquad (3.2)$$

Ici $PR(V_i)$ est la valeur de Page Rank de sommet V_i , $Out(V_j)$ est les liens sortants de V_j vers V_i , d est un facteur qui est fixé à 0.80 **[113]**. Actuellement le PageRank a été proposé pour classer les pages web mais il peut également utiliser pour déterminer l'importance d'un terme ou des phrases dans un cluster (document dans notre cas) **[84]**.

Expectation-Maximisation (EM) : qui calcule les probabilités d'appartenance au cluster. Cet algorithme est divisé en E-étape qui estime les valeurs manquantes de l'estimation actuelle et M-étape trouve les nouvelles estimations pour les paramètres qui sont maximisés

par les estimations des données manquantes. (C'est un algorithme itératif comme nous l'avons parlé au chapitre 2).

3.4.2 Les bases de notre méthodologie

En ce qui concerne notre travail, nous avons appliqué l'algorithme FRECCA dans le cas d'un résumé mono-document, où on a comme entrée une matrice de similarité entre chaque paire de phrases et un nombre de clusters :

- Pour estimer le nombre de cluster, on utilise soit la racine carrée du nombre total des phrases de tous les textes soit un pourcentage (ex 30 %) de la moyenne du nombre de phrases par texte.
- Pour calculer la similarité entre deux phrases, on a choisi d'utiliser la similarité cosinus (voir l'équation 3.1).

L'idée générale est de classer les phrases suivant leurs valeurs d'appartenance aux clusters selon les étapes principales suivantes :

1. **Initialisation et Normalisation :** Ici on a initialisés les valeurs d'appartenance des phrases aux clusters aléatoirement.

2. **Étape d'Expectation :** Dans cette étape on a calculé la valeur de Page Rank (PR) pour chaque phrase dans chaque cluster à l'aide de l'équation précédente 3.2, au lieu d'utiliser des pages on a utilisé des phrases où le lien entre eux est la mesure de similarité cosinus. Soient i et j deux phrases et C un cluster ; la valeur de PR pour la phrase i dans C augmente à chaque fois que le lien entre la phrase i et la phrase j dans C est fort en même temps que le lien entre la phrase j et les autres phrases dans C est faible, mais quand le lien entre cette dernière et les autres phrases est fort la valeur de PR pour la phrase i dans C diminue.

3. **Étape de Maximisation :** On a utilisé les valeurs calculées à l'étape d'Expectation afin de mettre à jour les valeurs d'appartenance au cluster qui sont initialisées aléatoirement.

À la fin, les phrases avec des valeurs d'appartenance au cluster les plus élevées peuvent être considérées comme un élément central du thème et donc ces phrases sont extraites pour générer notre résumé mono-document. On représente ci-dessous un schéma qui résume les différentes étapes qu'on a suivies (voir **la figure 3.17**).

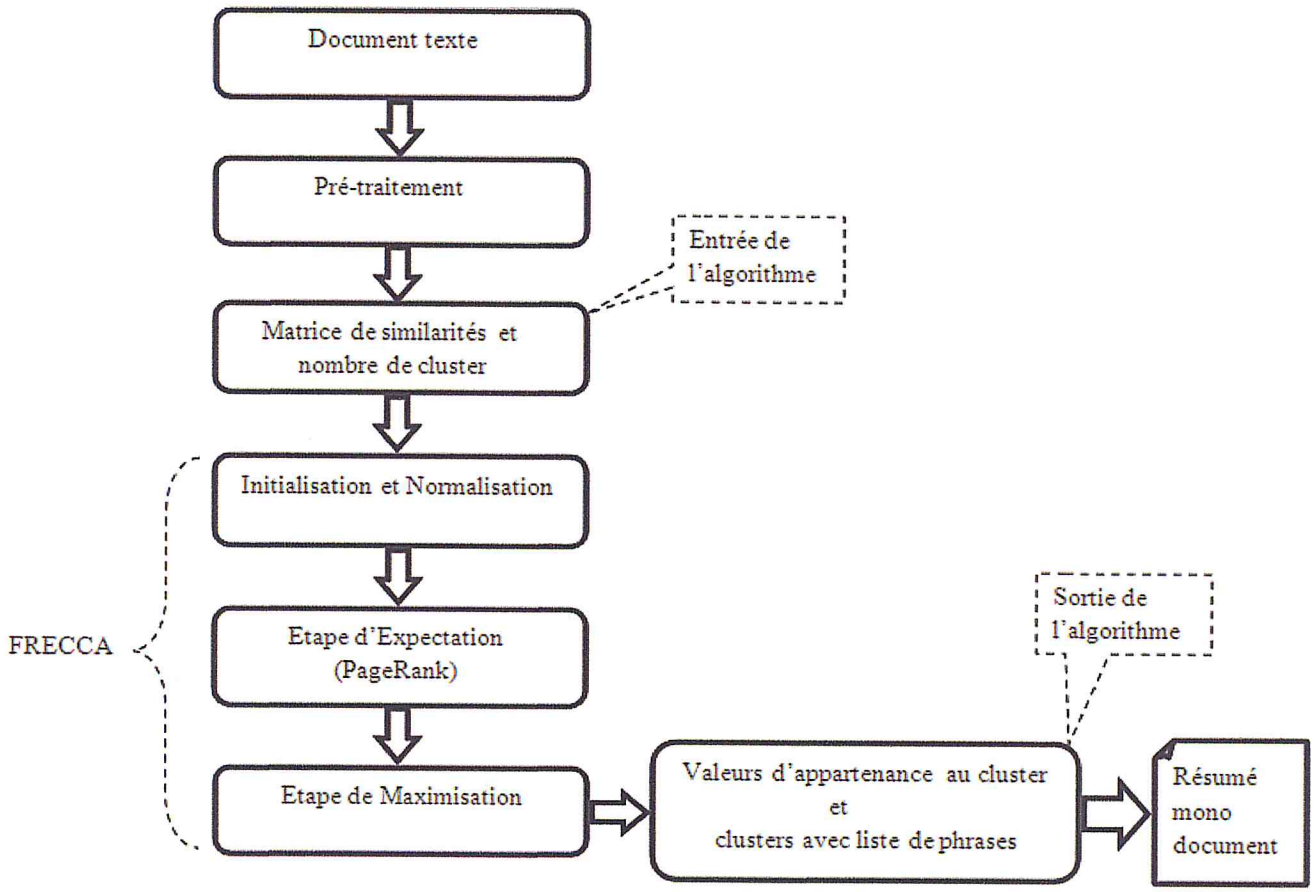


Figure 3.17 : Schéma de notre méthode de Clustering en utilisant FRECCA.

3.5 Conclusion

Dans ce chapitre nous avons détaillé et épuré notre système de résumé automatique et grâce à la conception, on a pu expliquer brièvement son fonctionnement en spécifiant les différentes étapes exécutées pour réaliser ce dernier ainsi que l'approche de Clustering proposée.

Dans le chapitre suivant nous allons présenter le côté pratique de notre application.

Chapitre 4

Réalisation

4.1 Introduction

Il est évident que les méthodes et les outils choisis pour concevoir et développer une application doivent être en fonction de l'environnement et du domaine d'application de celle-ci. Cela est bien expliqué par le génie logiciel.

Ce chapitre nous permet de montrer les résultats de notre application dans sa phase de test et d'essai, ce qui nous permettra d'envisager les améliorations possibles. Les difficultés rencontrées nous ont permis de dégager un ensemble de pistes susceptibles de nous aider.

