République Algérienne Démocratique et Populaire
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique
Université Saad Dahleb de Blida
Faculté des Sciences
Département d'informatique

Mémoire de fin d'étude

Pour L'Obtention du diplôme de Master en *Informatique*
Option : *Ingénerie de Logiciel*

Vers un meilleur résumé automatique multidocuments

Réalisé par :

Nom : Bouchelouche
Prénom : Khadidja

Nom : Sibsa
Prénom : Ahlem

Président : Mr. Bala Mahfoud
Examinatrice : Mme. Hadj Henni Malika
Promoteur : Mr. Kameche Abdallah Hicham

Soutenu le : **27 JUIN 2018**

REMERCIEMENT

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 souhaitons exprimer notre gratitude à notre Promoteur Kameche Abdallah Hicham. Nous le remercions de nous avoir encadrés, guidés, aidés et conseillés. Nous adressons nos sincères remerciements à tous les enseignants et tous ceux qui, par leurs paroles, leurs écrits, leurs conseils et leurs critiques ont guidé nos réflexions et ont accepté de nous rencontrer et de répondre à nos questions lors de nos recherches. Nous remercions nos chers parents, qui ont toujours été là pour nous. À tous ces intervenants, nous offrons nos remerciements, notre respect et notre gratitude.

ABSTRACT

The work done in the context of the automatic text summarization showed very encouraging results, but they are still to be improved. The problem of automatic summarization continues to evolve with the new fields of application that are required, which increases the constraints related to this task. We are interested in abstract text summarization multi-documents. For this, we examine the different existing approaches with a focus on the most recent work. We try to improve the task of automatic abstract multi-documents summarization systems by using the generative machine learning models of the artificial intelligence field.

Keywords: Automatic summarization, Multi documents, Machine learning, Abstract.

RÉSUMÉ

Les travaux menés dans le cadre du résumé automatique de texte ont montré des résultats à la fois très encourageants mais qui sont toujours à améliorer. La problématique du résumé automatique ne cesse d'évoluer avec les nouveaux champs d'application qui s'imposent, ce qui augmente les contraintes liées à cette tâche. Nous nous intéressons au résumé automatique abstractif multi-documents. Pour cela, nous examinons les différentes approches existantes en mettant l'accent sur les travaux les plus récents. Nous essayons d'améliorer la tâche des systèmes de résumé automatique abstractif multi-documents de texte en utilisant les modèles génératifs d'apprentissage automatique du domaine d'intelligence artificielle.

Mots clés: Résumé automatique, Multi documents, Apprentissage automatique, Abstractif.

ملخص

أظهرت الأعمال المنجزة في سياق الملخص النصي التلقائي نتائج مشجعة للغاية، ولكن لا يزال يتعين تحسينها. لا تزال مشكلة التلخيص التلقائي تتطور مع مجالات التطبيق الجديدة المطلوبة ، مما يزيد من القيود المتعلقة بهذه المهمة. لهذا نحن مهتمون بتطوير نظام الملخص النصي متعدد الوثائق المبني على إعادة الصياغة.

في هذا العمل سنقوم بدراسة مختلف الأساليب الموجودة مع التركيز على أحدث الأعمال. سنحاول تحسين مهمة الأنظمة التلخيصية للنصوص متعددة المستندات باستخدام النماذج التعليمية الآلية و العميقة المبتكرة في مجال الذكاء الاصطناعي.

الكلمات الدالة: التلخيص التلقائي، متعدد المستندات، التعلم الآلي، إعادة الصياغة.

TABLES DES MATIÈRES

LISTE DES TABLEAUX

TABLE DES FIGURES

INTRODUCTION GÉNÉRALE

De nos jours, les gens utilisent l'Internet pour trouver des informations grâce à des outils de récupération d'informations tels que Google, Yahoo, Bing et ainsi de suite. En raison du taux croissant de données, les gens ont besoin d'informations significatives. Ainsi, il n'est pas possible pour les utilisateurs de lire chaque document afin de trouver celui qui est utile. Parmi toutes les technologies modernes, le résumé multidocuments de texte est l'une des thématiques de recherche récentes à laquelle les chercheurs en Traitement Automatique des Langues se sont intéressés. Résumer un texte consiste à réduire ce texte en un nombre limité de mots. Le texte ainsi réduit doit rester fidèle aux informations et idées du texte original. Que ce soit pour des professionnels qui doivent prendre connaissance du contenu de documents en un temps limité ou pour un particulier désireux de se renseigner sur un sujet donné sans disposer du temps nécessaire pour lire l'intégralité des textes qui en traitent, le résumé est une aide contextuelle importante. Avec l'augmentation de la masse documentaire disponible électroniquement. La production automatique de résumés pose le problème de la détection et de la modélisation des informations contenues dans les textes. Elle suppose également la hiérarchisation de ces informations afin d'intégrer au résumé les plus importantes. Dans notre projet nous visant à améliorer le résumé automatique multidocuments de type abstractif, en combinant deux modèles génératifs en se basant sur l'apprentissage profond, du domaine d'intelligence artificielle. Les principales contributions de notre système du résumé sont :

1. La génération d'un résumé multi-documentaire à base d'apprentissage profond , alors qu'actuellement on ne peut générer que des headlines.
2. Les limitations des architectures récurrentes et l'utilisation de modèles génératifs à la place.
3. La combinaison de deux modèles génératifs.

Afin de faciliter la lecture de ce mémoire, nous allons présenter brièvement les chapitres qui le composent. Ces chapitres sont répartis comme suit :

Chapitre 1 : - Résumé automatique de texte - : Dans ce chapitre nous commencerons par définir le résumé automatique de texte, et présenter leurs différents types, ensuite, nous identifierons les étapes du processus de résumé automatique ainsi que les critères de sélection des unités de texte, et enfin nous terminerons par l'évaluation des résumés automatiques.

Chapitre 2 : - État de l'art - : Dans ce chapitre nous allons présenter les techniques de résumé automatique, pour le type extractif et le type abstractif, qui est lui-même classé en deux approches : l'approche structurelle et l'approche sémantique, et nous détaillerons les différentes techniques pour chaque type.

Chapitre 3 : - Apprentissage automatique - : Dans ce chapitre nous allons définir l'apprentissage automatique, les réseaux de neurones et les réseaux de neurones profonds.

Chapitre 4 : - Conception - : Dans ce chapitre nous allons définir la problématique initiale de génération des résumés automatiques multidocuments, décrire les étapes de génération des résumés pour notre système (les prétraitements, les traitements et les post-traitements), et nous terminerons ce chapitre avec une conclusion récapitulative de ce que nous avons fait.

Chapitre 5 : - Réalisation - : Dans ce chapitre nous allons présenter les outils utilisés pour la réalisation de notre système du résumé automatique multidocuments (hardware et software), les outils de l'évaluation des résumés générés, les benchmarks qui existent et en particulier ceux utilisés comme corpus pour l'entrainement et le test de notre système. Nous allons présenter aussi l'évaluation de notre système comme une étude paramétrique et comparative en testant le système avec différentes fonctions d'erreur et fonctions d'activation, et à la fin nous allons conclure ce chapitre par une conclusion récapitulative.

CHAPITRE 1

RÉSUMÉ AUTOMATIQUE DE TEXTE

1.1 INTRODUCTION :

Les données sur le World Wide Web se développent à un rythme exponentiel. De nos jours, les gens utilisent Internet pour trouver des informations grâce à des outils de récupération d'informations (IR) tels que Google, Yahoo, Bing et ainsi de suite. Cependant, avec la croissance exponentielle de l'information sur Internet, l'abstraction de l'information ou le résumé des résultats obtenus est devenu nécessaire pour les utilisateurs. Dans l'ère actuelle de la surcharge d'informations, le résumé de texte est devenu un outil important et opportun pour l'utilisateur de comprendre rapidement le grand volume d'informations. Le but du résumé automatique de texte est de condenser les documents dans une version plus courte et de préserver les contenus importants (Khan & Salim, 2014).
Dans ce chapitre, nous allons définir le résumé automatique de texte, présenter les différents types de résumé automatique, identifier les étapes du processus de résumé automatique ainsi que les critères de sélection des unités de texte et d'évaluation des résumés automatiques.

1.2 DÉFINITION :

Le résumé de texte est la tâche d'extraire des informations saillantes du document texte original. Dans ce processus, les informations extraites sont générées sous la forme d'un rapport condensé et présentées sous la forme d'un résumé concis à l'utilisateur (Khan & Salim, 2014).
Mais c'est quoi un résumé automatique du texte ?

Un résumé automatique de texte a la même définition qu'un résumé de texte, sauf que cette fois-ci, le résumé se fait automatiquement, sans l'intervention de l'être humain.
Les sources d'information pouvant être résumées sont nombreuses et hétérogènes : documents vidéo, sonores ou textuels.
Notre étude porte uniquement sur le résumé de textes.

1.3 TYPE DE RÉSUMÉ :

Dans cette partie nous allons présenter les différents types du résumé automatique de texte.

1.3.1 EXTRACTIF :

Le point fort du résumé par extraction est qu'il évite la génération de texte. Ceci permet d'une part, de se concentrer sur la sélection du contenu pertinent et d'autre part, d'obtenir un résumé lisible et linguistiquement correct. La cohérence n'est en revanche pas garantie. Par exemple, si le système de résumé sélectionne des phrases contenant des références (acronyme, pronom personnel, etc.) et ne sélectionne pas les phrases contenant leurs antécédents, il est fort probable que le résumé produit soit incompréhensible. Pour pallier ce problème, certains travaux considèrent le paragraphe comme unité d'extraction au lieu de la phrase (Salton, Singhal, Buckley & Mitra, 1996). Ceci permet de garder la cohérence du texte source mais ne peut pas être applicable dans le cas de résumés courts. De plus, il est évident que cette méthode réduit la précision du résumé en y incluant des phrases peu importantes juste pour améliorer la cohérence. D'autres chercheurs procèdent à des étapes de pré/post-traitement du texte qui améliorent partiellement la cohérence globale du résumé, comme par exemple la résolution des références anaphoriques dans le texte source (Trandaba, 2011).Le processus principal dans le résumé extractif est la sélection des segments de textes (généralement les phrases) pertinents et non redondants sans dépasser une taille limite de résumé. Ce principe limite la couverture des informations apportées par le texte source. Les résumés abstractifs souffrent moins de ce problème puisque l'information peut y être reformulée (Mnasri, 2015).

1.3.2 ABSTRACTIF :

Les systèmes abstractifs partagent avec le résumé dynamique une certaine forme de modélisation du contenu des documents, même si les critères d'extraction dans le cas dynamique sont généralement sémantiquement moins profonds. Les méthodes de résumé abstractives

imitent, jusqu'à un certain degré, le processus naturel accompli par l'homme pour résumer un document. Par conséquent, elles produisent des résumés plus similaires aux résumés manuels. Ce processus peut être décrit par deux étapes majeures : la compréhension du texte source et la génération du résumé (Khan & Salim, 2014).Ces deux tâches sont assez complexes. C'est pourquoi elles ont été simplifiées. La première étape vise à analyser sémantiquement le contenu du texte et à identifier les parties à exprimer dans le résumé. Elle a parfois pris la forme d'une tâche d'extraction d'information liée au domaine abordé ou de regroupement des phrases du texte source (Genest & Lapalme, 2011), (Genest & Lapalme, 2012), (Filippova, 2010). La génération de texte est un domaine en soi. Une des approches simplifiées consiste à appliquer des techniques de génération text-to-text : utilisation de paraphrases ou fusion et compression de phrases (Filippova, 2010), (Madnani & Dorr, 2010).Une alternative consiste à induire un modèle textuel du domaine (patron) et de l'instancier lors de la génération (Cheung, Poon, & Vanderwende, 2013). Ces méthodes ne sont pas très largement exploitées. Ceci peut être dû à la rareté des outils de génération du texte. Le domaine de la génération est toujours en cours de développement mais pas encore à maturité, ce qui freine parfois l'implémentation des systèmes abstractifs. Les chercheurs préfèrent alors se tourner vers les méthodes extractives, qui ne dépendent pas de ce prérequis. La majorité des travaux s'étant intéressés aux méthodes extractives, ces dernières ont connu un développement plus important. La majorité des évaluations menées sur le résumé ont aussi été conçues pour des résumés plutôt extractifs. Ceci explique en partie les résultats moins encourageants des systèmes abstractifs. Néanmoins, certains chercheurs pensent que ceux-ci pourraient susciter un certain regain d'intérêt. Cette prédiction est justifiée d'une part, par le besoin de résumés plus proches des résumés manuels et d'autre part, par le plafonnement des performances des techniques extractives (Nenkova & Mckeown, 2012), (Mnasri, 2015).

1.3.3 STATIQUE :

Pour ce type nous avons toujours un résumé unique pour un document d'entrée, donc un texte unique fixe, c'est le contraire de type dynamique (Gustavo & Javier, 2004).

1.3.4 DYNAMIQUE :

Un résumé automatique ne constitue pas un texte unique fixe mais un objet multidimensionnel composé des informations jugées saillantes selon un profil de filtrage, des liens qui permettent une synchronisation automatique avec le texte source et des opérations de visualisation et

La figure 1.1 représente un schéma récapitulatif des différents types de résumé :

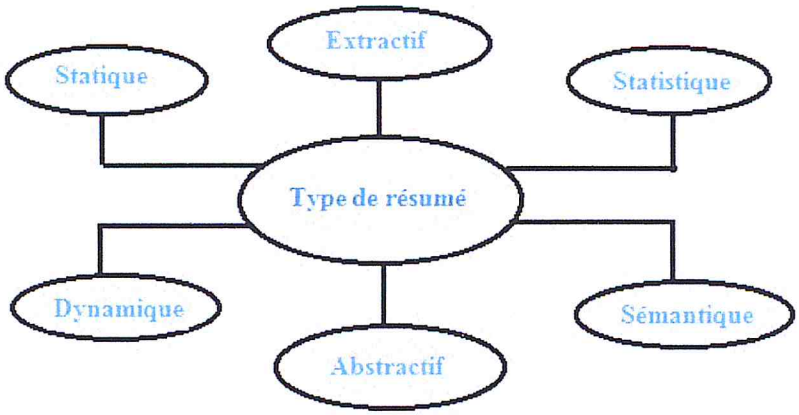

FIGURE 1.1 – Schéma représentatif des différents types de résumé.

1.4 PROCESSUS DU RÉSUMÉ AUTOMATIQUE :

1.4.1 PRÉTRAITEMENT :

Le Prétraitement dans le résumé automatique de textes (ou la recherche d'information en général) est une tâche très importante. Nous introduisons ci-dessous les différentes étapes utilisées dans le prétraitement, qui sont des techniques du traitement automatique du langage naturel (TALN) 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 (Brants, Ayoubi, Chada, Marchal, De Ven & Petit, 2004).

1.4.1.1 NORMALISATION :

La normalisation est la procédure qui transforme un document dans un format standard plus facile à manipuler(Heitz, 2006). 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 :

1. Suppression des caractères spéciaux et des chiffres.
2. Remplacement de certains mots équivalents, comme les abréviations (voir l'exemple précédent).

3. Dans certaines langues comme la langue Arabe, élimination des diacritiques.
4. 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. 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 l'une des trois marques de ponctuation précédentes (Nunberg, 1990). Mais il existe des cas où cette définition ne peut pas être appliquée :

   a) 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.

   b) Les phrases peuvent être délimitées par plusieurs autres marques de ponctuation (Palmer, 2010).

   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.

   a) 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 (Palmer, 2010) :

   a) 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.

   b) Parties du discours (Part of speech) : Palmer et Hearst 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 (Palmer & Hearst, 1997).

c) Longueur du mot : La longueur de mots avant et après un point, est utilisée par (Riley, 1989), comme un critère contextuel.

d) Les suffixes : l'analyse morphologique pour identifier les suffixes, et de ce fait filtrer les mots qui ne sont pas des abréviations est utilisée par (Hans & Natalis, 1980). Ceci est utilisé pour identifier les mots qui ne figurent pas dans la liste des abréviations.

e) Préfixes et suffixes : les préfixes et les suffixes des mots entourant la marque de ponctuation sont utilisés par (Reynar & Ratnaparkhi, 1997), comme un critère contextuel.

f) Les classes des abréviations : les abréviations sont divisées 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.)(Riley, 1989), (Reynar & Ratnaparkhi, 1997).

g) Les noms propres : la présence des noms propres à droite du point comme critère est utilisée par (Mikheev, 2002).

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 (Palmer, 2010) 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 (Aberdeen, Burger, Day, Hirschman, Robinson & Vilain, 1995). 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 (Riley, 1989). Sa méthode utilise les arbres de régression (Breiman, Friedman, Olshenet & Stone, 1984) 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.

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 :

   a) 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.

   b) 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.

   c) 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).

   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 (Indurkhya & Damerau, 2010).

1.4.1.3 RADICALISATION :

Dans tout texte, le même mot peut se produire dans 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 puissent 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 (Hassel, 2007). 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 (Hassel, 2007) :

1. Sur-radicalisation : se passe lorsque deux mots donnent la même racine, mais en réalité ils ne l'ont pas.
2. 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 (Brants, Ayoubi, Chada, Marchal, De Ven & Petit, 2004) : radicalisation linguistique avec dictionnaire et radicalisation de style Porter (Porter, 1997). 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, y compris le Malais (Tai, Ong & Abullah, 2000), le Latin (Greengrass, Robertson, Robyn & Willett, 1996), l'Indonésien (SN & Bressan, 2001), le Suédois (Carlberger, Dalianis, Duneld & Knutsson, 2001), le Néerlandais (Kraaij & Pohlmann, 1996), le Slovène (Popovic & Willett, 1992), le Turc (Ekmekcioglu, Lynch & Willett, 1996), et l'Arabe (Larkey, Ballesteros & Connell, 2002).

1.4.1.4 ELIMINATION DES MOTS VIDES :

L'élimination des 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 du 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) :

1. **To be or not to be**.
2. **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 (Brants, Ayoubi, Chada, Marchal, De Ven & Petit, 2004).