4.2 L'Environnement de travail

Pour que notre travail atteigne l'objectif qu'on visait, on a pris l'initiative d'exploiter et d'implémenter notre programme sur la version : Windows 7 Ultimate Edition SP1 64 BITS et afin de mener à bien ce projet, nous avons utilisé un ensemble de matériels dont les principales caractéristiques sont les suivantes :

CPU: Intel® Core™ i3-2120 CPU @ 3.30 GHz
RAM: 4 GO
DISQUE DUR : Capacité 1 TO

Figure 4.18 : Matériel informatique utilisé

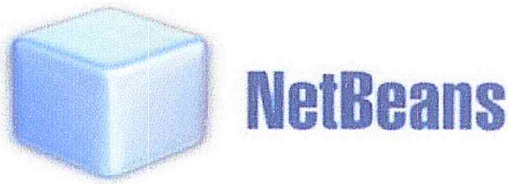
4.3 Langage de programmation

Notre application a été codée en sa globalité par le langage JAVA néanmoins il est possible de l'utiliser dans d'autres langages comme Python, C++, etc. Le Java est un langage de programmation orienté objet, développé par Sun Microsystems et lancé le 23 mai 1995. Ecrit par James Gosling, Il permet de créer des logiciels compatibles avec de nombreux systèmes d'exploitation (Windows, Linux, Macintosh, Solaris). Java donne aussi la possibilité de développer des programmes pour téléphones portables et assistants personnels. Enfin, ce langage peut être utilisé sur internet pour des petites applications intégrées à la page web (applet) ou encore comme langage serveur.

4.3.1 IDE : NetBeans

C'est un environnement de développement intégré (IDE) pour Java, placé en open source par Sun en juin 2000 sous licence CDDL (Common Development and Distribution License). En plus de Java, NetBeans permet également de supporter différents autres langages, comme Python, C, C++, XML et HTML. Il comprend toutes les caractéristiques d'un IDE moderne (éditeur en couleur, projets multi-langage, refactoring, éditeur graphique d'interfaces et de pages web).

NetBeans est disponible sous Windows, Linux, Solaris (sur x86 et SPARC), Mac OS X et Open.



4.4 Description de notre application

Dans cette partie nous allons présenter l'interface principale de notre application avec son fonctionnement.

4.4.1 Architecture de fonctionnement

Nous avons résumé le fonctionnement de notre application dans le schéma de la figure ci-dessous (**figure 4.19**) :

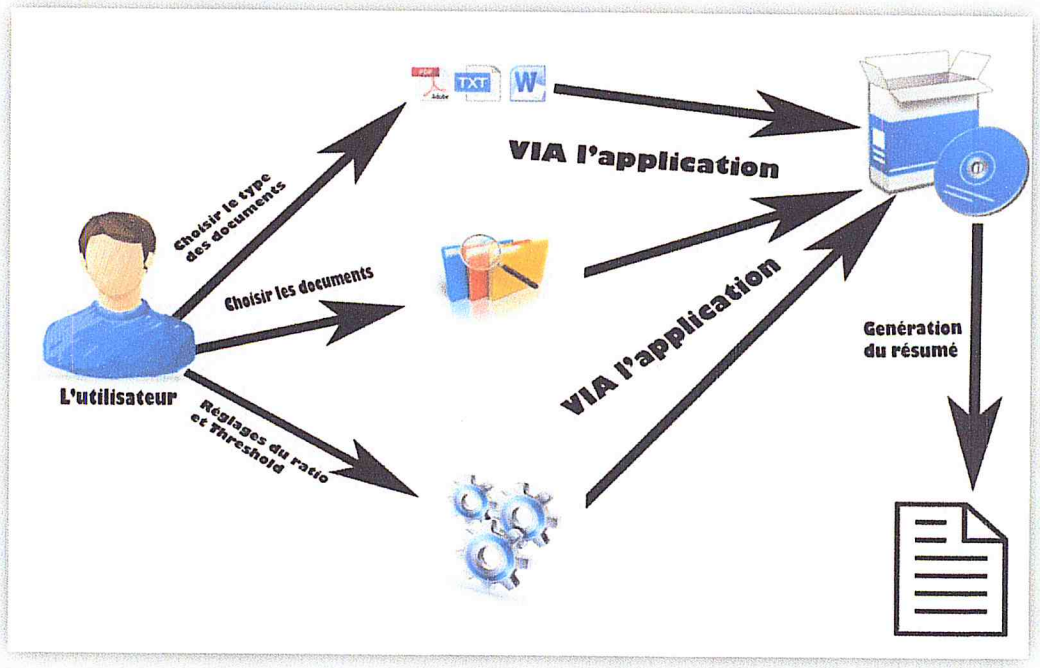


Figure 4.19 : Architecture de fonctionnement du système Multiple Summarizer

4.4.2 Interface utilisateur

Dans cette partie on va expliquer un peu l'interface de notre application, comme les fonctionnalités des boutons et le rôle de chaque outil à l'aide de la figure ci-dessous (**figure 4.20**) :

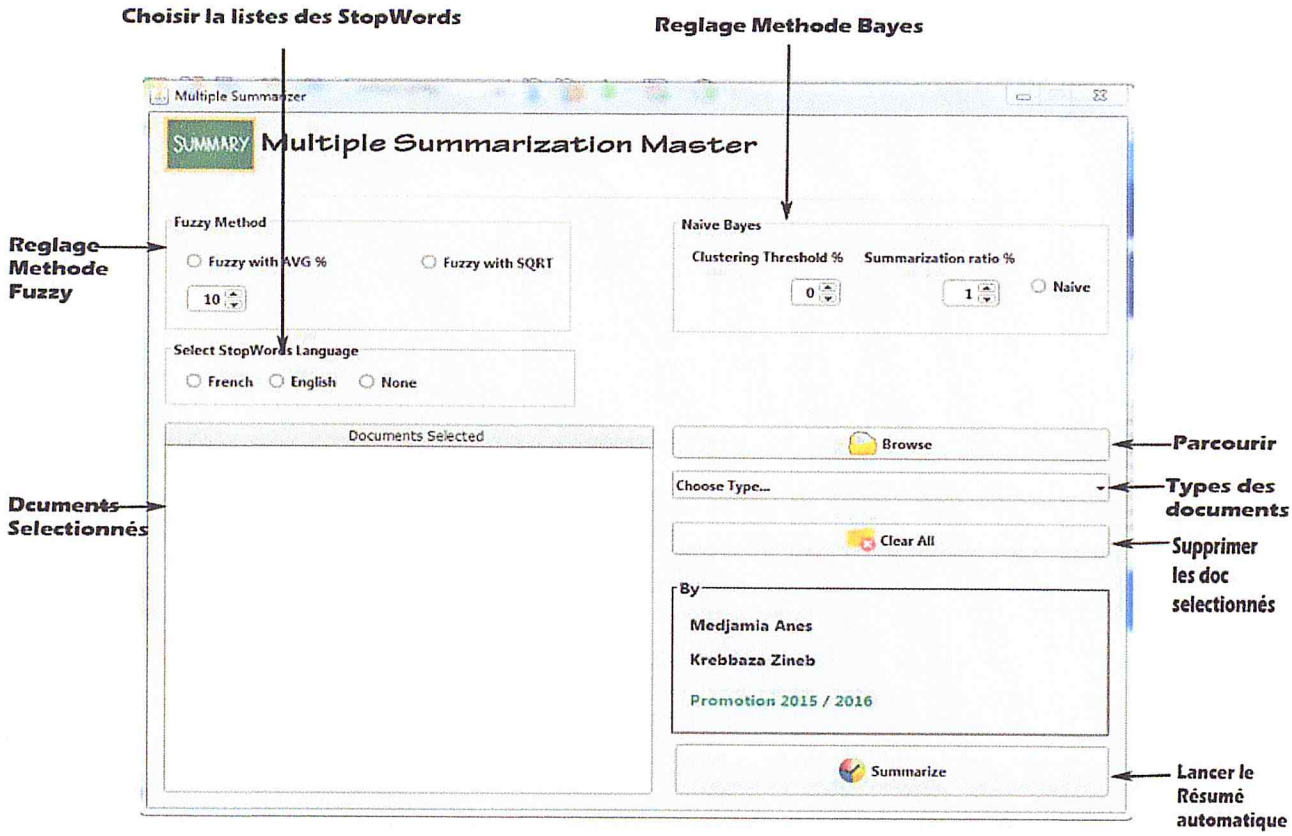


Figure 4.20 : Interface de l'application

- **Liste des documents** : Un tableau qui affiche les documents sélectionnés par l'utilisateur.
- **Liste des types** : L'utilisateur doit sélectionner le type des documents avant de choisir ces derniers.
- **Bouton Browse** : Ouvre une interface dialogue où l'utilisateur peut parcourir le répertoire de son système et choisir les documents à résumé.
- **Clear ALL** : L'utilisateur peut supprimer ou annuler la sélection des documents en cas d'erreur de sélection.
- **Paramètres Fuzzy (AVG, SQRT)** : L'utilisateur peut choisir entre la moyenne et la racine carrée.
- **Paramètres Naïve Bayes (Ratio, Threshold) :** L'utilisateur peut choisir librement la taille du résumé qui veut obtenir et la mesure de classification.
- **Choisir la liste des stops Word :** L'utilisateur choisit la liste des stops Word qui veut l'utiliser pour les différentes langues (anglais et français).

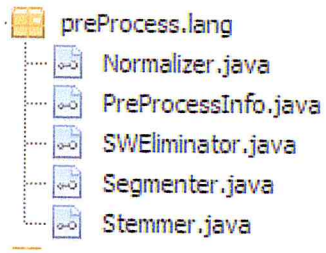
4.4.3 Diagramme de paquets

Dans cette partie on va parler un peu de la structure de nos paquets et nos classes utilisées pour concevoir cette application. Donc on va juste introduire les paquets essentiels en suivant le cheminement des étapes du résumé automatique.

4.4.3.1 Prétraitement

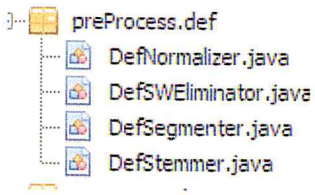
On a réalisé 3 paquets pour établir cette phase :

1. PreProcess.lang



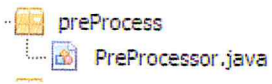
Ce premier paquet contient l'interface qu'on a besoin d'implémenter pour concevoir notre prétraitement. Comme on le voit chaque interface définie une étape de la phase du prétraitement. (Normalizer, Segmenter …etc.).

2. PreProcess.def



Ce paquet contient la classe qui implémente l'interface définie précédemment.

3. PreProcess



Ce paquet contient la classe finale qui est responsable de l'exécution de notre prétraitement, elle fait appel aux autres classes définies dans le 2éme paquet.

Comme le montre la figure ci-dessous (**figure 4.21**) :

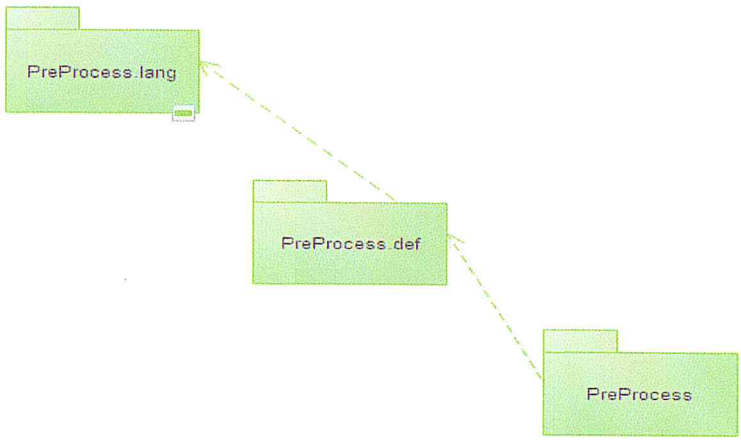
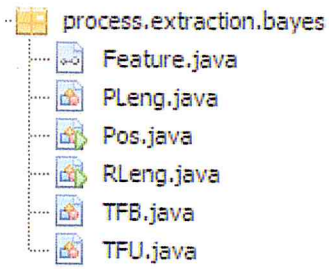


Figure 4.21 : Diagramme de paquet « prétraitement »

4.4.3.2 L'extraction du résumé

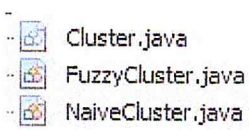
Dans cette étape on a aussi élaboré 3 paquets :

1. Process.extraction.bayes



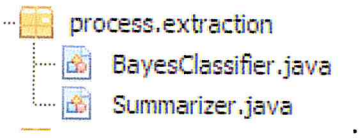
Dans ce paquet on trouve toutes les mesures utilisées pour établir la phase de classification (les mesures de base).

2. Process.extraction.cluster



Comme on le voit ce paquet contient deux classes pour le type de classification, afin d'élaborer les 2 méthodes de classification bayésienne et la classification fuzzy (bayes clustering & fuzzy clustring).

3. Process.reducation



Dans ce paquet on trouve les deux classes qui font appel au paquet précèdent pour en finir avec la phase de classification.

Comme le montre la figure ci-dessous (**figure 4.22**) :

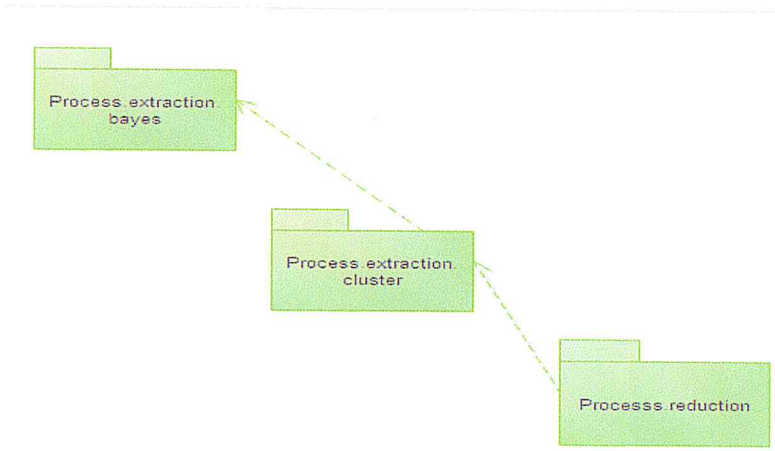
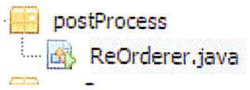


Figure 4.22 : Diagramme de paquet « Classification & extraction »

4.4.3.3 Post Traitement

Dans cette phase on a un seul paquet qui contient la class du ré-ordonnancement des phrases afin de générer notre résumé.



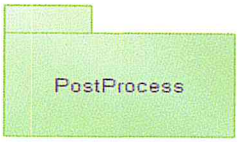


Figure 4.23 : Diagramme de paquet « Post Traitement »

4.4.4 Les bibliothèques java utilisées

On a utilisé juste 3 types de bibliothèques pour traiter les documents de type PDF, WORD et TXT, comme le montre la figure ci-dessous :

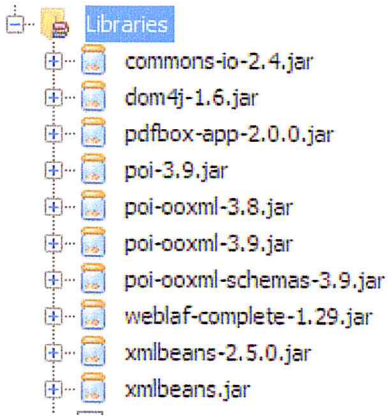


Figure 4.24 : Bibliothèque java

4.5 Expérimentation

Nous allons présenter les différentes expérimentations conduites. Premièrement, nous allons faire un test par rapport aux paramètres que nous avons utilisés afin de générer notre système. Ensuite nous allons évaluer nos approches en termes de qualité.