1.5 CRITÈRES DE SÉLECTION STATISTIQUES :

Une fois qu'on procède au prétraitement 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 qu'ils soient mono-document, multi-document ou dynamiques.

1.5.1 CRITÈRES LIÉS AU CONTENU DU TEXTE :

Cet ensemble de critères s'intéresse 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 le 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 (Luhn, 1958), 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 donné, 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, Singhal, Buckley & Mitra, 1996) ont défini 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 idf 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 le 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 SIMILARITÉ ENTRE LES PHRASES :

La similarité des textes est une notion très importante en TAL. Plusieurs mesures de similarité textuelle ont été établies(Bär, Zesch & Gurevych, 2015). 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. Le cas de l'algorithme de résumé mono-document *TextRank* (Graph-based ranking algorithms for sentence extraction, applied to text summarization). Ce critère est par ailleurs particulièrement important dans le cas multidocuments (Suanmali, Salim & Binwahlan, 2009).

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 (Luhn, 1958), où bien à la position de la phrase par rapport à une unité (document, section, paragraphe, etc.). 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.

1.5.2.2 MOTS DU TITRE ET DES SOUS-TITRES :

Dans (Edmundson, 1969),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 (Salton & Buckley, 1988), les auteurs proposent 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, (Ishikawa, Ando & Okumura, 2001) 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 :

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

Dans (Palmer & Hearst, 1997), 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 (1.1) :

$$Score_{titre}(p_i) = \frac{\sum_{m \in T \cap P_i} \frac{tf(m)}{tf(m)+1} idf(m)}{\sum_{m \in T} \frac{tf(m)}{tf(m)+1} idf(m)} \quad (1.1)$$

La deuxième méthode utilise les entités nommées et la fréquence de mots tf. Pour une entité nommée e, l'équation (1.2) est utilisée pour calculer le score d'une phrase p_i en se basant sur le titre :

$$Score_{titre}(p_i) = \frac{\sum_{e \in T \cap P_i} \frac{tf(e)}{tf(e)+1}}{\sum_{e \in T} \frac{tf(e)}{tf(e)+1}} \quad (1.2)$$

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 (Kupiec, Pedersen & Chen, 1995). 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 (Nobata & Sekine, 2004).

Dans le travail de (Nobata & Sekine, 2004), 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 (1.3))

$$Score_{long}(p_i) = \begin{cases} \frac{L_i}{L_{max}} & si(L_i \leq L_{max}) \\ 1 & sinon \end{cases} \quad (1.3)$$

La deuxième méthode donne un score négatif afin de pénaliser les phrases qui sont plus courtes que L_{min}(voir l'équation (1.4)).

$$Score_{long}(p_i) = \begin{cases} 0 & si(L_{min} \leq L_i) \\ \frac{L_i - L_min}{L_{min}} & sinon \end{cases} \quad (1.4)$$

Une autre formule pour calculer le score d'une phrase P_i dans un document d en utilisant sa longueur, elle est utilisée dans (Fattah & Ren, 2009)(voir l'équation (1.5)) :

$$Score_{pos}(P_i) = \frac{|P_i| * |\{P : P \in d\}|}{|d|} \quad (1.5)$$

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 phrases dans le document d.

1.5.2.4 MOTS INDICATIFS :

Les méthodes utilisant les mots indicatifs (*cuewords*), sont basées sur l'hypothèse que la pertinence d'une phrase est affectée par la présence de mots comme "*significatif* ", "*impossible*", et "*Difficilement*". Ces méthodes utilisent des mots sauvegardés dans un dictionnaire extrait à partir d'un corpus (Khan & Salim, 2014). 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 :

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

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(Paice, 1981), par exemple "Le but *principal de cet article est de vérifier ...*", "*Dans cet article, une méthode est décrite pour ...*", etc.

1.6 EXPLOITATION ET INTÉGRATION 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(Edmundson, 1969). 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 :

Celui-ci (Luhn, 1958) construit à partir des documents à résumer un paquet de mots importants dans ces documents. Seuls les mots dont la fréquence est inférieure à un seuil donné sont considérés importants et intégrés dans ce paquet. 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.

La figure 1.2 représente une illustration de la discriminance des mots selon leur fréquence d'après (Luhn, 1958).

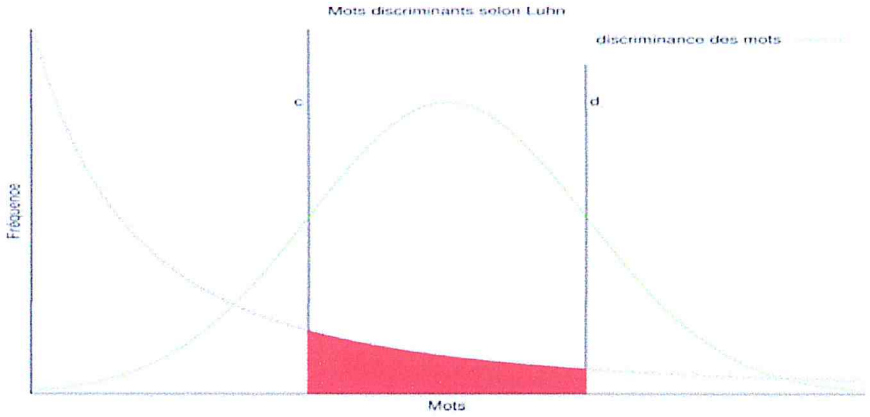

FIGURE 1.2 – Illustration de la discriminance des mots selon leur fréquence d'après (Luhn, 1958).

1.6.2 MÉTHODES PAR APPRENTISSAGE :

Du point de vue de l'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.

Une question se pose alors : comment déterminer la contribution de chacun de ces paramètres dans la sélection des phrases importantes ? Bien sûr, la réponse à cette question est dépendante du type de document que l'on veut résumer. Prenons, par exemple, le paramètre de position de la phrase dans le document. Dans le cas d'articles journalistiques, les premières phrases sont souvent les plus importantes (Brandow, Mitze & Rau, 1995) tandis que pour des articles scientifiques, les phrases provenant de la conclusion seront à privilégier. C'est dans cette optique que les approches par apprentissage s'avèrent être intéressantes, la valeur de chaque paramètre pouvant être estimée en comptant leurs occurrences dans un corpus. Ainsi de nombreuses recherches ont tenté d'analyser comment un corpus composé de paires [document/résumé associé généré manuellement] pouvait être utilisé afin d'apprendre automatiquement des règles ou des techniques pour la génération de résumé. Plus simplement, un exemple commun

d'utilisation de l'apprentissage réside dans le calcul de poids basés sur la fréquence des mots. La mesure tf.idf, largement utilisée dans le domaine de la Recherche d'Information (RI) (Sparck Jones, 1972), peut être utilisée en résumé automatique afin de distinguer les mots les plus discriminants d'un document et de les utiliser lors de l'étape de sélection.

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 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(Mnasri, 2015).

1.7.1 CORPUS DE TEST (RÉFÉRENCES) :

Pour arriver à une telle évaluation, on doit connaître d'abord les réponses idéales de l'utilisateur. Ainsi, l'évaluation d'un système s'est faite souvent avec certains corpus de test. Dans un corpus de test, il y a :

1. Un ensemble de documents ;
2. Un ensemble de requêtes ;
3. La liste de documents pertinents pour chaque requête.

L'évaluation d'un système ne doit pas se reposer seulement sur une requête. Pour avoir une évaluation assez objective, un ensemble de quelques dizaines de requêtes, traitant des sujets variés, est nécessaire. L'évaluation du système doit tenir compte des réponses du système pour toutes ces requêtes.

Finalement, il faut avoir les réponses idéales pour l'utilisateur pour chaque requête. Le dernier élément d'un corpus de test fournit cette information. Pour établir ces listes de documents pour toutes les requêtes, les utilisateurs (ou des testeurs simulant des utilisateurs) doit examiner chaque document de la base de document, et juger s'il est pertinent. Après cet exercice, on connaît exactement quels documents sont pertinents pour chaque requête. Pour la construction d'un corpus de test, les jugements de pertinence constituent la tâche la plus difficile.
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 (Mnasri, 2015).
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 (Mnasri, 2015).

$$\text{Précision} = \frac{\#\text{documents pertinents retrouvés}}{\#\text{documents retrouvés}}$$

$$\text{Rappel} = \frac{\#\text{documents pertinents retrouvés}}{\#\text{documents pertinents dans la base}}$$

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 (Mnasri, 2015).
Les méthodes et les corpus d'évaluation des résumés automatiques, en citant ROUGE, PYRAMID, Document Understanding Conference (DUC) et Gigaword seront présenté dans le chapitre 5.

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 types. Donc, nous avons vu les différents types de résume avec les principes de chaque type.
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 notre projet nous nous intéressant aux résumés automatiques multidocuments de type abstractif.
Dans le chapitre suivant, nous allons présenter les techniques de résumé automatique, pour le type extractif et pour le type abstractif.

CHAPITRE 2

ETAT DE L'ART

2.1 INTRODUCTION :

En raison du taux croissant de données, les utilisateurs ont besoin d'informations significatives. Ainsi, il n'est pas possible pour de lire chaque document afin de trouver celui qui est utile. Le résumé de texte est devenu un outil important pour permettre aux utilisateurs de comprendre rapidement le volume important d'informations. Le système automatique de résumé de texte, l'une des applications spéciales d'exploration de données qui aide cette tâche en fournissant un résumé rapide de l'information contenue dans les documents. L'approche de résumé de texte est généralement classée en deux catégories : extractive et abstractive (Yeasmin, Tumpa, Nitu & Palash, 2017).
De nombreuses techniques de résumé de texte abstractif ont été développées pour les langues comme l'anglais, l'arabe, l'hindi (Yeasmin, Tumpa, Nitu & Palash, 2017).
Dans ce chapitre nous allons présenter les techniques de résumé automatique, pour le type extractif et pour le type abstractif, qui est lui-même classé en deux approches : l'approche structurelle et l'approche sémantique.

2.2 TECHNIQUES DE RÉSUMÉ AUTOMATIQUE :

Le résumé de texte joue un rôle important dans le domaine de l'exploration de texte et du traitement du langage naturel. Le résumé de texte vise à compresser le texte source en une forme plus courte et concise tout en préservant son contenu informationnel et sa signification globale (Dave & Jaswal, 2015). Essentiellement, les techniques de résumé de texte sont classées

comme extractives et abstractives. Les techniques d'extraction effectuent le résumé de texte en sélectionnant des phrases de documents selon certains critères. Les techniques abstractives tentent d'améliorer la cohérence entre les phrases en éliminant les redondances et en clarifiant la contestation des phrases. Les deux techniques sont utilisées pour résumer le texte soit pour un document unique ou multidocuments. La notation des phrases est la technique la plus utilisée pour le résumé de textes extractifs. Ainsi, le résumé extractif implique l'attribution d'une mesure de saillance à certaines unités (par exemple des phrases, des paragraphes) des documents et l'extraction de ceux ayant les scores les plus élevés à inclure dans le résumé. Le résumé abstractif nécessite généralement la fusion d'informations, la compression de phrases et la reformulation. Le résumé abstractif est complexe car il nécessite une analyse plus approfondie des documents source et de concepts pour la génération de résumé (El-Ghannam & El-Shishtawy, 2013).

1. (Genest & Lapalme, 2012) Ont proposé une approche totalement abstraite de résumé guidé. Cet article montre que dans le contexte du résumé guidé, une abstraction complète peut être accomplie et décrit un travail en cours qui repose sur l'extraction d'information, la sélection de contenu statistique et la génération de langage naturel.

   Le résumé guidé :

   La tâche de résumé guidé est une tâche de résumé multidocuments orientée dans laquelle une catégorie est attribuée à un groupe de 10 documents sources à résumer en 100 mots ou moins. Il y a plusieurs catégories : les accidents et les catastrophes naturelles, les attaques, la santé et la sécurité, les ressources en péril et les enquêtes/essais. Chaque catégorie est associée à une liste d'aspects à traiter dans le résumé. **Les aspects de la catégorie des attaques sont donnés ci-dessous :**

   QUOI : que s'est-il passé ? **QUAND :** date, heure, autres marqueurs de placement temporel. **OU :** emplacement physique. **PERPETRATEURS :** individus ou groupes responsables de l'attaque. **POURQUOI :** les raisons de l'attaque. **QUI A AFFECTÉ :** les victimes (décès, blessures) ou les personnes autrement affectées négativement. **DEGATS :** dégâts causés par l'attaque. **CONTRE-MESURES :** contre-mesures, efforts de secours, efforts de prévention et autres réactions.

   Approche totalement abstractive : Extraction d'information (IE) : Un grand nombre de candidats sont trouvés par les règles d'IE pour chaque aspect. Les trois ressources ont aidé à concevoir des règles d'extraction, un thésaurus et des noms et verbes sémantiquement liés.

   Sélection de contenu : Le module de sélection de contenu sélectionne les meilleurs et les envoie au module de génération. L'heuristique de base est de sélectionner le candidat

et de même pour le choix d'une préposition ou d'un verbe pour la génération.

Génération : elle prend une structure de phrase et des mots dans la forme racine en entrée et donne une phrase avec des accords résolus et des marqueurs de phrase en sortie.

2.2.1 TECHNIQUES DU RÉSUMÉ EXTRACTIF :

Le résumé de texte identifie et extrait des phrases clés du texte source et les concatène pour former un résumé de concise. Afin d'identifier les phrases clés pour le résumé, une liste des caractéristiques sont discutées dans le chapitre 1 « Critères de sélection statistiques », peut être utilisé pour la sélection des phrases clés (Khan & Salim, 2014).

2.2.2 TECHNIQUES DU RÉSUMÉ ABSTRACTIF :

Le résumé abstractif est une forme efficace de résumé par rapport au résumé extractif, car il récupère des informations à partir de plusieurs documents pour créer un résumé précis de l'information. Ceci a gagné sa popularité en raison de la capacité de développer de nouvelles phrases pour dire l'information importante des documents de texte. Le résumé abstractif affiche les informations résumées sous une forme cohérente, facilement lisible et grammaticalement correcte. La lisibilité ou la qualité linguistique est un catalyseur important pour améliorer la qualité d'un résumé (Moratanch & Chitrakala, 2016).

Quelques travaux de recherche ont abordé le résumé abstractif de document unique et multidocuments dans le milieu académie. Différentes approches et systèmes pour le résumé abstractif de document unique et multi-documents ont été discutés dans la littérature (Khan & Salim, 2014).

Le résumé abstractif est classé en deux approches : l'approche structurelle et l'approche sémantique (voir figure 2.1).

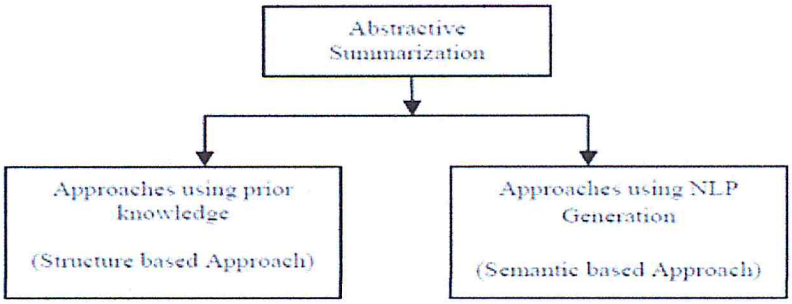

FIGURE 2.1 – Overview of Abstractive Summarization (Moratanch & Chitrakala, 2016).

1. L'approche structural :

   L'approche structurelle code les informations les plus importantes de document(s) par le biais de schémas cognitifs (Greenbacker, 2011) tels que Template, extraction Rules et d'autres structures telles que tree, ontology, lead and body phrase structure (voir figure 2.2).

   Différentes méthodes ont utilisé cette approche sont discutées comme suit (Khan & Salim, 2014).

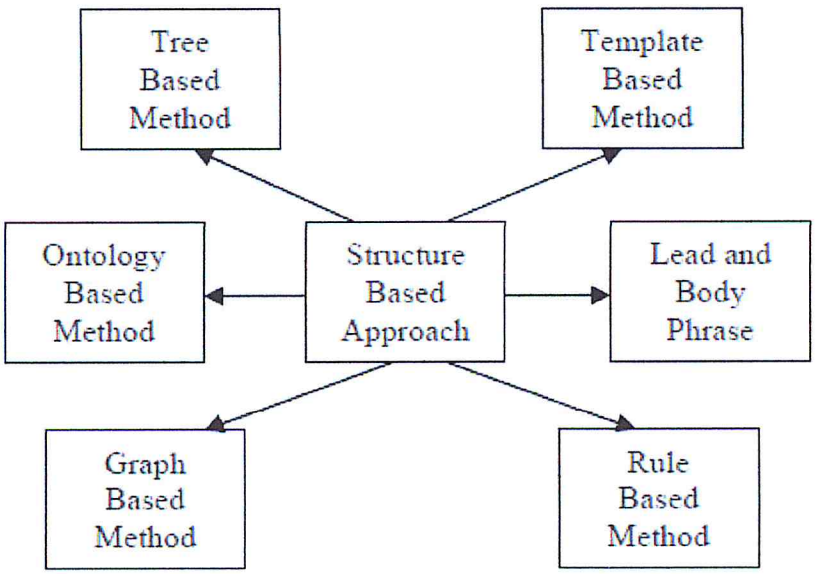

FIGURE 2.2 – Une vue sur les méthodes de l'approche structurelle (Moratanch & Chitrakala, 2016).

a) Tree based method :

Cette technique utilise un arbre de dépendance pour représenter le contenu du texte d'un document. Elle rassemble l'information tirée de plusieurs documents sous formes d'arbres de dépendances. Ces arbres sont par la suite fusionnés pour former un seul résumé.

Différents algorithmes sont utilisés pour la sélection de contenu pour un résumé, par exemple :

i. L'algorithme d'intersection de thèmes : qui permet de détecter les phrases communes, les sélectionner et les ordonner.

ii. L'algorithme qui utilise l'alignement local à travers une paire de phrases analysées : il fait l'alignement soit de bas en haut, ou bien de haut en bas pour sélectionner les phrases saillantes.