4.5.1 Évaluation de paramètre (naïve bayes)

Dans cette partie on va faire des tests qui vont permettre de comparer les résultats obtenus en fonction du ratio et du Threshold, aussi pour assurer que la réalisation de logiciel correspond aux besoins des utilisateurs et d'atteindre le niveau de qualité souhaitée. Pour en faire on a préparé 2 corpus de documents, un en français et l'autre en anglais, on a choisi le sujet « programmation » pour le corpus français et le sujet « *software* » pour le corpus anglais, les corpus ont différents formats de documents (PDF, TXT et DOC).

Comme le montre la figure ci-dessous (**figure 4.25**) :

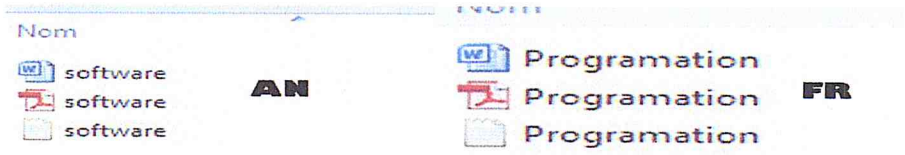


Figure 4.25 : Corpus de test Anglais & Français

Nous allons commencer avec le corpus en français, en testant avec différentes mesures pour le ratio et le Threshold, ensuite nous allons voir comment va être la qualité de notre résumé en pourcentage (%). Le tableau ci-dessous représente les résultats de nos tests (**figure 4.26**) :

Threshold	Ratio	Qualité de Résumé
20	50	30%
45	35	40%
60	25	55%
80	18	65%
100	10	75%

Figure 4.26 : Résultats de tests pour le corpus français

On a transformé le tableau en 2 représentations graphiques en courbe lisse comme le montre la figure ci-dessous (**figure 4.27**) :

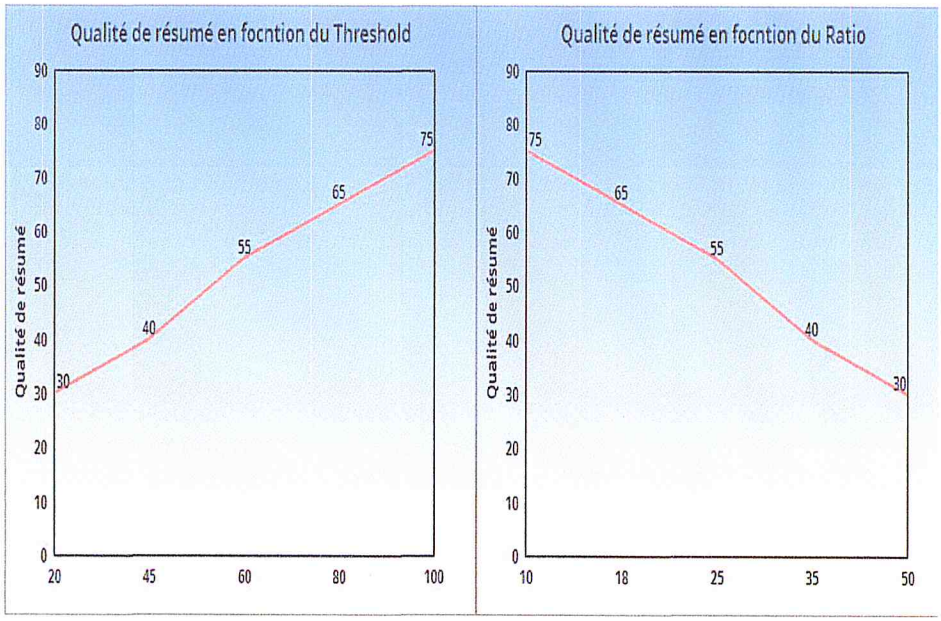


Figure 4.27 : Qualité de résumé en fonction du ratio et Threshold (corpus français)

On remarque que la qualité du résumé augmente avec l'augmentation du Threshold et elle diminue avec l'augmentation du ratio.

Et dans le cas d'un corpus en anglais on a obtenu les résultats suivants (**figure 4.28**) :

Threshold	Ratio	Qualité
20	50	40%
45	35	60%
60	25	70%
80	15	75%
100	10	85%

Figure 4.28 : Résultats de tests pour le corpus anglais

On a transformé le tableau en 2 représentations graphiques en courbe lisse comme le montre la figure ci-dessous (**figure 4.29**) :

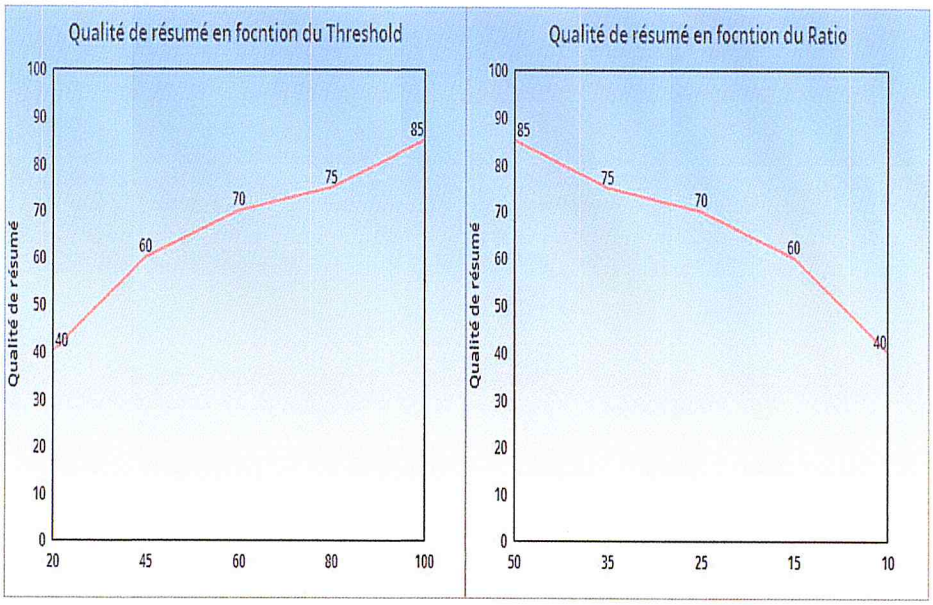


Figure 4.29 : Qualité de résumé en fonction du ratio et Threshold (corpus anglais)

Donc on remarque que les résultats sont presque les mêmes que pour le corpus français mais juste que la qualité du résumé et un peu petit peu meilleur.

À partir de ces résultats, on a remarqué que à chaque fois on augmente le ratio, la qualité du résumé diminue mais le contraire pour le Threshold (Pour les deux corpus français et anglais).

4.5.2 Corpus d'évaluation (en termes de qualité)

Afin d'évaluer nos approches appliquée au résumé multi-documents, nous avons opté pour le corpus DUC2004 utilisé dans la tâche 2 de la conférence DUC; résumés courts multi documents.

Le corpus contient 50 thèmes en Anglais, et chaque thème comporte 10 documents. Ces documents sont des articles de presse issus de deux agences de presse : AP et New York Times. Pour chaque thème, quatre résumés de référence sont fournis, chacun d'eux ne dépasse pas 665 caractères, espaces et ponctuations incluses. Dans notre cas on a utilisé 5 thèmes parmi les 50 thèmes pour évaluer nos approches appliquées (Naïve Bayes et Fuzzy), la figure ci-dessous montre les résultats obtenus en utilisant le paquet Rouge 2.0 avec l'existence des mesures de ROUGE (Rappel, Précision, F-mesure) :

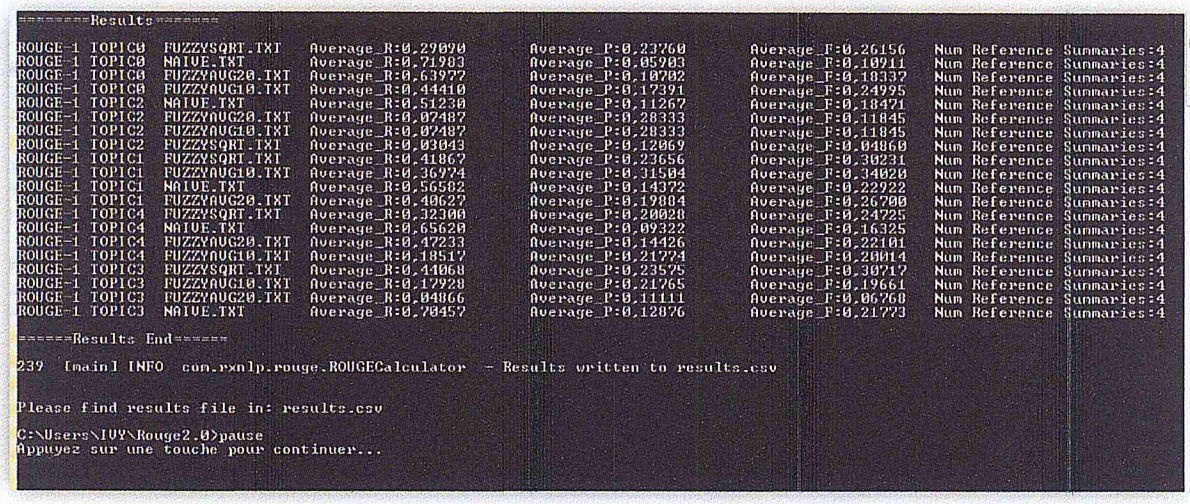


Figure 4.30 : Résultats obtenus en exécutant rouge2.0

Le tableau suivant résume les résultats obtenus précédemment pour les 5 différents thèmes :

Méthodes	Naïve Bayes			Fuzzy (raciné carrée)			Fuzzy (Moyenne 10 et 20)		
Mesures	Rappel	Précision	F-mesure	Rappel	Précision	F-mesure	Rappel	Précision	F-mesure
TOPIC0	71,98%	5,90%	10,91%	29,09%	23,76%	26.15%	54,19%	14,04%	21,66%
TOPIC1	56,58%	14,37%	22,92%	41,86%	23,65%	30,23%	38,80%	25,69%	30,76%
TOPIC2	51,23%	11,26%	18,47%	3,04%	12,06%	4,86%	7,48%	28,33%	11,84%
TOPIC3	70,45%	12,87%	21,77%	44,06%	23,57%	30,71%	11,39%	16,43%	13,21%
TOPIC4	65,62%	9,32%	16,32%	32,30%	20,02%	24,72%	32,87%	18,10%	21,05%
Moyenne	63,17%	10,74%	18,07%	30,07%	20,61%	23,33%	54,29%	20,51%	19,70%

Figure 4.31 : Résumé avec la moyenne des résultats obtenus depuis le ROUGE

La figure 4.31, représente la moyenne des résultats obtenus dans la figure 4.30 pour les 5 thèmes. Concernant le Rappel, on constate que l'approche Naïve bayes donne le meilleur résultat mais en terme de F-mesure et de Précision, on constate que l'approche Fuzzy pour la racine carrée donne un meilleur résultat par rapport à la moyenne. De là, on peut conclure que l'approche Naïve bayes augmente la performance de notre système de résumé automatique multi-document vue que le rappel est généralement plus difficile à obtenir que la précision.

4.6 Conclusion

Dans ce chapitre, nous avons décrit brièvement le processus de réalisation de notre application en spécifiant l'environnement de développement, les bibliothèques utilisées et la démarche qu'on a adoptée pour réaliser notre projet. Ensuite pour juger notre système, celui-ci est comparé à des résumés manuels (de référence) en utilisant le ROUGE pour le résumé multi-document. Cela nous a permis de calculer les trois métriques (rappel, précision et f-mesure) afin d'obtenir des mesures plus précise de la qualité de notre résumé. Toutes les parties sont fonctionnelles, malgré qu'il y aura toujours quelques fonctionnalités à optimiser vu que le fond du notre projet est la fouille de données.

Conclusion et perspectives

Dans ce travail notre objectif était de générer automatiquement un résumé multi-document qui récapitule les points importants au thème qui se trouvent dans le contenu des documents choisis par l'utilisateur. Pour ce faire nous avons proposé deux approches de classification et de segmentation dans le but de calculer la probabilité qu'une phrase représente les différents thèmes du texte d'entrée. Enfin les phrases sont triées selon leurs scores (approche bayésienne) ou selon les valeurs d'appartenance au cluster (approche relationnelle du Clustering flou), afin d'en extraire les premières phrases qui vont générer notre résumé multi-documents.

A la fin, nous avons effectué des tests sur nos deux approches. Nous avons utilisé le système d'évaluation Rouge sur le corpus DUC2004. Les expérimentations conduites ont montré d'une part que la méthode naïve bayes peut fournir des résultats meilleurs vue qu'elle améliore le Rappel du système, d'une autre part ont montré que la méthode de Clustering flou peut parfois fournir des bon résultats en termes de f-mesure et précision du système. En variant le nombre de cluster, il reste toujours à améliorer les résultats obtenus par cette dernière.

Ce projet a été très bénéfique pour nous car il nous a permis de renforcer et enrichir nos connaissances théoriques dans le domaine de la conception, et de mettre en application nos connaissances acquises le long de nos études. Nous avons pu faire le lien entre tous les modules que nous avons étudié : génie logiciel, Intelligence Artificielle, reconnaissance des formes, Data Mining et la programmation orientée objet (POO), où il nous a donné l'occasion de maîtriser le langage de programmation Java, de découvrir de plus en plus le domaine vaste du Data et du Text Mining, et encore de nous familiariser avec la conduite des projets informatiques.

En perspective, on vise à améliorer la qualité des résultats obtenus afin de générer automatiquement un résumé multi-documents dynamique qui récapitule les points importants au thème d'un mot clé tapé par un utilisateur dans un moteur de recherche.

Bibliographie

[1] ANSI, *American National Standard for Writing Abstracts* Z39-14, ANSI, New York, 1979.

[2] VAN DIJK T.A., *Text and context: explorations in the semantics and pragmatics of discourse,* Longman, Londres, 1980.

[3] Horacio Saggion and Guy Lapalme. 2002. Generating informative and indicative summaries with SumUM. *Computational Linguistics*, 28(4):497–526, Dec.

[4] Eduard Hovyet Chin-Yew Lin. *Automated text summarization and the* SUMMARIST *system.* In Proceedings of a workshop on held at Baltimore, Maryland: October13-15, 1998, pages 197–214. Association for Computational Linguistics, 1998.

[5] Karen Sparck Jones. *Automatic summarizing: factors and directions.* In Advances in automatic text summarization. Cambridge MA: MIT Press, 1999.

[6] Chris D. Paice et Paul A. Jones. *The identification of important concepts in highly structured technical papers.* In Research and Development in Information Retrieval, pages 69–78, 1993.

[7] Anna Kazantseva. *An approach to summarizing short stories.* In Proceedings of the Eleventh Conference of the European Chapter of the Association for Computational Linguistics: Student Research Workshop, EACL '06, pages 55–62, Stroudsburg, PA, USA, 2006. Association for Computational Linguistics.