La technique utilise soit un générateur de langage, soit un algorithme de génération de résumé (Khan & Salim, 2014).

Deux approches dans la littérature, qui ont utilisés cette méthode :

i. L'approche proposée dans (Barzilay, McKeown & Elhadad, 1999) fusionne automatiquement des phrases similaires à travers des articles de presse sur le même événement. La méthode utilise la génération de langage pour produire un résumé concis. Dans cette approche, d'abord les phrases similaires sont prétraitées à l'aide d'un analyseur superficiel, puis les phrases sont mappées pour prédiquer la structure des arguments. Ensuite, le planificateur de contenu utilise un algorithme d'intersection de thèmes pour déterminer les phrases communes en comparant les structures d'argument de prédicat. Les phrases qui véhiculent des informations communes sont sélectionnées et ordonnées et certaines informations y sont également ajoutées (références temporelles, descriptions d'entités). Enfin, la phase de génération de phrases utilise le générateur de langage FUF/ SURGE pour combiner et organiser les phrases sélectionnées en nouvelles phrases récapitulatives. La principale force de cette approche est que l'utilisation d'un générateur de langage a significativement amélioré la qualité des résumés résultants, c'est-à-dire la réduction des répétitions et l'augmentation de la fluidité. Le problème avec cette approche est que le contexte de la phrase n'a pas été inclus lors de la capture de la phrase recoupée. Le

contexte est important même s'il ne fait pas partie de l'intersection (Khan & Salim, 2014).

ii. Dans un autre travail, la fusion de phrases (Barzilay & McKeown, 2005) intègre des informations dans des phrases qui se chevauchent pour générer une phrase récapitulative qui ne se chevauche pas. Dans cette approche, les arbres de dépendance sont d'abord obtenus en analysant les phrases. Un arbre de base est défini en trouvant le centroïde des arbres de dépendance.Il augmente ensuite l'arbre de base avec les sous-arbres dans d'autres phrases et enfin élague les constituants prédéfinis. La limite de cette approche est qu'il manque un modèle complet qui inclurait une représentation abstraite pour la sélection de contenu (Khan & Salim, 2014).

b) Template based method :

Cette technique utilise un modèle pour représenter un document entier. Les modèles linguistiques ou les règles d'extraction sont mis en correspondance pour identifier les extraits de texte qui seront mappés dans des emplacements de modèle. Ces extraits de texte sont des indicateurs du contenu récapitulatif (Khan & Salim, 2014).

Une approche dans la littérature, qui a utilisé cette méthode :

i. L'approche proposée dans (Harabagiu & Lacatusu, 2002) présente un système de résumé multidocuments, GISTEXTER, qui produit des résumés abstractifs de multi documents de fil de presse/journaux s'appuyant sur la sortie du système d'extraction d'information CICERO (IE). Pour extraire des informations de plusieurs documents, CICERO requiert une représentation modèle du sujet pour extraire des informations de plusieurs documents. Dans cette approche, un sujet est représenté comme un ensemble de concepts apparentés et mis en œuvre en tant que cadre ou modèle contenant des emplacements et des remplisseurs. Les modèles sont remplis d'extraits de texte importants extraits par les systèmes d'extraction d'informations. Ces extraits de texte sont utilisés pour générer des récapitulatifs cohérents et informatifs à l'aide d'un algorithme de récapitulatif multi document basé sur IE. Un avantage significatif de cette approche est que le résumé généré est hautement cohérent car il repose sur des informations pertinentes identifiées par le système IE. Cette approche ne fonctionne que si les phrases récapitulatives sont déjà présentes dans les documents source. Il ne peut pas gérer la tâche si la récapitulation multidocuments

nécessite des informations sur les similitudes et les différences entre plusieurs documents(Khan & Salim, 2014).

c) Ontology based method :

De nombreux chercheurs ont fait des efforts pour utiliser l'ontologie (base de connaissances) pour améliorer le processus de résumé. La plupart des documents sur le Web sont liés au domaine car ils traitent du même sujet ou de même événement. Chaque domaine a sa propre structure de connaissances et cela peut être mieux représenté par l'ontologie (Khan & Salim, 2014).

Plusieurs méthodes basées sur l'ontologie principalement utilisées pour le résumé de texte abstractif sont illustrées dans le tableau suivant :

La méthode	Le but	La description
OntoMetric	Aide à choisir l'ontologie appropriée pour un nouveau projet.	Compare l'importance des objectifs du projet et étudie les caractéristiques des ontologies. Obtient pour chaque ontologie candidate une mesure quantitative de son adéquation.
Natural Language Application metrics	Permet d'évaluer le contenu des ontologies en fonction de différentes métriques : Métriques de précision et de rappel. Métriques d'évaluation basées sur les coûts. Mesure de tennis. Mesure du niveau de comparaison lexicale.	Mesure pour chaque ontologie (a) combien d'éléments identifiés sont corrects et (b) combien d'éléments qui auraient été identifiés sont effectivement identifiés. Caractérise la performance en termes de coût des erreurs ou de la valeur des choses correctes. Donne une mesure de l '"ajustement" entre une ontologie et un corpus (connaissance du domaine) en utilisant un modèle d'espace vectoriel d'instances (termes). Compare le contenu de deux ontologies sans tenir compte de leur structure conceptuelle.
OntoClean	Permet d'évaluer une ontologie formelle.	Nettoie la structure taxonomique des ontologies compare l'ontologie par rapport à une structure taxonomique idéale prédéfinie pour détecter les incohérences.
EvaLexon	Aide à évaluer les ontologies créées par l'ontologie. La méthode reste au niveau linguistique.	Compare le vocabulaire des triplés extraits avec le texte d'entrée en tant que tel et avec un ensemble de mots considérés comme pertinents pour ce texte.

TABLE 2.1 – Résumé abstractif basé sur l'ontologie (Yeasmin, Tumpa, Nitu & Palash, 2017).

La littérature utilisant cette méthode est discutée comme suit :

i. L'ontologie floue (fuzzy ontology) avec des concepts flous (fuzzy concepts) est introduite pour le résumé des nouvelles chinoises (Lee, Jian & Huang, 2005) afin de modéliser des informations incertaines et donc de mieux décrire la connaissance du domaine. Dans cette approche, les experts du domaine définissent d'abord l'ontologie de domaine pour les événements d'actualité. Ensuite, la phase de prétraitement des documents produit les termes significa-

tifs du corpus de nouvelles et du dictionnaire de nouveaux chinois. Ensuite, le classificateur de termes classifie les termes significatifs sur la base des événements des nouvelles. Pour chaque concept flou de l'ontologie floue, la phase d'inférence floue génère les degrés d'appartenance. Un ensemble de degrés d'appartenance de chaque concept flou est associé à divers événements de l'ontologie de domaine. Le résumé des nouvelles est fait par l'agent de nouvelles basé sur l'ontologie floue. L'avantage de cette approche est qu'elle exploite l'ontologie floue pour gérer des données incertaines que l'ontologie de domaine simple ne peut pas le faire. Cette approche a plusieurs limites. Premièrement, l'ontologie du domaine, le dictionnaire chinois et le corpus de nouvelles doivent être définis par un expert du domaine, ce qui prend beaucoup de temps. Deuxièmement, cette approche est limitée aux nouvelles chinoises et pourrait ne pas être applicable aux nouvelles en anglais (Khan & Salim, 2014).

d) Lead and body phrase method :

Cette méthode est basée sur les opérations de phrases (insertion et substitution) qui ont le même morceau de tête syntaxique dans les lead et body phrases afin de réécrire la lead phrase.

L'information la plus importante est succinctement mentionnée dans les lead phrases. Généralement, elles font office de phrases d'ouverture. Les noms propres et les détails sont parfois évités en faveur d'expressions plus abstraites telles que « grande compagnie d'assurance » par exemple.

Les lead phrases sont par la suite détaillées dans les body phrases en répondant aux questions : qui, quoi, quand, où, pourquoi et comment, et les noms propres qui ne sont pas mentionnés dans les leads phrases apparaissent ici.

Les informations nécessaires qui n'ont pas été couvertes dans les lead et body phrases sont placées dans les suppléments phrases.

La littérature utilisant cette méthode est discutée comme suit :

i. Une approche abstractive proposée par (Tanaka, Kinoshita, Kobayakawa, Kumano & Kato, 2009) permet de réviser les lead phrases dans une émission de nouvelles. Cette approche n'utilise pas la relation de coréférence des phrases nominales (NPs). Dans cette approche, d'abord les mêmes morceaux (également appelés déclencheurs) sont recherchés dans les lead et body phrases. Ensuite, les phrases maximales (candidats à la révision) de chaque déclencheur sont identifiées et alignées en utilisant la métrique de similarité. La substitution

de la body phrase pour la lead phrase a lieu si la body phrase a une phrase correspondante dans la lead phrase et la body phrase est plus riche en informations. L'insertion de la body phrase dans la lead phrase a lieu, si une body phrase n'a pas d'équivalent dans la lead phrase. L'avantage potentiel de cette méthode est qu'elle a trouvé des révisions sémantiquement appropriées pour réviser une lead phrase. Cette méthode a quelques faiblesses. Premièrement, les erreurs d'analyse dégradent la complétude de la forme, comme la grammaticalité et la répétition. Deuxièmement, il se concentre sur les techniques de réécriture, et il manque un modèle complet qui inclurait une représentation abstraite pour la sélection du contenu (Khan & Salim, 2014).

e) Rule based method :

Dans cette méthode, les documents à résumer sont représentés en termes de catégories et d'une liste d'aspects. Le module de sélection de contenu sélectionne le meilleur candidat parmi ceux générés par les règles d'extraction d'informations pour répondre à un ou plusieurs aspects d'une catégorie. Enfin, les modèles de génération sont utilisés pour générer des phrases récapitulatives (Khan & Salim, 2014).

i. La méthodologie de (Genest & Lapalme, 2012) génère des résumés abstractifs courts et bien écrits à partir de groupes d'articles de presse sur le même événement. La méthodologie est basée sur un schéma d'abstraction. Le schéma d'abstraction utilise un module d'extraction d'informations basé sur des règles, des heuristiques de sélection de contenu et un ou plusieurs modèles pour la génération de phrases. Chaque schéma d'abstraction traite d'un thème ou d'une sous-catégorie. Afin de générer des règles d'extraction pour le schéma d'abstraction, plusieurs verbes et noms ayant une signification similaire sont déterminés et la position syntaxique des rôles est également identifiée. Le module d'extraction d'information (IE) trouve plusieurs règles candidates pour chaque aspect de la catégorie. Sur la base de la sortie du module IE, le module de sélection de contenu sélectionne la meilleure règle candidate pour chaque aspect et la transmet au module de génération de résumé. Ce module forme un résumé du texte en utilisant des modèles de génération conçus pour chaque schéma d'abstraction. Le point fort de cette méthode est qu'elle a le potentiel de créer des résumés avec une plus grande densité d'information. Le principal inconvénient de cette méthodologie est que toutes les règles et les schémas sont

écrits manuellement, ce qui est fastidieux et prend du temps (Khan & Salim, 2014).

f) Graph based method :

De nombreux chercheurs utilisent une structure graphique de données (appelé Opinosis- Graph) pour représenter la langue de texte. La nouveauté introduite dans le système est que chaque nœud représente une unité de mot qui représente la structure de phrases pour diriger bords (Moratanch & Chitrakala, 2016).

i. (Lloret & Manuel, 2011) Ont concentrés sur la génération de résumés abstractifs en utilisant une méthode basée sur un graphe de mots. L'approche combine des informations extractives et abstractives pour générer des résumés. Cette méthode compresse et fusionne des informations basées sur la méthode des graphes de mots générant ainsi des résumés. Les mots dans le document forment un ensemble de sommets dans le graphe et le bord qui représente la relation adjacente entre deux mots. Une fonction de pondération a été formulée pour définir l'importance du seuil en utilisant la valeur Page Rank. L'algorithme de chemin le plus court est utilisé car il donne une phrase de longueur minimale avec plus d'informations provenant des nœuds pertinents dans le graphe. Le contenu important est trouvé en utilisant l'approche Compendium Text Summarization par deux méthodes :

A. Un ensemble de phrases est donné comme entrée du graphe de mots et ensuite il est transmis au compendium.

B. Sélectionnant le contenu important du document source puis appliquant la méthode du graphe.

Auteur/Année	Techniques/Méthodes	Représentation du Texte	Sélection du Contenu	Génération du résumé
Barzilay and McKeown,1999	Treebased	arbre de dépendance	Algorithme d'intersection de thèmes	Générateur de langage FUF/SURGE
Harabagiu and Lacatusu, 2002	Template based	Template/Frame ayant des emplacements et des remplisseurs	Modèles linguistiques ou règles d'extraction	Algorithme de résumé basé sur IE
Lee and Jian, 2005	Ontologybased	Fuzzyontology (L'ontologie floue)	Classificateur	News agent (l'agent de nouvelles)
Barzilay and McKeown, 2005	Treebased	Arbre de dépendance	Algorithme utilise l'alignement local à travers une paire de phrases analysées	Algorithme pour réutiliser et modifier des phrases à partir de phrases d'entrée
Tanaka and Kinoshita, 2009	Lead and Body phrase	Lead, body and supplement structure	Révision des candidats (Phrases maximales de même morceau de tête dans les Lead et Body phrases)	Opérations d'insertion et de substitution sur les phrases
Elena Lirot, 2011	Graph based	Compresse et fusionne	Graphe de mots	Compression de phrases de longueur minimale
Genest and Lapalme, 2012	Rule based	Catégories et Aspects	Règles d'extraction	Modèles de génération

TABLE 2.2 – Montre une étude comparative sur les méthodes du résumé abstractif pour l'approche structurelle basée sur la représentation du texte, la sélection du contenu et la génération du résumé (Khan & Salim, 2014),(Moratanch & Chitrakala, 2016).

g) Abstractive Text Summarization using Sequence-to-sequence Recurrent Neural Networks "RNNs" and Beyond :

Dans ce travail, (Nallapati, Zhou, Gulcehre, Xiang & al., 2016) ont modélisé le

résumé de texte abstractif à l'aide de réseaux de neurone récurrents de codeur-décodeur attentionnel, et ont montré qu'ils atteignent des performances de pointe sur deux corpus différents. Ils ont proposé plusieurs nouveaux modèles qui traitent des problèmes critiques de résumé qui ne sont pas modélisés adéquatement par l'architecture de base, comme la modélisation des mots-clés, la capture de la structure hiérarchique des phrases et l'émission de mots rares ou invisibles. Ils ont montré aussi que les modèles proposés dans leur travail contribuent à l'amélioration de la performance. Ils ont proposé également un nouveau jeu de données composé de résumés à plusieurs phrases, et ont établi des repères de performance pour d'autres recherches.

i. Encoder-Decoder RNN with Attention and Large Vocabulary Trick :

   Ce modèle correspond au modèle de traduction par machine neurale utilisé dans (Bahdanau, Cho & Bengio, 2014). Le codeur est constitué d'un GRU-RNN bidirectionnel (Chung, Gulcehre, Cho & Bengio, 2014), tandis que le décodeur est constitué d'un GRU-RNN unidirectionnel avec la même taille de la couche cachée que celui du codeur, et d'un mécanisme d'attention sur la source de la couche cachée et une couche soft-max sur le vocabulaire cible pour générer des mots. Dans l'intérêt de l'espace, il renvoie le lecteur à l'article original pour un traitement détaillé de ce modèle. En plus du modèle de base, ils sont également adaptés au problème du résumé, le «trick» du grand vocabulaire (LVT) décrit dans (Jean, Cho, Memisevic & Bengio, 2014). Dans cette approche, le vocabulaire du décodeur de chaque mini-lot est limité aux mots dans les documents sources de ce lot. De plus, les mots les plus fréquents dans le dictionnaire cible sont ajoutés jusqu'à ce que le vocabulaire atteigne une taille fixe. Le but de cette technique est de réduire la taille de la couche soft-max du décodeur qui est le principal goulot d'étranglement de calcul. De plus, cette technique accélère également la convergence en focalisant l'effort de modélisation uniquement sur les mots essentiels à un exemple donné. Cette technique est particulièrement bien adaptée au résumé car une grande partie des mots du résumé provient du document source dans tous les cas (Nallapati, Zhou, Gulcehre, Xiang & al., 2016).

ii. Capturing Keywords using Feature-rich Encoder :

   En résumé, l'un des principaux défis consiste à identifier les concepts clés et les entités clés du document, autour desquels l'histoire tourne. Pour atteindre cet objectif, ils doivent aller au-delà de la représentation basée sur l'intégration

de mots « word-embeddings-based representation » du document d'entrée et capturer des caractéristiques linguistiques supplémentaires telles que des parties de discours (part of speech tags or POS tags), des étiquettes d'entités nommées (named entity recognition or NER) et des statistiques TF et IDF des mots. Ils créent donc des matrices d'intégration basées sur la recherche pour le vocabulaire de chaque type d'étiquette, similaire aux intégrations pour les mots. Pour les fonctions continues telles que TF et IDF, ils les convertissent en valeurs catégorielles en les discrétisant dans un nombre fixe de classes, et utilisent des représentations à une seule chaîne pour indiquer le numéro de groupe auquel elles appartiennent. Cela leur permet de les mapper dans une matrice d'intégration comme n'importe quel autre type d'étiquetage. Enfin, pour chaque mot du document source, ils cherchent simplement ses intégrerions à partir de toutes les étiquettes associées dans les matrices d'intégration « embedding matrices » et les concaténent en un seul vecteur long, comme le montre la figure 2.3. Du côté de la cible, ils continuent à utiliser uniquement des intégrations basées sur les mots « word-based embeddings » en tant que représentation (Nallapati, Zhou, Gulcehre, Xiang & al., 2016).

Part of speech tags or POS tags : elle consiste à associer aux mots d'un texte les informations grammaticales correspondantes comme « la partie du discours », « le genre »...etc.

Named entity recognition or NER : c'est une sous-tâche de l'activité d'extraction d'information dans des corpus documentaires, elle consiste à rechercher des objets textuels catégorisables dans des classes telles que « noms de personnes », « nom d'organisations ou d'entreprises », « noms de lieux », « quantités », « distances », « valeurs », « dates »... etc.

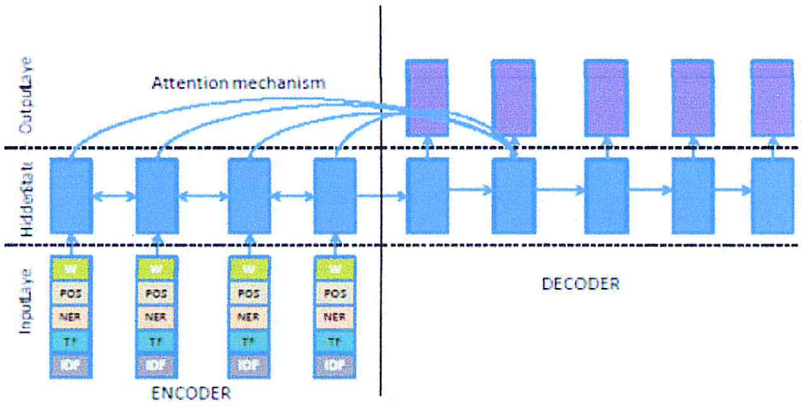

FIGURE 2.3 – Feature-rich-encoder : Utilisation d'un vecteur d'intégration « embedding vector » pour POS, NER et les valeurs TF et IDF discrétisées, qui sont concaténées avec des intégrations basées sur des mots « word-basedembeddings » comme entrée du codeur (Nallapati, Zhou, Gulcehre, Xiang & al., 2016).

iii. Modeling Rare/Unseen Words using Switching Generator-Pointer :

Souvent, dans le résumé, les mots clés ou les entités nommées dans un document de test qui sont au centre du résumé peuvent en fait être invisibles ou rares en ce qui concerne les données d'entrainement. Puisque le vocabulaire du décodeur est fixé au moment d'entraînement, il ne peut pas émettre ces mots invisibles. Au lieu de cela, la façon la plus courante de gérer ces mots hors vocabulaire (Out-Of-Vocabulary « OOV ») consiste à émettre un jeton «UNK» pour garder l'emplacement de ces mots. Cependant, cela n'aboutit pas à des résumés lisibles. En résumé, une manière intuitive de gérer de tels mots OOV consiste simplement à pointer leur emplacement dans le document source. Dans ce modèle ils ont modélisé cette notion en utilisant la nouvelle architecture « switching decoder / pointer » qui est représentée graphiquement sur la figure 2.4. Dans ce modèle, le décodeur est équipé d'un « commutateur » qui décide entre utiliser le générateur ou un pointeur à chaque pas de temps. Si le commutateur est activé, le décodeur produit un mot à partir de son vocabulaire cible de la manière normale. Cependant, si le commutateur est désactivé, « le cas pour un mot invisible », le décodeur génère à la place, un pointeur pour pointer une position d'ensemble des positions du mot dans le document source. Le mot à l'emplacement du pointeur est ensuite copié dans le résumé. Le commutateur est modélisé comme une fonction d'activation sigmoïde sur

une couche linéaire basée sur l'ensemble du contexte disponible à chaque pas de temps, comme indiqué ci-dessous (Nallapati, Zhou, Gulcehre, Xiang & al., 2016).

$$P(s_i = 1) = \sigma\left(v^s.\left(W_h^s h_i + W_e^s E[O_{i-1}] + W_c^s c_i + b^s\right)\right)$$

Où $P(s_i = 1)$ est la probabilité de l'activation du commutateur au ième pas de temps du décodeur, h_i est la couche cachée, $E[O_{i-1}]$ est le vecteur d'intégration « embedding vector » de l'émission du pas de temps précédent, c_i est le vecteur de contexte pondéré par l'attention, et W_h^s ; W_e^s ; W_c^s ; b^s et v^s sont les paramètres du commutateur. Ils ont utilisé la distribution de l'attention sur les positions des mots dans le document comme une distribution pour échantillonner le pointeur.

$$P_i^a(j) \propto exp\left(v^a.\left(W_h^a h_{i-1} + W_e^a E[O_{i-1}] + W_c^a h_j^d + b^a\right)\right)$$

$$p_i = \text{argmax}_j\left(P_i^a(j)\right) \quad for\ j \in \{1,\ldots,N_d\}$$

Dans l'équation ci-dessus, p_i est la valeur du pointeur à la ième position du mot dans le résumé, échantillonnée à partir de la distribution d'attention P_i^a sur les positions de mots de document $j \in \{1,\ldots,N_d\}$, où $P_i^a(j)$ est la probabilité que le ième pas de temps dans le décodeur pointe vers la jième position dans le document, et h_j^d est la couche cachée du codeur à la position j.

Au moment d'entrainement, ils ont fourni au modèle des informations explicites sur le pointeur chaque fois que le mot de résumé n'existe pas dans le vocabulaire cible. Lorsque le mot OOV du résumé apparait dans plusieurs positions de document, la première occurrence sera choisie. Au moment d'entrainement, ils ont optimisé la log-vraisemblance conditionnelle montrée ci-dessous, avec des pénalités de régularisation supplémentaires.

$$\log P(y \mid x) = \sum_i \left(g_i \log\{P(y_i \mid y_{-i}, x) P(s_i)\} + (1 - g_i) \log\{P(p(i) \mid y_{-i}, x)(1 - P(s_i))\}\right)$$

Où y et x sont les mots de résumé et de document respectivement, g_i est une fonction d'indicateur qui est mise à 0 chaque fois que le mot à la position i dans le résumé est Invisible (OOV) par rapport au vocabulaire du décodeur. Au moment du test, le modèle décide automatiquement à chaque pas de temps de générer ou de pointer, en fonction de la probabilité de commutation estimée $P(s_i)$. Ils ont simplement utilisé l'$argmax$ de la probabilité postérieure de génération ou de pointage pour générer la meilleure sortie à chaque pas de temps.

Le mécanisme de pointeur peut être plus robuste dans la manipulation de mots rares car il utilise la représentation de la couche cachée de mots rares du codeur pour décider quel mot du document doit être pointé. Puisque la couche cachée dépend de l'ensemble du contexte du mot, le modèle est capable de pointer avec précision vers des mots invisibles, bien qu'ils n'apparaissent pas dans le vocabulaire cible (voir figure 2.4).

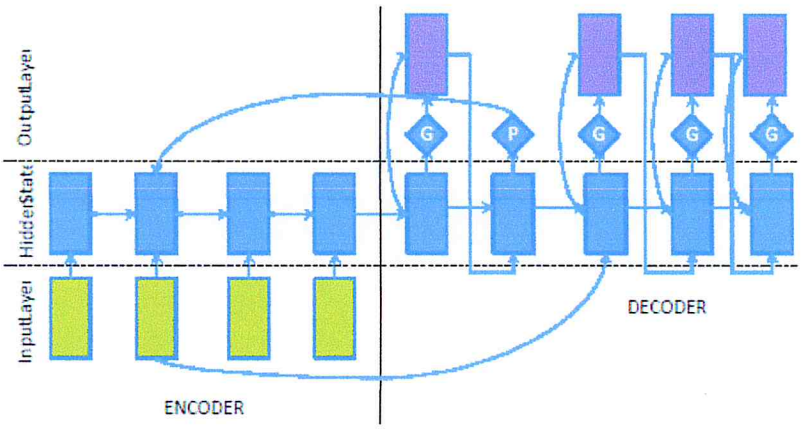

FIGURE 2.4 – Switching generator/pointer model : Lorsque le commutateur affiche 'G', le générateur traditionnel constitué de la couche softmax est utilisé pour produire un mot, et quand il montre 'P', le réseau de pointeurs est activé pour copier le mot de l'un des positions du document source. Lorsque le pointeur est activé, l'intégration « embedding » de la source est utilisée comme entrée pour le pas de temps suivant, comme indiqué par la flèche du codeur au décodeur en bas (Nallapati, Zhou, Gulcehre, Xiang & al., 2016).

iv. Capturing Hierarchical Document Structure with Hierarchical Attention :

Dans les jeux de donnée où le document source est très long, en plus d'identifier les mots-clés dans le document, il est également important d'identifier les phrases clés à partir desquelles le résumé peut être établi. Ce modèle vise à capturer cette notion de deux niveaux d'importance en utilisant deux RNN bidirectionnels du côté de la source, l'un au niveau du mot et l'autre au niveau de la phrase. Le mécanisme d'attention fonctionne simultanément aux deux niveaux. L'attention au niveau du mot est de nouveau pondérée par l'attention correspondante au niveau de la phrase et renormalisée comme indiqué ci-dessous (Nallapati, Zhou, Gulcehre, Xiang & al., 2016) :

$$P^a(j) = \frac{P_w^a(j)\, P_s^a(s(j))}{\sum_{k=1}^{N_d} P_w^a(k) P_s^a(s(k))}$$

Où $P_w^a(j)$ est le poids d'attention au niveau du mot à la jième position du document source, et $s(j)$ est l'ID de la phrase à la jième position du mot, $P_s^a(l)$ est le poids d'attention au niveau de la phrase pour la lième phrase dans le document source, N_d est le nombre de mots dans le document source, et $P^a(j)$ est l'attention redimensionnée à la jième position du mot. L'attention redimensionnée est ensuite utilisée pour calculer le vecteur de contexte à pondération attentionnelle qui va être une entrée de la couche cachée du décodeur. En outre, ils ont concaténé des intégrations « embeddings » positionnels supplémentaires à la couche cachée du RNN au niveau de la phrase afin de modéliser l'importance positionnelle des phrases dans le document. Cette architecture modélise donc des phrases clés ainsi que des mots-clés dans ces phrases conjointement. Une représentation graphique de ce modèle est présentée à la figure 2.5.

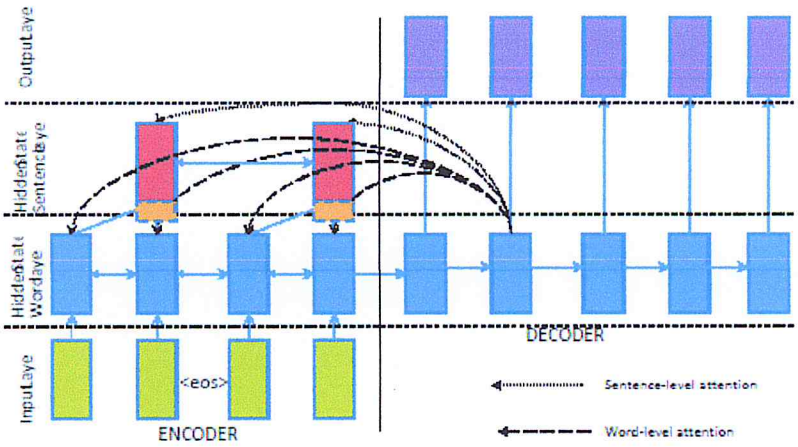

FIGURE 2.5 – Hierarchical encoder with hierarchical attention : les poids d'attention au niveau du mot, représentés par les flèches pointillées, sont redimensionnés par les poids d'attention de niveau de phrases correspondantes, représentés par les flèches des points. Les cases en pointillés au bas de la couche supérieure RNN représentent des intégrations « embeddings » positionnelles de niveau phrase concaténées aux couches cachées correspondantes (Nallapati, Zhou, Gulcehre, Xiang & al., 2016).

h) Deep Recurrent Generative Decoder for Abstractive Text Summarization :

Un nouveau Framework est proposé pour le résumé de texte abstractif basée sur un modèle de codeur-décodeur séquence à sequence orienté « sequence-to-sequence oriented encoder-decoder » équipé d'un décodeur génératif récurrent profond « deep recurrent generative decoder » (DRGN). L'information de structure latente impli-

quée dans les résumés cibles est apprise sur la base d'un modèle aléatoire latent récurrent pour améliorer la qualité de résumé. L'inférence variationnelle neurale est utilisée pour traiter l'inférence postérieure intraitable pour les variables latentes récurrentes. Les résumés abstractifs sont générés en fonction à la fois des variables latentes génératives « generative latent variables » et des états déterministes discriminatifs « discriminative deterministic states ». Des expériences approfondies sur certains jeux de données de référence dans différentes langues montrent que DRGN réalise des améliorations par rapport aux méthodes les plus récentes (Li, Lam, Bing & Wang, 2017).

Intuitivement, si ils peuvent incorporer l'information de structure latente des résumés dans le modèle de résumé abstractif, cela améliorera la qualité des résumés générés. Cependant, très peu d'ouvrages existants considèrent spécifiquement les informations de structure latente des résumés dans leurs modèles de résumé. Bien qu'un Framework de séquence à séquence très populaire basé sur un réseau de neurones ait été proposé pour traiter le problème de résumé abstractif (Lopyrev, 2015),(Rush, Chopra & Weston, 2015),(Nallapati, Zhou, Gulcehre, Xiang & al., 2016), le calcul des états de décodage interne est entièrement déterministe. Les transformations déterministes dans ces modèles discriminatifs conduisent à des limitations sur la capacité de représentation de l'information de structure latente. (Miao & Blunsom, 2016) Ont étendu le Framework de séquence à séquence et proposé un modèle génératif pour capturer l'information récapitulative latente, mais ils n'ont pas considéré les dépendances récurrentes dans leur modèle génératif conduisant à une capacité de représentation limitée.

Pour résoudre les problèmes mentionnés ci-dessus, ce modèle conçoit un nouveau Framework basé sur un modèle codeur-décodeur orienté séquence à séquence équipé d'un composant de modélisation de structure latente. Des auto-encodeurs varationnels (VAE) (Kingma & Welling, 2013), (Rezende, Mohamed & Wierstra, 2014) sont utilisés comme modèle de base pour ce Framework génératif qui peut gérer le problème d'inférence associé à la modélisation générative complexe. Cependant, le Framework standard des VAE n'est pas conçu pour les tâches liées à la modélisation de séquences. Inspiré par (Chung, Kastner, Dinh, Goel, Courville & Bengio, 2015), des dépendances historiques sont ajoutées sur les variables latentes des VAE et proposons un décodeur génératif récurrent profond (DRGD) pour la modélisation de structure latente. Ensuite, le décodeur déterministe discriminatif standard et le décodeur génératif récurrent sont intégrés dans un Framework

de décodage unifié. Les résumés cibles seront décodés sur la base des variables déterministes discriminatives et de l'information structurelle latente générative. Tous les paramètres neuraux sont appris par rétropropagation dans un paradigme d'entraînement de bout en bout.

Les principales contributions de ce Framework sont résumées comme suit (Li, Lam, Bing & Wang, 2017) :

Ils ont proposé un modèle codeur-décodeur de séquence à séquence orienté équipée d'un décodeur génératif récurrent profond (DRGD) pour modéliser et apprendre l'information de structure latente impliquée dans les résumés cibles des données d'apprentissage. L'inférence variationnelle neurale est utilisée pour traiter l'inférence postérieure intraitable pour les variables latentes récurrentes.

Les informations structurelles latentes génératives et les variables déterministes discriminantes sont considérées conjointement dans le processus de génération des résumés abstractifs.

Les résultats expérimentaux de certains jeux de données de référence dans différentes langues montrent que ce Framework atteint de meilleures performances que les modèles de pointe.

i. Aperçu :

   Comme le montre la figure 2.6, le Framework de base de cette approche (Li, Lam, Bing & Wang, 2017) est un Framework codeur-décodeur basé sur le réseau de neurone pour l'apprentissage de séquence à séquence. L'entrée est une séquence de longueur variable $X = \{x1, x2, \ldots, xm\}$ représentant le texte source. Le mot embedding x_t est initialisé aléatoirement et appris pendant le processus d'optimisation. La sortie est également une séquence $Y = \{y1, y2, \ldots, yn\}$ qui représente les résumés abstractifs générés. L'unité récurrente gérée « Gated Recurrent Unit » (GRU) (Cho, Van Merrienboer, Gulcehre, Bahdanau, Bougares, Schwenk & Bengio, 2014) est utilisée comme composant de modélisation de séquence de base pour le codeur et le décodeur. Pour la modélisation de structures latentes, des dépendances historiques sont ajoutées aux variables latentes des VAE et un décodeur génératif récurrent profond (DRGD) est proposé pour distiller les structures latentes complexes impliquées dans les résumés cibles des données d'entrainement. Enfin, les résumés abstractifs seront décodés en fonction à la fois des variables déterministes discriminantes H et de l'information structurelle latente générative Z.

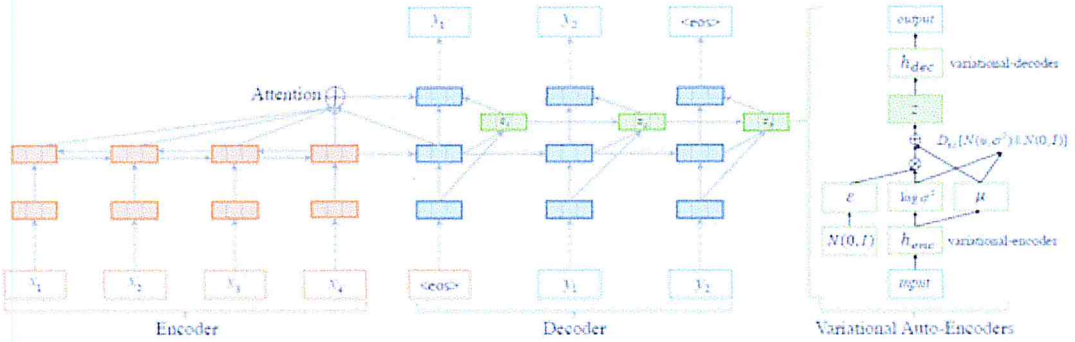

FIGURE 2.6 – Le décodeur génératif à récurrence profonde (DRGD) pour la modélisation de structure latente (Li, Lam, Bing & Wang, 2017).

ii. Décodeur génératif récurrent :

Supposons que nous avons obtenu la représentation du texte source $h^e \in R^{k_h}$. Le but du décodeur est de traduire ce code source h^e en des séries des couches cachées $\{h_1^d, h_2^d, \ldots, h_n^d\}$, puis rétablir ces couches cachées dans une séquence de mots et générer le résumé.