[8] ChioriHori et SadaokiFurui. *Advances in automatic speech summarization.* In Proceedings of the 7th European Conference on Speech Communication and Technology, pages 1771–1774, 2001.

[9] Akira Inoue, Takayoshi Mikami et Yoichi Yamashita. *Improvement of Speech Summarization Using Prosodic Information.* In Proc. Speech Prosody, 2004.

[10] A. Ekin, A. M. Tekalp et R. Mehrotra. *Automatic soccer video analysis and summarization.* Image Processing, IEEE Transactions on, vol. 12, no. 7, pages 796–807, July 2003.

[11] M. Albanese, M. Fayzullin, A. Picariello et V.S. Subrahmanian. *The priority curve algorithm for video summarization.* Information Systems, vol. 31, no. 7, pages 679–695, November 2006.

[12] C. Carson, S. Belongie, H. Greenspan et J. Malik. Blob world: *image segmentation using expectation-maximization and its application to image querying.* IEEE

Transactions on Pattern Analysis and Machine Intelligence, vol. 24, no. 8, pages 1026– 1038, August 2002.

[13] Li Fei-Fei, Rob Fergus et Pietro Perona. *A Bayesian Approach to Unsupervised One-Shot Learning of Object Categories.* In Proceedings of the Ninth IEEE International Conference on Computer Vision - Volume 2, ICCV '03, pages 1134–,Washington, DC, USA, 2003. IEEE Computer Society

[14] Jiaming Zhan, Han Tong Lohet Ying Liu. *Gather customer concerns from online product reviews - A text summarization approach.* Expert Syst. Appl., vol. 36, no. 2, pages 2107–2115, March 2009.

[15] Lawrence H. Reeve, Hyoil Han et Ari D. Brooks. The use of domain-specific concepts in biomedical text summarization. In Information Processing and Management 43, volume 43, pages 1765–1776. Elsevier Ltd., 2007.

[16] Tadashi Nomoto et Yuji Matsumoto. *The diversity-based approach to open domain text summarization.* Inf. Process. Manage., vol. 39, no. 3, pages 363–389, May 2003.

[17] H. P. Luhn. *The automatic creation of literature abstracts.* IBM J. Res. Dev., vol. 2, no. 2, pages 159–165, April 1958.

[18] H. P. Edmundson. *New Methods in Automatic Extracting* .J. ACM, vol. 16, no. 2, pages 264–285, April 1969.

[19] Ruslan Mitkov. *Automatic Abstracting in a Limited Domain.* pages 204–210, 1993.

[20] Julian Kupiec, Jan Pedersen et Francine Chen. *A trainable document summarizer.* In Proceedings of the 18th annual international ACM SIGIR conference on Research and development in information retrieval, SIGIR '95, pages 68–73, New York, NY, USA, 1995. ACM.

[21] Atefeh Farzindar, Guy Lapalme et Jean-Pierre Des clés. *Résumé de textes juridiques par identification de leur structure thématique.* Traitement automatique de la langue (TAL), vol. 45, no. 1, pages 39–64, jan 2004.

[22] Kathleen McKeown et Dragomir R. Radev. *Generating summaries of multiple news articles.* In Proceedings of the 18th annual international ACM SIGIR conference on Research and development in information retrieval, SIGIR '95, pages 74–82, New York, NY, USA, 1995. ACM

[23] Florian Boudin et Juan-Manuel Torres-Moreno. *Résumé automatique multi document et indépendance de la langue : une première évaluation en français.* In Traitement Automatique des Langues Naturelles (TALN), 2009.

[24] Anna Kazantseva. *An approach to summarizing short stories.* In Proceedings of the Eleventh Conference of the European Chapter of the Association for Computational Linguistics: Student Research Workshop, EACL '06, pages 55–62, Stroudsburg, PA, USA, 2006. Association for Computational Linguistics.

[25] Shixia Liu, Michelle X. Zhou, Shimei Pan, Weihong Qian, Weijia Cai et Xiaoxiao Lian. *Interactive, topic-based visual text summarization and analysis.* In Proceedings of the 18th ACM conference on Information and knowledge management, CIKM '09, pages 543–552, New York, NY, USA, 2009. ACM. 11, 12.

[26] Javier Couto et Gustavo Crispino. *Construction automatique de résumés : une approche dynamique.* Traitement automatique des langues (TAL), Résumé automatique de textes, vol. 45, no. 1/2004, pages 95–120, April 2004.

[27] Nicolas Usunier, Massih-Reza Amini et Patrick Gallinari. *Résumé automatique de texte avec un algorithme d'ordonnancement.* In Proceedings of the 2nd Conférence en Recherche d'Informations et Applications (CORIA 2005), pages 25–40, 2005.

[28] Praveen Bysani. *Detecting novelty in the context of progressive summarization.* In Proceedings of the NAACL HLT 2010 Student Research Workshop, HLT-SRWS '10, pages 13–18, Stroudsburg, PA, USA, 2010. Association for Computational Linguistics.

[29] Dragomir R. Radev, Eduard Hovyet Kathleen McKeown. *Introduction to the special issue on summarization.* Comput. Linguist., vol. 28, no. 4, pages 399–408, December 2002.

[30] T Elizabeth Workman, Marcelo Fiszmanet John F Hurdle. *Text summarization as a decision support aid.* BMC Medical Informatics and Decision Making, October2012.

[31] Michel Généreux et Aurélien Bossard. *Résumé automatique de textes d'opinions.* In TALN 2009, 2009.

[32] Thorsten Brants. *Natural Language Processing in Information Retrieval.* In Proceedings of CLIN 2004, pages 1–13, Antwerp, Belgium, 2004.

[33] Geoffrey Nunberg. The Linguistics of Punctuation (Center for the Study of Language and Information - Lecture Notes). Center for the Study of Language and Inf, August 1990.

[34] David D. Palmer. *Text Preprocessing.* In Nitin Indurkhya et Fred J. Damerau, editeurs, Handbook of Natural Language Processing, Machine learning & Pattern Recognition.CRC Press, Taylor and Francis Group, Boca Raton, FL, deuxième édition, 2010.

[35] David D. Palmer et Marti A. Hearst. Adaptive multilingual sentence boundary disambiguation. Comput. Linguist., vol. 23, no. 2, pages 241–267, June 1997.

[36] Michael D. Riley. Some applications of tree-based modelling to speech and language. In Proceedings of the workshop on Speech and Natural Language, HLT '89, pages 339–352, Stroudsburg, PA, USA, 1989. Association for Computational Linguistics.

[37] Hans Müller, V. Amerl et G. Natalis. *Worterkennungsverfahren als Grundlage einer Universalmethode zur automatischen Segmentierung von Texten in Sätze.* Ein Verfahren zur maschinellen Satzgrenzenbestimmung im Englischen. vol. 4, no. 1, pages 46–64, 1980.

[38] Jeffrey C. Reynar et Adwait Ratnaparkhi. *A maximum entropy approach to identifying sentence boundaries.* In Proceedings of the fifth conference on Applied natural language processing, ANLC '97, pages 16–19, Stroudsburg, PA, USA, 1997. Association for Computational Linguistics.

[39] Andrei Mikheev. Periods, capitalized words, etc. Comput. Linguist., vol. 28, no. 3, pages 289–318, September 2002.

[40] John Aberdeen, John Burger, David Day, Lynette Hirschman, Patricia Robinson et Marc Vilain. MITRE : description of the Alembic system used for MUC-6. In Proceedings of the 6th conference on Message understanding, MUC6 '95, pages 141–155, Stroudsburg, PA, USA, 1995. Association for Computational Linguistics

[41] L. Breiman, J. H. Friedman, R. Olshenet C. J. Stone. Classification and regression trees. Wadsworth & Brooks, 1984. 39.

[42] Richard Sproat, William Gale, Chilin Shih et Nancy Chang. *A stochastic finitestate word-segmentation algorithm for Chinese.* Comput. Linguist., vol. 22, no. 3, pages 377–404, September 1996.

[43] David D. Palmer. *A trainable rule-based algorithm for word segmentation.* In Proceedings of the 35th Annual Meeting of the Association for Computational Linguistics and Eighth Conference of the European Chapter of the Association for Computational Linguistics, ACL '98, pages 321–328, Stroudsburg, PA, USA, 1997. Association for Computational Linguistics.

[44] Julia Hockenmaier et Chris Brew. *Error-driven learning of Chinese word segmentation.* In J. Guo, K. T. Lua et J. Xu, editeurs, 12th Pacific Conference of Language and Information, pages 218–229. Chinese and Oriental Languages Processing Society, Singapore, 1998.

[45] W. J. Teahan, Rodger McNab, Yingying Wen et Ian H. Witten. *A compression based algorithm for Chinese word segmentation.* Comput. Linguist., vol. 26, no. 3, pages 375–393, September 2000.

[46] Jianfeng Gao, Mu Li, Andi Wu et Chang-Ning Huang. *Chinese Word Segmentation and Named Entity Recognition : A Pragmatic Approach.* Comput. Linguist., vol. 31, no. 4, pages 531–574, December 2005.

[47] Martin Hassel. *Resource Lean and Portable Automatic Text Summarization.* PhDthesis, KTH School of Computer Science and Communication, Stockholm, Sweden, 2007.

[48] M. F. Porter. *Readings in information retrieval.* chapitre An algorithm for suffix stripping, pages 313–316. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1997.

[49] Sock Yin Tai, Cheng Soon Onget Noor Aida Abullah. *On designing an automated Malaysian stemmer for the Malay language (poster session).* In Proceedings of the fifth international workshop on on Information retrieval with Asian languages, IRAL '00, pages 207–208, New York, NY, USA, 2000. ACM.

[50] Mark Greengrass, Alexander M. Robertson, Robyn Schinkeet Peter Willett. *Processing morphological variants in searches of Latin text.* Inf. Res., vol. 2, no. 1, 1996.

[51] Vinsensius Berlian Vega SN et Stéphane Bressan. *Indexing the Indonesian Web: Language Identification and Miscellaneous Issues.* In WWW Posters, 2001.

[52] Johan Carlberger, Hercules Dalianis, Martin Hassel et Ola Knutsson. *Improving Precision in Information Retrieval for Swedish using Stemming.* In Proceedings of NODALIDA'01 - 13th Nordic Conference on Computational Linguistics, Uppsala, Sweden, May 21-22 2001.

[53] Wessel Kraaij et Renée Pohlmann. *Viewing stemming as recall enhancement.* In Proceedings of the 19th annual international ACM SIGIR conference on Research and development in information retrieval, SIGIR '96, pages 40–48, New York, NY, USA, 1996. ACM.

[54] Christof Monz et Maarten de Rijke. *Shallow Morphological Analysis in Monolingual Information Retrieval for Dutch, German, and Italian.* In Revised Papers from the Second Workshop of the Cross-Language Evaluation Forum on Evaluation of Cross-Language Information Retrieval Systems, CLEF '01, pages 262–277, London, UK, UK, 2002. Springer-Verlag.

[55] Isabelle Moulinier, J. Andrew McCulloh et Elizabeth Lund. *West Group at CLEF 2000: Non-english Monolingual Retrieval.* In Revised Papers from the Workshopof Cross-Language Evaluation Forum on Cross-Language Information Retrieval and Evaluation, CLEF '00, pages 253–260, London, UK, UK, 2001. Springer-Verlag.

[56] Mirko Popovi˘cet Peter Willett. *The effectiveness of stemming for natural language access to Slovene textual data.* J. Am. Soc. Inf. Sci., vol. 43, no. 5, pages 384–390, 1992.

[57] F. Çuna Ekmekçioglu, Michael F. Lynch et Peter Willett. *Stemming and n-gram matching for term conflation in Turkish texts.* In Information Research News, volume 7, pages 2–6, 1996.

[58] Leah S. Larkey, Lisa Ballesteros et Margaret E. Connell. *Improving stemming for Arabic information retrieval: light stemming and co-occurrence analysis.* In Proceedings of the 25th annual international ACM SIGIR conference on Research and development in information retrieval, SIGIR '02, pages 275–282, New York, NY, USA, 2002. ACM.

[59] G. Salton et C. S. Yang. *On the specification of term values in automatic indexing. Journal of Documentation.*, vol. 29, no. 4, pages 351–372, 1973.

[60] Exploitation of Named Entities in Automatic Text Summarization for Swedish. In In Proceedings of NODALIDA 03 - 14 th Nordic Conference on Computational Linguistics, Reykjavik, Iceland.

[61] NG J. P., BYSANI P., LIN Z., KAN M.-Y. & TAN C.-L. SWING : Exploiting Category-Specific Information for Guided Summarization. *In Proceedings of the Text Analysis Conference (TAC 2011)*, Gaithersburg, Maryland, USA.

[62] *Composing Measures for Computing Text Similarity.*Rapport interne TUD-CS-2015-0017, TU Darmstadt, Allemagne.

[63] Graph-based ranking algorithms for sentence extraction, applied to text summarization. *In Proceedings of the ACL Interactive Poster and Demonstration Sessions.*

[64] P. B. Baxendale. *Machine-made index for technical literature : an experiment.* IBM J. Res. Dev., vol. 2, no. 4, pages 354–361, October 1958.

[65] Chin-Yew Lin et Eduard Hovy. *Identifying topics by position.* In Proceedings of the fifth conference on Applied natural language processing, ANLC '97, pages 283–290, Stroudsburg, PA, USA, 1997. Association for Computational Linguistics.

[66] Chikashi Nobata et Satoshi Sekine. CRL/NYU *Summarization System at DUC-2004.*In DUC, 2004.

[67] *Identifying Topics by Position. In Proceedings of the Fifth Conference on Applied Natural Language Processing*, p. 283–290, Washington, DC, USA.

[68] Towards Multidocument Summarization by Reformulation: Progress and Prospects. In Proceedings of the Sixteenth National Conference on Artificial Intelligence and the Eleventh Innovative Applications of Artificial Intelligence Conference Innovative Applications of Artificial Intelligence, AAAI '99/IAAI '99, p. 453–460, Menlo Park, CA, USA.

[69] Mohamed Abdel Fattah et Fuji Ren. *GA, MR, FFNN, PNN and GMM based models for automatic text summarization.* Comput. Speech Lang., vol. 23, no. 1, pages 126–144, January 2009.

[70] Gerard Salton et Christopher Buckley. *Term-weighting approaches in automatic text retrieval.* Inf. Process. Manage., vol. 24, no. 5, pages 513–523, August 1988.

[71] Kai Ishikawa, Shinichi Ando et Akitoshi Okumura. *Hybrid Text Summarization Method based on the TF Method and the LEAD Method.* In Proceedings of the Second.

[72] Joel LaroccaNeto, Alex Alves Freitaset Celso A. A. Kaestner. *Automatic Text Summarization Using a Machine Learning Approach.* In Proceedings of the 16th Brazilian Symposium on Artificial Intelligence: Advances in Artificial Intelligence, SBIA '02, pages 205–215, London, UK, UK, 2002. Springer-Verlag.

[73] C. D. Paice. *The automatic generation of literature abstracts: an approach based on the identification of self-indicating phrases.* In Proceedings of the 3rd annual ACM conference on Research and development in information retrieval, SIGIR '80, pages172–191, Kent, UK, UK, 1981. Butterworth & Co.

[74] Gerard Salton et M.J. McGill: *Introduction to Modern Information Retrieval.* 1983.