Pour les décodeurs récurrents standards, à chaque instant t, la couche cachée $h_t^d \in R^{k_h}$ est calculée en utilisant le symbole d'entrée dépendant $y_{t-1} \in R^{k_w}$ et la couche cachée précédente h_{t-1}^d :

$$h_t^d = f\left(y_{t-1}, h_{t-1}^d\right) \tag{2.1}$$

Où f(·) est un réseau neuronal récurrent tel que vanilla RNN , Long Short-Term Memory (LSTM) (Hochreiter & Schmidhuber, 1997), et Gated Recurrent Unit (GRU) (Cho, Van Merriënboer, Bahdanau & Bengio, 2014). Quel que soit celui que nous utilisons pour f (·), l'opération de transformation commune est la suivante :

$$h_t^d = g\left(W_{yh}^d y_{t-1} + W_{hh}^d h_{t-1}^d + b_h^d\right) \tag{2.2}$$

Où $W_{yh}^d \in R^{k_h \times k_w}$ et $W_{hh}^d \in R^{k_h \times k_h}$ sont les matrices de transformation linéaire. b_h^d Est le biais. k_h Est la dimension des couches cachées, et k_w est la dimension du mot intégrations « embeddings ». g(·) est la fonction d'activation non linéaire. A partir de l'équation (2.2), nous pouvons voir que toutes les transformations sont déterministes, ce qui conduit à une couche cachée récurrente

déterministe h_t^d. De leurs investigations, ils ont trouvé que le pouvoir de représentation de telles variables déterministes est limité. Certaines structures latentes plus complexes dans les résumés cibles, telles que les caractéristiques syntaxiques de haut niveau et les sujets latents, ne peuvent pas être modélisées efficacement par les opérations et les variables déterministes (Li, Lam, Bing & Wang, 2017).

Récemment, un modèle génératif appelé VAE (Variational Auto-Encoders) (Kingma, & Welling, 2013), (Rezende, Mohamed & Wierstra, 2014) montrent une forte capacité à modéliser des variables aléatoires latentes et améliore la performance des tâches dans différents domaines tels que la génération de phrases (Bowman, Vilnis, Vinyals, Dai, Jozefowicz & Bengio, 2015) Et la génération d'images (Gregor, Danihelka, Graves, Rezende & Wierstra, 2015). Cependant, les VAE standards ne sont pas conçus pour la modélisation directe de la séquence. Inspiré par (Chung, Kastner, Dinh, Goel, Courville & Bengio, 2015), ils ont étendu les VAE standard en introduisant les dépendances de variables latentes historiques pour les rendre capable de modéliser des données de séquence. Le Framework de modélisation de structures latentes proposé peut être vu comme un modèle génératif de séquence qui peut être divisé en deux parties : l'inférence (encodeur variationnel) et la génération (décodeur variationnel).

Comme indiqué dans le composant décodeur de la figure 2.6, l'entrée des VAE d'origine contient uniquement la variable observée y_t, et le codeur variationnel peut la mapper à une variable latente $z \in R^{k_z}$, qui peut être utilisée pour reconstruire l'entrée originale. Pour la tâche de résumé, dans le composant de décodeur de séquence, l'information de structure latente précédente doit être considérée pour construire des représentations plus efficaces pour la génération de l'état suivant.

Pour l'étape d'inférence, le codeur variationnel peut mapper la variable observée $y_{<t}$ et l'information de structure latente précédente $z_{<t}$ à la distribution de probabilité postérieure de la variable de structure latente $p_\theta(z_t \mid y,\ z_{<t})$. Il est évident qu'il s'agit d'un processus d'inférence récurrent dans lequel z_t contient l'information historique de structure latente dynamique. Comparé au processus d'inférence variationnelle $p_\theta(z_t \mid y_t)$ du modèle VAE typique, le Framework récurrent peut extraire des caractéristiques de structure latente plus complexes et plus efficaces impliquées dans les données de séquence.

Pour le processus de génération, basé sur la variable de structure latente z_t, le mot cible y_t à l'instant t est tiré d'une distribution de probabilité conditionnelle $p_\theta(y_t \mid z_t)$. Le but est de maximiser la probabilité de chaque résumé généré $Y = \{y1, y2, \ldots, yt\}$ en basant sur le processus de génération selon :

$$p_{\theta(y)} = \prod_{t=1}^{T} \int p_\theta(y_t \mid z_t) p_\theta(z_t) d_{z_t} \tag{2.3}$$

Dans le but de résoudre l'intégrale intraitable de la vraisemblance marginale comme montré dans l'équation (2.3), un modèle de reconnaissance $q_{\phi(z_t|y_{<t},\ z_{<t})}$ est introduit comme une approximation du vrai postérieur intraitable $p_{\theta(z_t|y_{<t},\ z_{<t})}$. Les paramètres du modèle de reconnaissance ϕ et les paramètres du modèle génératif θ peuvent être appris conjointement. L'objectif est de réduire la divergence Kulllback-Leibler (KL) entre $q_\phi(z_t \mid y_{<t},\ z_{<t})$ et $p_\theta(z_t \mid y_{<t},\ z_{<t})$:

$$D_{KL}\left[q_\phi(z_t \mid y_{<t},\ z_{<t}) || p_\theta(z_t \mid y_{<t},\ z_{<t})\right] = \int_z q_\phi(z_t \mid y_{<t},\ z_{<t})\ log \frac{q_\phi(z_t \mid y_{<t},\ z_{<t})}{p_\theta(z_t \mid y_{<t},\ z_{<t})} d_z$$

$$= E_{q_{\phi(z_t|y_{<t},\ z_{<t})}}\left[\log q_\phi(z_t \mid .) - \log p_\theta(z_t \mid .)\right]$$

Où · désigne les variables conditionnelles $y_{<t}$ et $z_{<t}$. La règle de Bayes est appliquée à $p_\theta(z_t \mid y_{<t},\ z_{<t})$, et nous pouvons extraire le $\log p_\theta(z)$ de l'attente, transférer le terme d'attente $E_{q_{\phi(z_t|y_{<t},\ z_{<t})}}$ à la divergenceKL, et réorganiser tous les termes. En conséquence, ce qui suit est valable :

$$\log p_\theta(y_{<t}) = D_{KL}\left[q_\phi(z_t \mid y_{<t},\ z_{<t}) || p_\theta(z_t \mid y_t,\ z_{<t})\right] + E_{q_{\phi(z_t|y_{<t},\ z_{<t})}}[\log p_\theta(y_{<t} \mid z_t)] -$$

$$D_{KL}\left[q_\phi(z_t \mid y_{<t},\ z_{<t}) || p_\theta(z_t)\right] \tag{2.4}$$

Soit $\Gamma(\theta, \varphi, y)$ les deux derniers termes de la partie droite de l'équation (2.4) :

$$\Gamma(\theta, \varphi, y) =$$

$$E_{q_{\phi(z_t|y_{<t},\ z_{<t})}}\left\{\sum_{t=1}^{T} \log p_\theta(y_t \mid z_t) - D_{KL}\left[q_\phi(z_t \mid y_{<t},\ z_{<t}) || p_\theta(z_t)\right]\right\} \tag{2.5}$$

Puisque le premier terme de divergence KL de l'équation (2.4) est non négatif, nous avons $\log p_\theta(y_{<t}) \geq \Gamma(\theta, \phi, y)$ signifiant que $\Gamma(\theta, \phi, y)$ est une borne inférieure (l'objectif à maximiser) sur la vraisemblance marginale. Afin de différencier et d'optimiser la borne inférieure $\Gamma(\theta, \phi, y)$, en suivant l'idée centrale des VAE, nous utilisons un Framework de réseau neuronal pour le codeur probabiliste $q_\phi(z_t \mid y_{<t}, z_{<t})$ pour une meilleure approximation.

iii. Génération de Résumé Abstractif :

Ils ont conçu également un Framework basé sur un réseau de neurones pour conduire l'inférence variationnelle et la génération pour le composant décodeur génératif récurrent similaire à des conceptions dans des travaux antérieurs (Kingma & Welling, 2013), (Rezende, Mohamed & Wierstra, 2014), (Gregor, Danihelka, Graves, Rezende & Wierstra, 2015). Le composant codeur et le composant décodeur sont intégrés dans un Framework unifié de résumé abstractif. Considérant que GRU a des performances comparables mais avec moins de paramètres et un calcul plus efficace, ils ont utilisé GRU comme modèle récurrent de base qui met à jour les variables en fonction des opérations suivantes (Li, Lam, Bing & Wang, 2017) :

$$r_t = \sigma(W_{xr}x_t + W_{hr}h_{t-1} + b_r)$$

$$z_t = \sigma(W_{xz}x_t + W_{hz}h_{t-1} + b_z)$$

$$g_t = \tanh(W_{xh}x_t + W_{hh}(r_t \odot h_{t-1}) + b_h)$$

$$h_t = z_t \odot h_{t-1} + (1 - z_t) \odot g_t$$

Où r_t est la porte de réinitialisation, z_t est la porte de mise à jour. $\odot$ désigne la multiplication par élément. tanh est la fonction d'activation tangente hyperbolique.

Comme indiqué dans le bloc gauche de la figure 2.6, le codeur est conçu sur la base de réseaux neuronaux récurrents bidirectionnels. Soit x_t le vecteur d'intégration de mot du $t$ième mot dans la séquence de source. GRU mappe x_t et la couche cachée précédente h_{t-1} à la couche cachée actuelle h_t dans la direction d'alimentation vers l'avant et arrière-avant respectivement :

$$\overrightarrow{h_t} = GRU\left(x_t, \overrightarrow{h_{t-1}}\right)$$

$$\overleftarrow{h_t} = GRU(x_t, \overleftarrow{h_{t-1}}) \tag{2.6}$$

Ensuite, la couche cachée finale $h_t^e \in R^{2k_h}$ est concaténé en utilisant les couches cachées des deux directions : $h_t^e = \overrightarrow{h_t} || \overleftarrow{h_t}$ Comme indiqué dans le bloc central de la figure 2.6, le décodeur est constitué de deux composantes : le décodage déterministe discriminatif et la modélisation générative de structure latente.

Le décodage déterministe discriminatif est un décodeur de séquence récurrent basé sur la modélisation d'attention améliorée. La première couche cachée h_1^d est initialisée en utilisant la moyenne de toutes les couches d'entrée source : $h_1^d = \frac{1}{T^e}\sum_{t=1}^{T^e} h_t^e$, où h_t^e est la couche cachée de l'entrée source. T^e est la longueur de la séquence d'entrée.

La couche cachée du décodeur déterministe h_t^d est calculée en utilisant deux couches de GRUs. Sur la première couche, la couche cachée est calculée uniquement en utilisant l'intégration « embedding » du mot d'entrée actuel y_{t-1} et la couche cachée précédente $h_{t-1}^{d_1}$:

$$h_t^{d_1} = GRU_1\left(y_{t-1},\ h_{t-1}^{d_1}\right) \tag{2.7}$$

Où l'exposant d_1 désigne la première couche GRU du décodeur. Ensuite, les poids d'attention à l'instant t sont calculés sur la base de la relation de $h_t^{d_1}$et de toutes les couches cachées source $\{h_t^e\}$. Soit $a_{i,j}$ le poids d'attention entre $h_i^{d_1}$ et h_j^e, qui peut être calculé en utilisant la formulation suivante :

$$a_{i,j} = \frac{\exp(e_{i,j})}{\sum_{j'=1}^{T^e}\exp(e_{i,j'})}$$

$$e_{i,j} = v^T \tanh(W_{hh}^d h_i^{d_1} + W_{hh}^e h_j^e + b_a)$$

Où $W_{hh}^d \in R^{k_h \times k_h}$, $W_{hh}^e \in R^{k_h \times 2k_h}$, $b_a \in R^{k_h}$, et $v \in R^{k_h}$. Le contexte d'attention est obtenu par la combinaison linéaire pondérée de toutes les couches cachées source :

$$c_t = \sum_{j'=1}^{T^e} a_{t,j'} h_{j'}^e$$

La couche cachée déterministe finale $h_t^{d_2}$ est la sortie de la deuxième couche GRU du décodeur, considérant conjointement le mot y_{t-1}, la couche cachée précédente $h_{t-1}^{d_2}$, et le contexte d'attention c_t :

$$h_t^{d_2} = GRU_2\left(y_{t-1},\ h_{t-1}^{d_2},\ c_t\right)$$

Pour le composant du modèle génératif récurrent, inspirée par certaines idées dans des travaux antérieurs (Kingma & Welling, 2013), (Rezende, Mohamed & Wierstra, 2014), (Gregor, Danihelka, Graves, Rezende & Wierstra, 2015), ils ont supposé que les variables latentes, a priori et postérieures, sont Gaussien, c'est-à-dire $p_\theta(z_t) = \mathcal{N}(0, \boldsymbol{I})$ et $q_\phi(z_t \mid y_{<t}, z_{<t}) = \mathcal{N}(z_t; \mu, \sigma^2 \boldsymbol{I})$, où μ et σ désignent respectivement la moyenne variationnelle et l'écart-type, qui peuvent être calculés via un perceptron multicouche. Précisément, étant donné le mot intégré « embedding » y_{t-1}, la variable de la structure latente précédente z_{t-1} , et la couche cachée déterministe précédente h_{t-1}^d , sont projetés d'abord vers un nouvel espace caché :

$$h_t^{e_z} = g\left(W_{yh}^{e_z} y_{t-1} + W_{zh}^{e_z} z_{t-1} + W_{hh}^{e_z} h_{t-1}^d + b_h^{e_z}\right)$$

Où $W_{yh}^{e_z} \in R^{k_h \times k_w}$, $W_{zh}^{e_z} \in R^{k_h \times k_z}$,$W_{hh}^{e_z} \in R^{k_h \times k_h}$, et $b_h^{e_z} \in R^{k_h}$. g est la fonction d'activation sigmoïde : $\sigma(x) = 1/(1+e^{-x})$. Ensuite, les paramètres gaussiens $\mu_t \in R^{k_z}$ et $\sigma_{t \in R^{k_z}}$peuvent être obtenus via une transformation linéaire basée sur $h_t^{e_z}$:

$$\mu_t = W_{h\mu}^{e_z} h_t^{e_z} + b_\mu^{e_z}$$
$$\log\left(\sigma_t^2\right) = W_{h\sigma} h_t^{e_z} + b_\sigma^{e_z}$$

La variable de structure latente $z_t \in R^{k_z}$ peut être calculée en utilisant l'astuce de ré-para-métrisation :

$$\varepsilon \sim \mathcal{N}(0, \boldsymbol{I}), \quad z_t = \mu_t + \sigma_t \otimes \varepsilon$$

Où $\varepsilon \in R^{k_z}$ est une variable de bruit auxiliaire. Le processus d'inférence pour trouver z_t basé sur les réseaux de neurones peut être traité comme un processus de codage variationnel.

Pour générer des résumés précisément, ils ont intégré d'abord le composant de décodage génératif récurrent avec le composant de décodage déterministe discriminatif, et mapper la variable de structure latente z_t et la couche cachée de décodage déterministe $h_t^{d_2}$ vers une nouvelle variable cachée :

$$h_t^{d_y} = \tanh\left(W_{zh}^{d_y} z_t + W_{hh}^{d_z} h_t^{d_2} + b_h^{d_y}\right)$$

Etant donné l'état de décodage combiné $h_t^{d_y}$ à l'instant t, la probabilité de générer un mot cible y_t est donnée comme suit :

$$y_t = \text{ç}\left(W_{hy}^d h_t^{d_y} + b_{hy}^d\right)$$

Où $W_{hy}^{d} \in R^{k_y \times k_h}$ et $b_{hy}^{d} R^{k_y}$. ç(.) Est la fonction softmax. Enfin, ils ont utilisé un algorithme de recherche de faisceau « beam search algorithm » (Koehn,, 2004) pour décoder et générer le meilleur résumé (Li, Lam, Bing & Wang, 2017).

iv. Apprentissage :

Bien que le modèle proposé contienne un décodeur génératif récurrent, l'ensemble du Framework est entièrement différentiable. Comme le montre la section précédente, le décodeur déterministe récurrent et le décodeur génératif récurrent sont conçus à partir de réseaux neuronaux. Par conséquent, tous les paramètres de ce modèle peuvent être optimisés dans un paradigme de bout en bout utilisant la rétropropagation. Ils ont utilisé $\{x\}_N$ et $\{y\}_N$ pour désigner la source d'apprentissage et la séquence cible. Généralement, l'objectif de ce Framework est constitué de deux termes. Un terme est le log-vraisemblance « log-likelihood » négatif des résumés générés, et l'autre est la limite inférieure variationnelle $\Gamma(\theta, \phi, y)$ mentionnée dans l'équation 2.5. Puisque la limite inférieure variationnelle $\Gamma(\theta, \phi, y)$ contient également un terme de vraisemblance, il est possible de le fusionner avec le terme de vraisemblance des résumés. La fonction objective finale, qui doit être minimisée, est formulée comme suit (Li, Lam, Bing & Wang, 2017) :

$$\Gamma = \frac{1}{N} \sum_{n=1}^{N} \sum_{t=1}^{T} \left\{ -\log \left[p\left(y_t^{(n)} \,\middle|\, y_{<t}^{(n)}, x^{(n)} \right) \right] + D_{KL} \left[q_\phi \left(z_t^{(n)} \,\middle|\, y_{<t}^{(n)}, z_{<t}^{(n)} \right) || p_\phi (z_t^{(n)}) \right] \right\}$$

2. L'approche sémantique :

Dans l'approche sémantique, la représentation sémantique des documents est utilisée pour alimenter le système de génération de langage naturel (NLG). Cette approche se concentre sur l'identification des phrases nominales et des phrases verbales en traitant les données linguistiques (voir figure 2.7) (Saggion & Lapalme, 2002).

Différentes méthodes utilisant cette approche sont discutées ici.

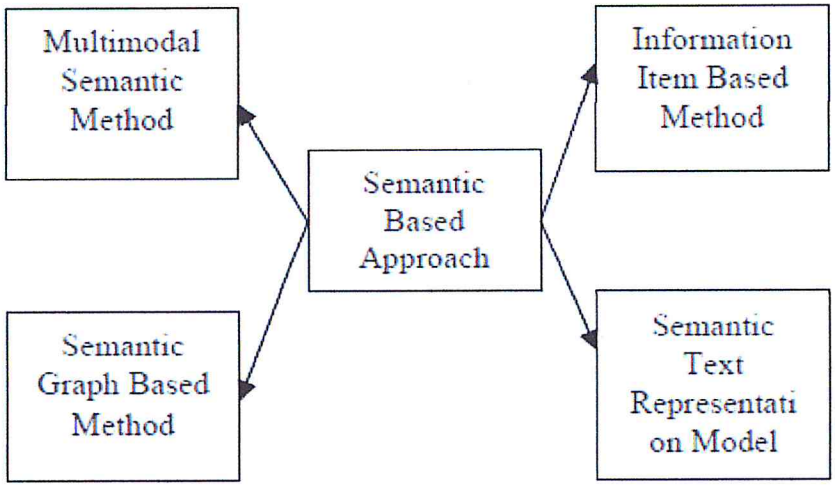

FIGURE 2.7 – Une vue sur les méthodes de l'approche sémantique (Moratanch & Chitrakala, 2016).

a) Multimodal semantic model :

Dans cette technique, un modèle linguistique, qui capture les concepts et la relation entre les idées, est conçu pour représenter le contenu comme le texte et les images utilisés pour les documents multimodaux. Les idées importantes sont évaluées à l'aide de certaines mesures et, éventuellement, les concepts choisis sont exprimés en phrases pour créer un résumé (Moratanch & Chitrakala, 2016).

i. (Gatt & Reiter, 2009) ont proposé un moteur de réalisation qui vise à construire des systèmes NLG de données à texte à grande échelle, dont la tâche est de résumer des volumes massifs de données numériques et symboliques. SimpleNLG fournit des interfaces pour offrir un contrôle direct sur la façon dont les phrases sont construites et combinées, les opérations morphologiques flexionnelles et la linéarisation. Les principales étapes de la construction d'une structure syntaxique et sa linéarisation en tant que texte avec SimpleNLG sont l'initialisation des constituants de base requis pour les items lexicaux, la combinaison des constituants dans des structures plus grandes, le passage au linéarisateur qui traverse la structure inflexions et ordre linéaire en fonction des caractéristiques, et plus tard, la chaîne réalisée est renvoyée.

b) Information item based method :

Dans cette méthodologie, les informations sur le résumé sont générées à partir de la représentation abstraite des documents d'approvisionnement, au lieu des

phrases provenant des documents d'approvisionnement. L'illustration abstraite est un élément d'information qui est la plus petite partie de l'information cohérente dans un texte (Moratanch & Chitrakala, 2016).

i. (Genest & Lapalme, 2011) ont généré un résumé par un item d'information (INIT) qui est le plus petit élément d'information cohérente dans un texte ou une phrase. L'objectif important est d'identifier toutes les entités de texte, leurs attributs, les prédicats entre eux et les caractéristiques des prédicats. Lors de la sélection, l'analyse des documents source qui mène à une liste d'INIT va procéder à la sélection du contenu pour le résumé. Les modèles basés sur la fréquence, tels que ceux utilisés pour le résumé extractif, pourraient être appliqués à la sélection INIT au lieu de la sélection de la phrase. La plupart des INIT ne donnent pas lieu à des phrases complètes, et il est nécessaire de les combiner dans une structure de phrases avant de les réaliser en tant que texte. Les décisions locales sont conçues pour présenter l'information au lecteur et dans quel ordre pendant la génération sont maintenant dirigées par les décisions globales de l'étape de sélection d'INIT. La génération finale de résumé est faite par le classement des phrases générées et un certain nombre de phrases qui sont intentionnellement en excès de la taille limite du résumé est d'abord sélectionné.

c) Semantic graph model :

Cette technique vise à résumer un document en créant un graphe linguistique appel rich semantic graph (RSG) pour le document initial en réduisant le graphe linguistique généré puis en générant le résumé abstractif final à partir du graphe linguistique réduits (Moratanch & Chitrakala, 2016).

i. (Moawad & Aref, 2012) ont résumé un seul document en créant un graphe sémantique appelé Rich Semantic Graph (RSG) à partir du document original. Ensuite, ils réduisent le graphe sémantique généré, et le résumé abstractif final est produit à partir du graphe sémantique réduit. Dans le module de génération de sous-graphes sémantiques riches, pour chaque entrée, le processus d'instanciation des sens instancie un ensemble de concepts de mots pour les sens du verbe et du nom basés sur l'ontologie du domaine. Les processus de validation de concepts sont interconnectés et validés pour générer plusieurs sous-graphes sémantiques riches. Les concepts de phrases sont liés entre eux par les relations syntaxiques et sémantiques générées dans le module de prétraitement. Le

processus de classement des phrases vise à définir le seuil des sous-graphes sémantiques les mieux classés pour chaque phrase. Au cours du module de génération de graphe sémantique riche, un ensemble de règles heuristiques est appliqué au graphe sémantique riche généré pour le réduire en fusionnant, en supprimant ou en consolidant les nœuds de graphe.

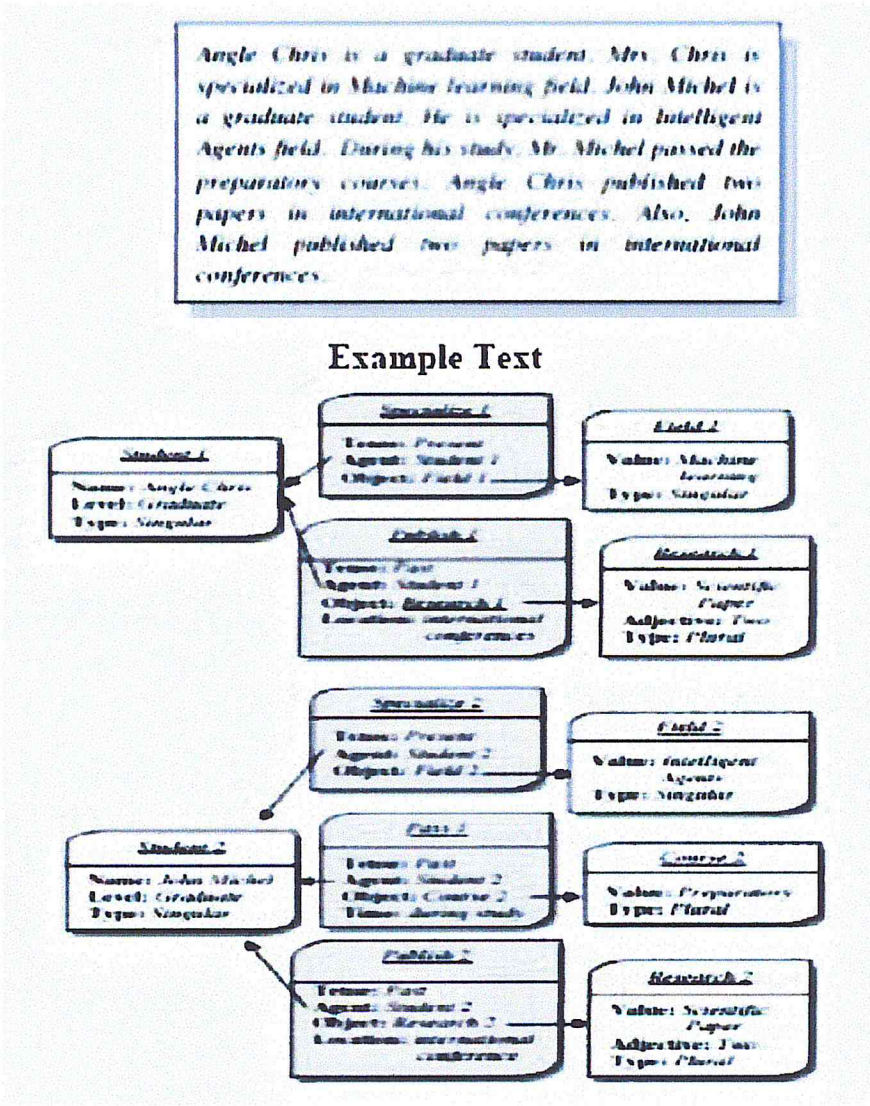

FIGURE 2.8 – Représentation textuelle d'un graphe sémantique (Moawad & Aref, 2012).

Dans figure 2.8, les nœuds dans rich semantic graph représentent les objets des classes d'ontologie de domaine pour les noms et les verbes de texte d'entrée.

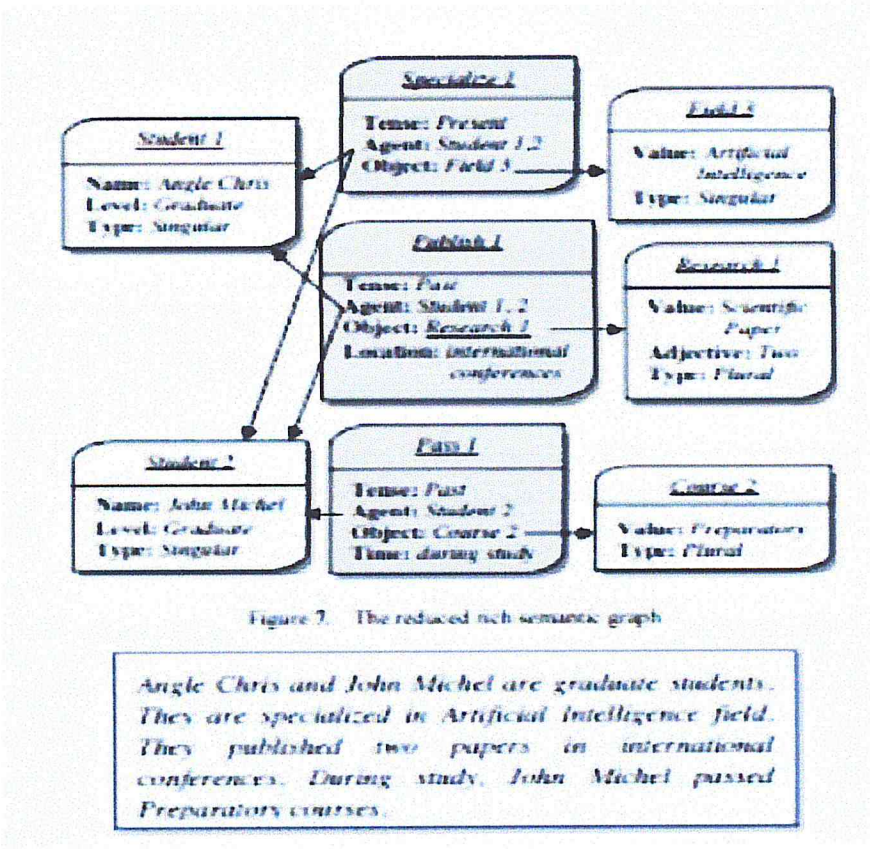

FIGURE 2.9 – Résumé de la réduction de graphe sémantique (Moawad & Aref, 2012).

La figure 2.9 représente le graphe sémantique réduit et le texte résumé obtenu à partir du graphe réduit. Le graphe sémantique réduit est obtenu en appliquant une règle heuristique de réduction. Le résumé abstrait généré ne contient que 50% du document de texte original.

ii. (Moawad & Aref, 2012) ont présenté un article sur l'approche de la réduction de graphe sémantique pour le résumé de textes abstractifs. Cet article propose une approche pour résumer un document unique en créant un graphe sémantique appelé Rich Semantic Graph (RSG). L'approche se compose de trois phases :

A. **La phase de création de graphe sémantique riche :** Cette phase commence par une analyse syntaxique profonde du texte d'entrée. Ensuite, génère des relations de dépendance typées (relations grammaticales), et des étiquettes syntaxiques et morphologiques pour chaque mot. Après cela, pour chaque phrase, le modèle accède à l'ontologie du domaine pour instancier, interconnecter et valider les concepts de la phrase. Les RSG

représentent différentes représentations sémantiques du document entier, où le RSG classé maximum est le RSG considéré.

Module de prétraitement : Il se compose de quatre processus principaux nommés reconnaissanced'entités, analyse morphologique et syntaxique, résolution de références croisées et processus de résolution pronominaux.

Module de Génération de sous-graphes sémantiques riches : Chaque phrase prétraitée est composée d'une suite de mots : Si = [Wi1, Wi2, ... Win], Chaque mot est représenté par une séquence triple Wij = [St, T, D], où St représente le radical du mot, T représente l'ensemble des citations et D représente l'ensemble des relations de dépendance typées.

Module de génération de graphe sémantique riche : Enfin, il génère les graphiques sémantiques riches finaux à partir des sous-graphes sémantiques riches qui ont la valeur la plus élevée de classement des phrases de document. Les sous-graphes sémantiques du document d'entrée seront fusionnés pour former le graphe sémantique riche final.

B. **La phase de réduction de graphe sémantique riche :** Dans cette phase, un ensemble de règles heuristiques est appliqué sur le graphe sémantique riche généré pour le réduire en fusionnant, en supprimant ou en consolidant les nœuds de graphe. La règle heuristique de chaque phrase est composée de trois nœuds : le nœud Sujet (SN), le nœud Verbe principal(MV) et le nœud Objet (ON).

C. **La Phase de Génération de Résumé de texte :** Enfin, la Phase de Génération de résumé de texte vise à générer le résumé abstractif à partir du graphe sémantique riche réduit. Cette phase accepte une représentation sémantique sous la forme de RSG et génère le texte résumé. Le graphe d'entrée contient les informations nécessaires pour générer le texte final. Pour accomplir sa tâche, la phase accède à l'ontologie du domaine, qui contient les informations nécessaires dans le même domaine de RSG (Yeasmin, Tumpa, Nitu & Palash, 2017).

iii. (Fathy, Fadl, & Aref, 2012) ont proposé une approche basée sur la représentation sémantique riche pour la génération de texte. Cet article a proposé un nouveau modèle pour générer un texte en anglais de RSG. Ce modèle accède à l'ontologie WordNet pour générer plusieurs textes selon les synonymes des mots. Il y a cinq phases : Planification du texte, Planification de la phrase,

Réalisation de la surface, Styles d'écriture Sélectionné pour la rédaction et Évaluation du texte.

A. **La phase de planification du texte :** Cette phase comprend la détermination du contenu qui décide quelles informations doivent être incluses dans le texte généré.

B. **La phase de planification de la phrase :** Cette phase reçoit des objets nominaux et verbaux et génère des demi-paragraphes améliorés.

   La planification de la phrase se compose de quatre processus principaux :

   Processus de lexicalisation : Pour sélectionner les synonymes les plus appropriés pour chaque nom/verbe, un poids (W) est assigné pour chaque synonyme.

   W = (E+(1-NR/RT)+(NGS/TG)/3)*10.

   Où E est la probabilité d'existence du synonyme dans le graphe sémantique riche d'entrée, NR représente le synonyme rang deWordNet, RT représente la valeur totale de tous les rangs synonymiques, et NGS représente le groupe de WordNet par similarité pour synonyme, TG représente le nombre total de groupes par similarité pour tous les synonymes.

   Processus de structuration du discours : Pour chaque objet nominal, une pseudo-phrase est composée pour chaque attribut, et une pseudo-phrase est composée pour chaque verbe associé à cet objet.

   Processus d'agrégation : Ce processus décide comment les pseudo-phrases doivent être combinées en demi-paragraphes.

   Processus d'expression référant : Ce processus identifie et remplace le référent prévu par son pronom approprié.

C. **La phase de réalisation de surface :** Cette phase vise à transformer les demi-paragraphes améliorés en paragraphes en les corrigeant grammaticalement et en ajoutant la ponctuation requise.

D. **La phase de génération des styles d'écriture sélectionnés :** Cette phase donne la sortie des paragraphes finaux en fonction du style d'écriture sélectionné par l'utilisateur final.

E. **La phase d'évaluation :** Cette phase évalue et classe les paragraphes en fonction de la cohérence entre les phrases de paragraphe et des synonymes de mots de paragraphe les plus fréquemment utilisés. La mesure de la

cohérence génère des résultats très proches, de sorte que les synonymes de mot de paragraphe les plus fréquemment utilisés sont utilisés comme facteur d'évaluation supplémentaire.

iv. (Bartakke, Sawarkar, & Gulati, 2016) ont proposé une approche basée sur la sémantique pour le résumé abstractif de textes multi-documents. Cet article utilise les multiples documents afin de créer un résumé abstractif. Dans cet article, un graphe sémantique est généré pour chaque phrase. Le graphe généré est réduit et les règles heuristiques ont été utilisées pour générer un résumé abstractif. L'approche se compose de trois phases : phase de création de graphe sémantique riche, phase de génération de sous-graphes sémantiques riches, et Phase de génération de résumé de texte.

A. **Phase de création de graphe sémantique riche :** L'objectif principal de la phase de création de graphe sémantique riche est de représenter sémantiquement les documents d'entrée en utilisant le graphe sémantique riche. Elle se compose de trois modules : le prétraitement, la génération de sous-graphes sémantiques riches, et les modules de génération de graphe sémantique riche.