[75] Chikashi Nobata, Satoshi Sekineet Hitoshi Isahara: Evaluation of features for sentence extraction on different types of corpora. *In Proceedings of the ACL 2003 workshop on Multilingual summarization and question answering*, pages 29–36, Morristown, NJ, USA, 2003. Association for Computational Linguistics.

[76] Lucia Helena Machado Rinoet Maria das Graças Volpe Nunes: Extractive summarization: how to identify the GIST of a text. *In Proceedings of I2TS'2002 - International Information Technology Symposium*, Florianopolis, SC, Brasil, 2002.

[77] Florian Boudin et Juan-Manuel Torres-Moreno: Neo-cortex: A perform ant user oriented multi-document summarization system. *In* Alexander F. Gelbukh, éditeur : *CICLing*, volume 4394 de *Lecture Notes in Computer Science*, pages 551–562. Springer, 2007.ISBN 3-540-70938-X.

[78] Jen-Yuan Yeh, Hao-RenKeet Wei-Pang Yang: Chinese text summarization using a trainable summarizer and latent semantic analysis. *In ICADL '02: Proceedings of the5th International Conference on Asian Digital Libraries*, pages 76–87, London, UK, 2002. Springer-Verlag. ISBN 3-540-00261-8.

[79] Miles Osborne: Using maximum entropy for sentence extraction. *In Proceedings of the ACL-02 Workshop on Automatic Summarization*, pages 1–8, Morristown, NJ, USA, 2002. Association for Computational Linguistics.

[80] KAIKHAH K. Automatic Text Summarization with Neural Networks. In Proceedings of IEEE International Conference on Intelligent Systems, volume 1, p. 40–44.

[81] MCDONALD R. A study of global inference algorithms in multi-document summarization. In Proceedings of the 29th European Conference on IR Research, ECIR'07, p. 557–564, Berlin, Heidelberg : Springer-Verlag.

[82] LI P., WANG Y., GAO W. & JIANG J. Generating aspect-oriented multi-document summarization with event aspect model. *In Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing,* p. 1137–1146, Edinburgh, Scotland, UK.

[83] WOODSEND K. & LAPATA M. Multiple aspect summarization using integer linear programming. In Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning, p. 233–243, Jeju Island, Korea.

[84] Günes Erkan et Dragomir R. Radev: Lexrank: Graph-based centrality as salience in text summarization. *Journal of Artificial Intelligence Research (JAIR),* vol.22, pp. 457-479.

[85] PAGE L., BRIN S., MOTWANI R. & WINOGRAD T. The Page Rank Citation Ranking: Bringing Order to the Web. Technical Report 1999-66, Stanford Info Lab.

[86] Dragomir Radev, Timothy Allison: MEAD - a platform for multidocument multilingual text summarization. *In Proceedings of LREC 2004*, Lisbon, Portugal, May 2004.

[87] Chin-Yew Lin. ROUGE : A Package for Automatic Evaluation of summaries. In Proc. ACL workshop on Text Summarization Branches Out, page 10, 2004.

[88] NENKOVA A. & PASSONNEAU R. J. Evaluating content selection in summarization: The pyramid method. In *HLT-NAACL*, p. 145–152.

[89] PASSONNEAU R. J. Automated pyramid scoring of summaries using distribution al semantics. In Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Short Papers), p. 143–147, Sofia, Bulgaria.

[90] P.Hayes, S.P.Weinstein « Construe/Tis: A system for content-based indexing of a database of news stories »

[91] R.Armstrong, D.Freitag, « WebWatcher: a Learning apprentice for the World Wide Web »

[92] Bisson, G., La similarité : une notion symbolique/numérique. Chap. XX of : Apprentissage symbolique-numérique (tome 2). Editions CEPADUES.2002.

[93] G.Saporta « Probabilités, Analyse des données et Statistique »

[94] L. Candillier, ''Contextualisation, visualisation et évaluation en apprentissage non supervisée'', Thèse de doctorat, page 68, Université Charles De Gaulle Lile 3. 2006.

[95] A. K. Jain, M. N. Murty, and P. J. Flynn. "Data clustering: a review", ACM Computer Survey, 31(3):264–323. 1999.

[96] T. N. Tran, R. Wehrens, and L. M. Buydens, "Clustering multispectral images: a tutorial Chemometrics and Intelligent Laboratory Systems", 77(1–2):3–17. 2005.

[97] J. B. MacQueen, "Some methods for classification and analysis of multivariate observations", In L. M. Le Cam and J. Neyman, editors, Proc. of the fifth Berkeley Symposium on Mathematical Statistics and Probability, volume 1, pages 281–297. University of California Press. 1967.

[98] Dunn J. Well, "Separated clusters and optimal fuzzy partitions", Journal of Cybernetics, 4, 95-104. 1974.

[99] E. Schikuta et M. Erhart, "The bang clustering system: Grid-based data analysis", In International Symposium on Advances in Intelligent Data Analysis, Reasoning a bout Data, pages 513–524. 1997.

[100] C. C. Aggarwal, J. L. Wolf, P. S. Yu, C. Procopiuc, et J. S. Park. ''Fast algorithms for projected clustering", In Proceeding of the ACM coneference of SIGMOD Record, 28(2) : 61–72. 1999.

[101] A. P. Dempster, N. M. Laird, et D. B. Rubin, "Maximum likelihood from incomplete data via the em algorithm", Journal of the Royal Statistical Society, 39(1): 1–38. 1977.

[102] Benzécri J.P. L'analyse des données. Dunod, Paris, 197.

[103] Celeux, G., Diday, E., Govaert, G., Lechevallier, Y., and Ralambondrainy, H. Classification automatique des données, environnement statistique et informatique. DUNOD informatique. 1989.

[104] Ball, G. H. et Hall, D. J. ISODATA, an Iterative Method of Multivariate Analysis and Pattern Recognition. Behavior Science, 153, 1967.

[105] J. C. Dunn (1973) : "A Fuzzy Relative of the ISODATA Process and Its Use in Detecting Compact Well-Separated Clusters", Journal of Cybernetics, no 3, pp 32-57. 1973.

[106] J. C. Bezdek (1981) : "Pattern Recognition with Fuzzy Objective Function Algoritms", Plenum Press, New York. 1981.

[107] V. di Gesu. « Mathematical Morphology and Image Analysis : A Fuzzy Approach ».Workshop on Knowledge-Based Systems and Models of Logical Reasoning, Reasoning, 1988.

[108] H. Hartley, Maximum likelihood estimation from incomplete data. Biometrics, no 14, pp174–194. 1958.

[109] A.P. Dempster, N.M. Laird , and D.B. Rubin : Maximum-vraisemblance from incomplete data via the EM algorithm", J.Royal Statist. Society, B39, pp1-38, 1977.

[110] M. Tanner, Tools for Statistical Inference. Springer Verlag, New York. Third Edition.1996.

[111] Geoffrey. McLachlan and Thriyambakam. Krishnan. The EM Algorithm and Extensions (2nd edition). Wiley, New Jersey, 2008.»

[112] Andrew Skabar and Khaled Abdalgader, "Clustering Sentence Level Text Using A Novel Fuzzy Clustering Algorithm". January 2013, IEEE Transactions on Knowledge and Data Engineering, Vol. 25, no. 1, pp 62-75.

[113] S. Brin and L. Page, "The Anatomy of a Large-Scale Hypertextual Web Search Engine," Computer Networks and ISDN Systems, vol. 30, pp. 107-117, 1998.

[114] A.P. Dempster, N.M. Laird, and D.B. Rubin, "Maximum Likelihood from Incomplete Data via the EM Algorithm," J. the Royal Statistical Soc. Series B (Methodological), vol. 39, no. 1, pp. 1-38, 1977.