Le module de prétraitement : Il se compose de quatre processus principaux : **La reconnaissance d'entité nommée** définit des catégories telles que les noms de personnes, les organisations. **L'analyse morphologique** divise chaque mot en morphèmes et **l'analyse syntaxique** analyse la phrase entière et construit l'arbre d'analyse, et les dépendances typées pour exprimer les relations entre les mots. **Les processus de résolution de coréférence et de référence pronominale :** les processus de résolution identifient les entités nommées de coréférence et résolvent les références pronominales dans tous le texte d'entré.

Génération de sous-graphes sémantiques riches : L'objectif principal du module Génération de sous-graphes sémantiques riches est de générer plusieurs sous-graphes sémantiques riches pour chaque phrase prétraitée d'entrée. Ce module comprend trois processus : **Instanciation des sens des mots :** Ce processus instancie les concepts de mots pour les sens du nom et du verbe en fonction de l'ontologie du domaine. **La validation des concepts :** les concepts de phrases instanciés est interconnectée et validée pour générer plusieurs sous-graphes sémantiques riches. **Le processus de classement des phrases sémantiques :** pour générer un graphe

sémantique unique riche, le processus prend en considération le premier sous-graphe sémantique riche classé.

Le module de Génération de graphe sémantique riche : les sous-graphes sémantiques du document d'entrée seront fusionnés pour former le graphe sémantique riche final.

B. **La phase de réduction de graphe sémantique riche** : Dans cette phase, un ensemble de règles heuristiques est appliqué sur le graphe sémantique riche généré pour le réduire en fusionnant, en supprimant ou en consolidant les nœuds de graphe.

C. **La phase de génération de texte :** Cette phase vise à générer le résumé abstractif à partir du Rich Semantic Graph (RSG) réduit. Il existe quatre modules à savoir : **Planification de texte :** Il décide quelles informations doivent être incluses dans le texte généré. **Planification de la phrase :** La planification de la phrase comprend quatre processus principaux : **Processus de lexicalisation :** Les synonymes sont sélectionnés pour chaque objet verb/nom en accédant à l'ontologie de Word. **Processus de structuration du discours :** Contenant les synonymes de l'objet sélectionné sous la forme de pseudo-phrases. **Processus d'agrégation :** Décide comment les pseudo-phrases doivent être combinées en demi-paragraphes. **Processus d'expression référant :** Ce processus identifie le pronom approprié. **Module de réalisation de surface :** Transformer les demi-paragraphes améliorés en paragraphes en corrigeant grammaticalement et en ajoutant la ponctuation requise. **Le module d'évaluation :** Évaluer et classer les paragraphes en fonction de deux facteurs : la cohérence entre les phrases de paragraphe et les synonymes du mot de paragraphe le plus fréquemment utilisé.

d) Semantic text representation model :

Cette technique vise à analyser le texte d'entrée en utilisant la sémantique des mots plutôt que la syntaxe/structure du texte (Moratanch & Chitrakala, 2016).

i. (Khan, Salim, & Kumar, 2015) ont proposé un Framework pour le résumé abstractif de plusieurs documents sous la forme d'une représentation sémantique des documents d'approvisionnement. La sélection du contenu est effectuée en classant les structures d'argument de prédicat les plus significatives. Enfin, le résumé est généré à l'aide d'un outil de génération de langage. Mais le

système ne gère pas la sémantique plus détaillée dans l'approche de résumé en raison de son hypothèse que le texte à traiter sont résolus anaphore et sens désambiguïsés.

ii. (Khan, Salim, & Kumar, 2015) ont suggéré l'étiquetage du rôle sémantique pour extraire la structure des arguments prédicats de chaque phrase et l'ensemble de documents est divisé en phrases avec son numéro de document et son numéro de position. Le numéro de position est attribué à l'aide de l'API d'étiquetage de rôle sémantique SENNA. La matrice de similarité est construite à partir du graphe sémantique pour les scores de similarité sémantique. Après cela, un algorithme de classement basé sur un graphique modifié est utilisé pour déterminer la structure de prédicat, la similarité sémantique et la relation de jeu de documents. Après le prédicat, MMR est utilisé pour réduire la redondance pour la synthèse.

Auteur/Année	**Techniques/Méthodes**	**Représentation du Texte**	**Sélection du Contenu**	**Génération du résumé**
Albert Gatt, 2009	Multimodal Semantic Based	Modèle sémantique	Simple NLG	Technique de génération : Arbre synchrone
Pierre-Etienne Genest, 2011	Information item Based	Représentation abstraite : Information Item (INIT)	Les phrases générées peuvent être classées en fonction de la fréquence du document	NLG réalise un NLG simple
Ibrahim, 2012	Semantic Graph based	Rich semantic graph	Calcul de chaque concepts et leurs phrases	Graphe sémantique réduit
Atif, 2015	Reduced Semantic graph	Étiquetage du rôle sémantique	Graphe sémantique pour le score de similarité	NLG réalise un NLG simple

TABLE 2.3 – Montre une étude comparative sur les méthodes du résumé abstractif pour l'approche sémantique basée sur la représentation du texte, la sélection du contenu et la génération du résumé (Moratanch & Chitrakala, 2016).

Auteur/Année	Techniques/Méthodes	Les avantages	Les inconvénients
Barzilay and McKeown, 1999	Tree based	l'utilisation d'un générateur de langage a significativement amélioré la qualité des résumés résultants, c'est-à-dire la réduction des répétitions et l'augmentation de la fluidité.	le contexte de la phrase n'a pas été inclus lors de la capture de la phrase recoupée.
Harabagiu and Lacatusu, 2002	Template based	que le résumé généré est hautement cohérent car il repose sur des informations pertinentes identifiées par le système IE.	Elle ne fonctionne que si les phrases récapitulatives sont déjà présentes dans les documents source. elle ne peut pas gérer la tâche si la récapitulation multi-documents nécessite des informations sur les similitudes et les différences entre plusieurs documents
Lee and Jian, 2005	Ontology based	qu'elle exploite l'ontologie floue pour gérer des données incertaines que l'ontologie de domaine simple ne peut pas le faire.	l'ontologie du domaine, le dictionnaire chinois et lecorpus de nouvelles doivent être définis par un expert du domaine, ce qui prend beaucoup de temps. Elle est limitée aux nouvelles chinoises et pourrait ne pas être applicable aux nouvelles en anglais
Barzilay and McKeown, 2005	Tree based	Génère un résumé cohérent de qualité.	il manque un modèle complet qui inclurait une représentation abstraite pour la sélection de contenu.
Tanaka and Kinoshita, 2009	Lead and Body phrase	elle a trouvé des révisions sémantiquement appropriées pour réviser une lead phrase.	les erreurs d'analyse dégradent la complétude de la forme, comme la grammaticalité et la répétition. elle se concentre sur les techniques de réécriture, et il manque un modèle complet qui inclurait une représentation abstraite pour la sélection du contenu
Genest and Lapalme, 2012	Rule based	elle a le potentiel de créer des résumés avec une plus grande densité d'information56	que toutes les règles et les schémas sont écrits manuellement, ce qui est fastidieux et prend du temps.

TABLE 2.4 – Tableau comparatif récapitulatif de différentes méthodes de l'approche structural.

2.3 CONCLUSION :

De nombreuses techniques de résumé de texte abstractif ont été développées pour les langues comme l'anglais, l'arabe, l'hindi (Yeasmin, Tumpa, Nitu & Palash, 2017).
Dans ce chapitre nous avons présenté les techniques de résumé automatique, pour le type extractif et pour le type abstractif. Nous avons présenté deux approches structurelles abstractives : « Abstractive Text Summarization using Sequence-to-sequence RNNs and Beyond » et « Deep Recurrent Generative Decoder for Abstractive Text Summarization » qui sont basées sur le Gated Recurrent Unit « GRU ».
Pour notre projet nous nous intéressant à l'approche abstractive, structurelle, nous allons utiliser Variational auto-encoder « VAE » et Generative Adversarial Network « GAN », avec Gated Recurrent Unit "GRU" qui vont être définir dans le chapitre suivant.

CHAPITRE 3

APPRENTISSAGE PROFOND

3.1 INTRODUCTION :

L'apprentissage automatique est un sous-domaine de l'Intelligence Artificielle, il s'intéresse à la recherche de solutions pour résoudre des tâches que nous voulons confier à la machine (Xavier, 2014).
L'apprentissage automatique, est un domaine qui consiste à utiliser des données pour apprendre une solution aux problèmes que nous voulons confier à la machine, le modèle des Réseaux de Neurones Artificiels (RNA) est un outil précieux. Il a été inventé voilà maintenant près de soixante ans, et pourtant, il est encore de nos jours le sujet d'une recherche active. Récemment, avec l'apprentissage profond, il a en effet permis d'améliorer l'état de l'art dans de nombreux champs d'applications comme la vision par ordinateur, le traitement de la parole et le traitement des langues naturelles (Xavier, 2014).
Dans ce chapitre, nous allons définir l'apprentissage automatique, les réseaux de neurones et les réseaux de neurones profonds.

3.2 APPRENTISSAGE AUTOMATIQUE :

L'apprentissage est une phase du développement d'un réseau de neurones durant laquelle le comportement du réseau est modifié jusqu'à l'obtention du comportement désiré (Touzet, 1992).
Dans le cas des réseaux de neurones artificiels, on ajoute souvent à la description du modèle l'algorithme d'apprentissage. Le modèle sans apprentissage présente en effet peu d'intérêt.

Dans la majorité des algorithmes actuels, les variables modifiées pendant l'apprentissage sont les poids des connexions (Touzet, 1992).
Le sur-apprentissage ou sur-ajustement (en anglais «overfitting») est un problème pouvant survenir dans les méthodes mathématiques et informatiques d'apprentissage automatique. Il est en général provoqué par un mauvais dimensionnement de la structure utilisée pour classifier. De par sa trop grande capacité à stocker des informations, une structure dans une situation de sur-apprentissage aura de la peine à généraliser les caractéristiques des données. Elle se comporte alors comme une table contenant tous les échantillons utilisés lors de l'apprentissage et perd ses pouvoirs de prédiction sur de nouveaux échantillons (Xavier, 2014).

3.2.1 L'APPRENTISSAGE SUPERVISÉ :

Il s'agit de la forme d'apprentissage la plus intuitive. Dans ce paradigme les données sont composées d'entrées et de sorties (ou cibles). Nous connaissons donc la solution de la tâche pour un certain ensemble d'observations. On peut représenter les données sous la forme suivante : $d_i = (x_i, y_i)$ avec x l'entrée, y cible associée et i l'indice de l'observation. Lorsque la valeur de la sortie est une valeur discrète :$y \in [[1, n]]$ représentant la classe (ou étiquette) de l'exemple x, on parle de tâche de classification. Il peut s'agir par exemple de la tâche de reconnaissance d'objets : x_i est un vecteur contenant la valeur des pixels d'une image représentant un objet en particulier et y_i est un entier correspondant à cet objet. En pratique, on utilise un vecteur « onehot » pour représenter la sortie. Il s'agit d'un vecteur de dimension n. La dimension correspondant à la classe de l'exemple a la valeur 1 et les autres 0. S'il n'y a que deux classes on parle de classification binaire. Lorsque la valeur de sortie est continue, il s'agit d'une tâche de régression. Par exemple, imaginez que l'on ait un commentaire textuel d'un utilisateur vis-à-vis d'un produit (film, livre, objets...) et que la cible y_i est la note, par exemple sur une échelle entre 0 et 4, au quelle ce commentaire correspondrait. Cette tâche s'appelle l'analyse de sentiment. Vous pouvez remarquer qu'il faut trouver un moyen de représenter le commentaire textuel. Une solution courante est d'utiliser les sacs de mots binaires : on fixe un vocabulaire x de n mots, x_i est un vecteur de n dimensions chacune associées à un mot du vocabulaire, 1 indiquant la présence du mot dans le commentaire et 0 son absence (Xavier, 2014).

3.2.2 L'APPRENTISSAGE NON SUPERVISÉ :

Parfois, les cibles ne sont pas disponibles : on a seulement $d_i = (x_i)$. Dans ce cas, un apprentissage est toujours possible. On peut en effet modéliser la distribution qui génère les entrées

$P(x)$, on parle alors d'estimation de densité de probabilité. On peut aussi regrouper les données par similarité, il s'agit du problème de partitionnement des données. Enfin, il est possible d'apprendre une nouvelle représentation des données, et ce, dans deux contextes différents. Soit dans le but de compresser les données pour les manipuler ou les visualiser plus facilement, il s'agit de la réduction de dimensionnalité (Xavier, 2014).

3.2.3 L'APPRENTISSAGE SEMI-SUPERVISÉ :

A la frontière de ces deux derniers paradigmes, il est aussi possible de n'avoir les cibles que pour une partie de l'ensemble des données. Il est en effet souvent couteux d'obtenir des cibles pour un grand nombre d'observations. L'apprentissage semi-supervisé consiste à utiliser l'information présente dans les données non-étiquetées pour améliorer les performances de l'apprentissage supervisé. On peut par exemple faire un apprentissage non-supervisé sur toutes les données pour extraire des caractéristiques et effectuer l'apprentissage supervisé sur la nouvelle représentation, ou alors, on peut d'abord effectuer un apprentissage supervisé puis prédire les cibles sur les données non-étiquetées et, enfin, incorporer ces nouvelles données à l'entrainement supervisé (Xavier, 2014).

3.2.4 L'APPRENTISSAGE PAR RENFORCEMENT :

Dans le cadre de l'apprentissage par renforcement, le but est d'apprendre quelles actions effectuer, étant donné un contexte, afin de maximiser une mesure de récompense qui peut, elle même, dépendre des actions passées. Des prédictions sont donc effectuées pendant l'apprentissage pour prendre des décisions et explorer l'espace des solutions. Cette approche fait intervenir le compromis exploration-exploitation, c'est à dire, soit essayer de nouvelles stratégies pour trouver une meilleure solution, quitte à faire des erreurs, soit maximiser la récompense avec la solution la plus performante actuellement apprise. C'est ce type de paradigme qui intervient pour l'apprentissage des tâches de contrôle, comme la marche, ou apprendre à jouer à des jeux de société(Xavier, 2014).

3.3 RÉSEAUX DE NEURONES :

Les réseaux de neurones artificiels sont des réseaux fortement connectés de processeurs élémentaires fonctionnant en parallèle. Chaque processeur élémentaire calcule une sortie unique sur la

base des informations qu'il reçoit. Toute structure hiérarchique de réseaux est évidemment un réseau (Touzet, 1992).
Les réseaux de neurones artificiels représentent un outil d'apprentissage automatique.

3.3.1 DÉFINITION D'UN NEURONE :

Chaque neurone artificiel est un processeur élémentaire. Il reçoit un nombre variable d'entrées en provenance de neurones amonts. A chacune de ces entrées est associée un poids w représentatif de la force de la connexion. Chaque processeur élémentaire est doté d'une sortie unique, qui se ramifie ensuite pour alimenter un nombre variable de neurones avals. A chaque connexion est associée un poids (voir figure 3.1) (Touzet, 1992).

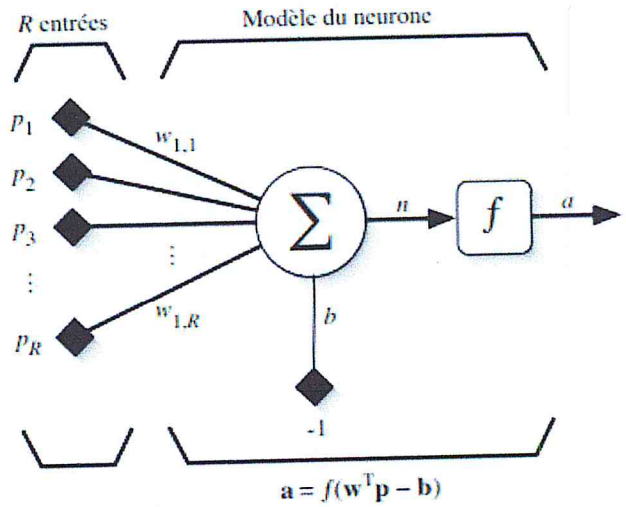

FIGURE 3.1 – Modèle d'un neurone artificiel (Marc, 2006).

1. $P = \{p_1, p_2, \ldots, p_R\}$ représente les entrées de réseau de neurones.
2. b représente le biais.
3. $W = \{w_{1,1}, \ldots, w_{1,R}\}$ représente les poids des arcs.
4. Σ représente la somme pondérée des entrées et poids.
5. f représente la fonction d'activation.

3.3.2 DÉFINITION DES RÉSEAUX DE NEURONES :

Les connexions entre les neurones qui composent le réseau décrivent la topologie du modèle. Elle peut être quelconque, mais le plus souvent il est possible de distinguer une certaine régularité.

Les neurones sont arrangés par couche. Il n'y a pas de connexion entre neurones d'une même couche et les connexions ne se font qu'avec les neurones des couches avales. Habituellement, chaque neurone d'une couche est connecté à tous les neurones de la couche suivante et celle-ci seulement. Ceci nous permet d'introduire la notion de sens de parcours de l'information (de l'activation) au sein d'un réseau et donc définir les concepts de neurone d'entrée, neurone de sortie. Par extension, on appelle couche d'entrée l'ensemble des neurones d'entrée, couche de sortie l'ensemble des neurones de sortie. La couche intermédiaire n'a aucun contact avec l'extérieur est appelée couche cachée (voir figure 3.2) (Touzet, 1992).

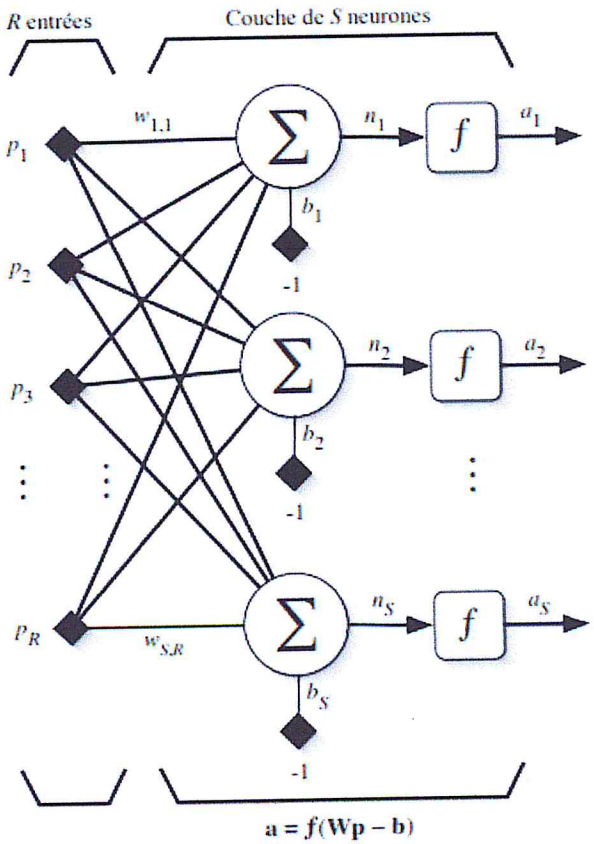

FIGURE 3.2 – Architecture de réseaux de S neurones (Marc, 2006).

1. $P = \{p_1, p_2, \ldots, p_R\}$ représente les entrées de réseau de neurones.
2. $b = \{b_1, b_2, \ldots, b_S\}$ représente les biais.
3. $W = \{w_{1,1}, \ldots, w_{S,R}\}$ représente les poids des arcs.
4. $\sum$ représente la somme pondérée des entrées et poids.
5. f représente la fonction d'activation.

3.3.3 LES LIMITATIONS DU RÉSEAU DE NEURONE CLASSIQUE :

Pour résoudre des problèmes composés de plusieurs transformations non-linéaires successives, il faut utiliser un réseau multicouche (Marc, 2006).
Le RNA classique n'est pas adapté aux données de taille variable, ni aux données séquentielles, car pour les données séquentielles, il faut un réseau de neurones à décalage temporel, avec un stockage d'entrée adéquat.
Les principales difficultés sont les problèmes des minima locaux et celui des gradients diminuant/augmentant ("vanishing/exploding") (Bengio, Simard, & Frasconi, 1994). Avec les méthodes d'apprentissage par gradient, le signal d'erreur doit remonter dans le temps sur les connexions de retour aux entrées passées pour constituer un stockage d'entrée adéquat. La rétropropagation conventionnelle, cependant, souffre d'un temps d'apprentissage trop long, lorsque les décalages de temps minimaux entre les intrants pertinents et les signaux des enseignants correspondants sont étendus. Par exemple, avec «propagation en arrière dans le temps» ou «apprentissage récurrent en temps réel», les signaux d'erreur qui remontent dans le temps tendent à disparaître, qui va empêcher la convergence de réseau de neurone, et donc il va tend vers 0 ou bien l'infini (Hochreiter, 1998).

3.4 RÉSEAUX DE NEURONES PROFONDS :

Un réseau de neurones profond permet aux modèles computationnels composés de plusieurs couches de traitement d'apprendre des représentations de données avec de multiples niveaux d'abstraction (Marc, 2006).
Un réseau de neurones profond est le même qu'un réseau de neurones artificiels, sauf que la couche intermédiaire contient plusieurs couches cachées.
L'apprentissage profond consiste à entrainer des modèles composés de plusieurs transformations non-linéaires successives.

Un RNA avec plusieurs couches cachées en est donc un exemple. Pourtant, nous avons vu qu'un RNA avec une couche cachée est déjà un approximateur universel, quel est alors l'avantage d'augmenter le nombre de couches ?
Il y a plusieurs arguments en faveur de tels réseaux. Premièrement, des résultats mathématiques montrent que certaines classes de fonctions, représentables de manière compacte avec un réseau de profondeur d, nécessiteraient un nombre exponentiel de paramètres avec un réseau de profondeur $d-1$ (Håstad & Goldmann, 1991), (Delalleau & Bengio, 2011). Ensuite, on peut noter que l'architecture des réseaux profonds permet la réutilisation de paramètres ou de caractéristiques extraites, propriété désirable pour la modélisation de fonctions complexes (Xavier, 2014).

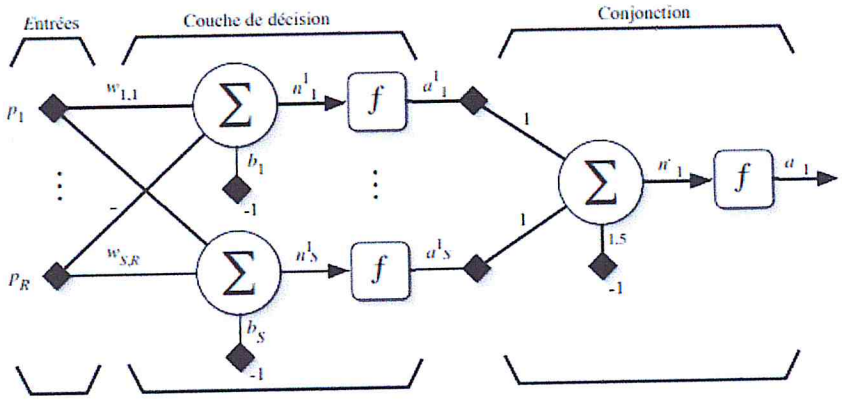

FIGURE 3.3 – Exemple de réseau multicouche (Marc, 2006).

Un réseau multicouche n'est rien d'autre qu'un assemblage de couches concaténées les unes aux autres, de la gauche vers la droite, en prenant les sorties d'une couche et en les injectant comme les entrées de la couche suivante comme le montre la figure 3.3 (Marc, 2006).
Un réseau multicouche doit toujours posséder des neurones avec fonctions de transfert non-linéaires sur ses couches cachées. Sur sa couche de sortie, selon le type d'application, il pourra comporter des neurones linéaires ou non-linéaires (Marc, 2006).

1. La descente de gradient : La descente de gradient est la technique d'apprentissage automatique la plus importante (Bouzy, 2005). Elle est à la base de la formation des systèmes intelligents. Elle repose sur une idée très simple, au lieu de deviner immédiatement la meilleure solution à un objectif donné :
   a) Nous imaginons une solution initiale.
   b) Après trouver une direction proche d'une meilleure solution.

L'algorithme répète ce processus jusqu'à ce qu'il arrive à une solution, c'est assez bon, puisque nous ne pouvons pas connaître la meilleure solution dès le départ, cette méthode d'estimation et de vérification est extrêmement utile. Elle est utilisée dans de nombreux types différents de modèles d'apprentissage automatique, et elle est utilisée pour optimiser ou améliorer la précision des prédictions de modèles utilisés (Bouzy, 2005).

Adam (Adaptive Moment Estimation) (Kingma & Ba,2014) permet de choisir une descente de gradient basée sur la stratégie d'optimisation, puisque nous sommes en train de calculer les taux d'apprentissage pour chaque paramètre, et mettre à jour les paramètres.

2. La propagation vers l'avant (forward propagation) : Dans la propagation vers l'avant (Hirasawa, Ohbayashi, Koga & Harada,1996), les données circuleront dans une direction de la couche d'entrée à la sortie. En ayant des poids qui relient chaque neurone dans une couche à chaque neurone dans la couche suivante. Il faut simplement répéter le même processus pour calculer la prédiction de sortie.A chaque itération, nous calculons une valeur d'erreur, nous souhaitons optimiser cette erreur. Le but est de trouver la direction vers l'erreur minimale, il faut prendre des mini-étapes en descendant le gradient.
3. La rétropropagation (back propagation) : La rétropropagation est une méthode utilisée dans les réseaux de neurones artificiels pour calculer un gradient nécessaire au calcul des poids à utiliser dans le réseau (Marc, 2006). Elle est couramment utilisée pour former des réseaux neuronaux profonds. Dans le contexte de l'apprentissage, la rétropropagation est couramment utilisée par l'algorithme d'optimisation de la descente de gradient pour ajuster le poids des neurones en calculant le gradient de la fonction de perte. Cette technique est parfois appelée propagation vers l'arrière des erreurs, car l'erreur est calculée en sortie et redistribuée à travers les couches réseau. Le but de la rétro-propagation est de comprendre les dérivées partielles de notre fonction d'erreur par rapport à chaque poids individuel dans le réseau, donc on peut utiliser celles en descente de gradient, cela nous donne un moyen de calculer l'erreur pour chaque couche et ensuite, reliant ces erreurs à la quantité d'intérêt réel. Après avoir eu l'erreur pour la couche de sortie, nous calculons une erreur pour chaque neurone dans la couche cachée en remontant couche par couche. Les étapes de l'algorithme de la rétropropagation sont présentés comme suit (Marc, 2006) :

   a) Initialiser tous les poids du réseau à de petites valeurs aléatoires.

   b) Pour chaque association, dans la base d'apprentissage :

i. Propager les entrées vers l'avant à travers les couches du réseau.

ii. Retro-propager les sensibilités vers l'arrière à travers les couches du réseau.

iii. Mettre à jour les poids et biais.

c) Si le critère d'arrêt est atteint, alors **stop**.

d) Sinon, permuter l'ordre de présentation des associations de la base d'apprentissage.

e) Recommencer à l'étape 2.

3.4.1 RÉSEAUX DE NEURONES CONVOLUTIONNELS :

Les réseaux convolutionnels (LeCun & Bengio, 1994) ont été les premières architectures profondes entrainées efficacement.Ils sont inspirés d'observations du traitement de l'information dans le cortex visuel où les neurones sont organisés en colonnes corticales. Le réseau n'admet que des connections locales, et de plus, les paramètres de ces connections sont partagés au sein d'une couche donnée. Cela réduit drastiquement le nombre de paramètres du modèle. Mathématiquement, la multiplication matricielle est remplacée par une convolution (voir figure 3.4 et figure 3.5) : les caractéristiques sont extraites sur tout le champ visuel plutôt qu'être spécifique à une région de ce dernier. En addition, une couche d'agrégation, ou "pooling", est souvent utilisée pour réduire la taille des couches cachées (dans le cas convolutionnel appelées cartes de caractéristiques), cela permet aussi d'obtenir une invariance à des petites translations. La réduction du nombre de paramètres et le choix de l'architecture, appropriée au type de données à traiter, permet de faciliter l'apprentissage ; même dans le cas d'architectures profondes. Ce type de réseaux est très performant pour les tâches de traitement d'images (Szegedy, et al., 2014), et est également utilisé pour le traitement de vidéo (Taylor, Fergus, LeCun & Bregler, 2010), d'audio (Lee, Grosse, Ranganath & Ng, 2009) et de texte (Collobert, Weston, Bottou, Karlen, Kavukcuoglu, & Kuksa, 2011) (Xavier, 2014).

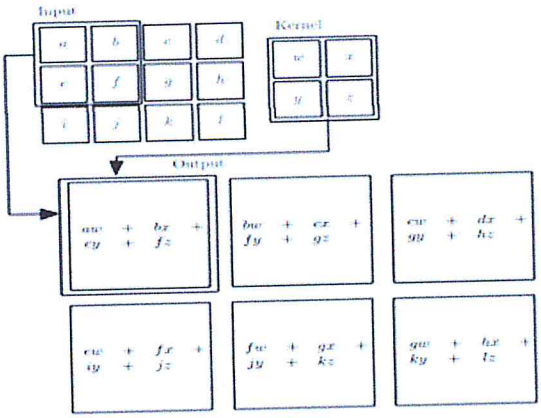

FIGURE 3.4 – Un exemple de convolution 2D sans retournement de noyau (kernel). Dans ce cas, nous limitons la sortie aux seules positions où le noyau se trouve entièrement dans l'image, appelée convolution "valide" dans certains contextes. Nous dessinons des boîtes avec des flèches pour indiquer comment l'élément supérieur gauche du tenseur de sortie est formé en appliquant le noyau à la région supérieure gauche correspondante du tenseur d'entrée (Goodfellow, Bengio, Courville & Bengio, 2016).

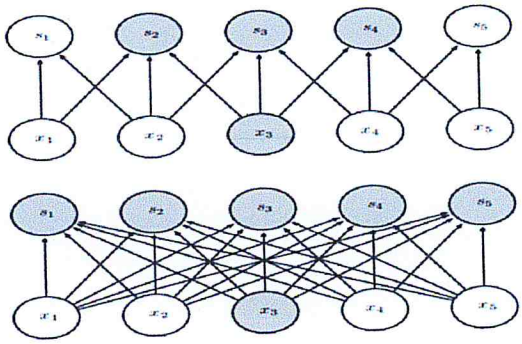

FIGURE 3.5 – Connectivité clairsemée, vue de dessous : Nous mettons en évidence une unité d'entrée, x3, et mettons également en surbrillance les unités de sortie dans s qui sont affectées par cette unité. (Haut) Lorsque s est formé par convolution avec un noyau de largeur 3, seules trois sorties sont affectées par x. (Bas) Lorsque s est formé par la multiplication matricielle, la connectivité n'est plus éparse, donc toutes les sorties sont affectées par x3 (Goodfellow, Bengio, Courville & Bengio, 2016).

3.4.2 RÉSEAUX NEURONAUX RÉCURRENTS :

Les réseaux neuronaux récurrents ou RNN (Rumelhart, Hinton, & Williams, 1986) sont une famille de réseaux de neurones pour le traitement de données séquentielles. Tout comme un réseau convolutif est un réseau neuronal spécialisé pour traiter une grille de valeurs X telle qu'une image, un réseau neuronal récurrent est un réseau neuronal spécialisé pour traiter une suite de valeurs $x(1),...,x(t)$. Tout comme les réseaux convolutifs peuvent facilement évoluer vers des images de grande largeur et hauteur, et certains réseaux convolutionnels peuvent traiter des images de taille variable, les réseaux récurrents peuvent évoluer vers des séquences beaucoup plus longues que pour les réseaux sans spécialisation séquentielle. La plupart des réseaux récurrents peuvent également traiter des séquences de longueur variable. Un réseau neuronal récurrent partage les mêmes poids sur plusieurs pas de temps (Goodfellow, Bengio, Courville & Bengio, 2016).

Les réseaux récurrents partagent des paramètres d'une manière différente. Chaque membre de la sortie est une fonction des membres précédents de la sortie. Chaque membre de la sortie est produit en utilisant la même règle de mise à jour appliquée aux sorties précédentes. Cette formulation récurrente entraîne le partage de paramètres à travers un graphe de calcul très profond (voir figure 3.6) (Goodfellow, Bengio, Courville & Bengio, 2016).

Pour la simplicité de l'exposition, nous nous référons aux RNN comme fonctionnant sur une séquence qui contient des vecteurs $x(t)$ avec l'indice de pas de temps t allant de 1 à t. En pratique, les réseaux récurrents opèrent généralement sur des mini-batchs de telles séquences, avec une longueur de séquence différente pour chaque membre de la mini-batch. Les RNN peuvent également être appliqués en deux dimensions à travers des données spatiales telles que des images, et même lorsqu'elles sont appliquées à des données temporelles, le réseau peut avoir des connexions qui remontent dans le temps, à condition que toute la séquence soit observée avant d'être fournie au réseau (Goodfellow, Bengio, Courville & Bengio, 2016).

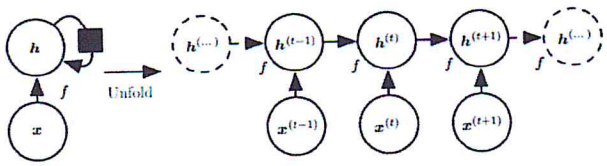

FIGURE 3.6 – Représente Un réseau récurrent sans sorties. Ce réseau récurrent ne traite que les informations de l'entrée x en les incorporant dans l'état h qui est transmis dans le temps. (Gauche) Schéma de circuit. Le carré noir indique un retard d'un pas de temps unique. (Droite) Le même réseau vu comme un graphe de calcul déplié, où chaque nœud est maintenant associé à une instance de temps particulière (Goodfellow, Bengio, Courville & Bengio, 2016).

Unfold : est l'opération qui mappe un circuit comme dans le côté gauche de la figure à un graphique de calcul avec des morceaux répétés comme dans le côté droit.
Le processus « Unfold » présente deux avantages majeurs :

1. Quelle que soit la longueur de la séquence, le modèle d'apprentissage a toujours la même taille d'entrée, car il est spécifié en termes de transition d'un état à un autre, plutôt que spécifié en termes d'historique d'états de longueur variable (Goodfellow, Bengio, Courville & Bengio, 2016).
2. Il est possible d'utiliser la même fonction de transition f avec les mêmes paramètres à chaque pas de temps (Goodfellow, Bengio, Courville & Bengio, 2016).

Il existe différents types de cellules des RNN, qui vont être présentés comme suit :
Les modèles de séquence les plus efficaces utilisés dans les applications pratiques sont appelés gated RNNs. Ceux-ci comprennent Long Short Term Memory et Gated recurrent unit. Ils sons basés sur l'idée de créer des chemins à travers le temps qui ont des dérivées qui ne disparaissent "vanish" ni n'explosent "explode". Gated recurrent unit généralise ceci aux poids de connexion qui peuvent changer à chaque pas de temps (Goodfellow, Bengio, Courville & Bengio, 2016).

1. Cellule Long Short Term Memory « LSTM » :

   Les réseaux récurrents LSTM ont des "cellules LSTM" qui ont une récurrence interne (une auto-boucle), en plus de la récurrence externe du RNN. Chaque cellule a les mêmes entrées et sorties qu'un réseau récurrent ordinaire, mais possède plus de paramètres et un système d'unités de contrôle qui contrôle le flux d'informations (voir figure 3.7) (Goodfellow, Bengio, Courville & Bengio, 2016).

   L'idée ingénieuse d'introduire des auto-boucles pour produire des chemins où le gradient peut circuler pendant de longues durées est une contribution essentielle du modèle initial de mémoire à long terme (LSTM) (Hochreiter & Schmidhuber, 1997). Un ajout crucial a été de faire en sorte que le poids de cette auto-boucle soit conditionné au contexte plutôt que fixe (Gers, Eck & Schmidhuber, 2000). En faisant en sorte que le poids de cette auto-boucle soit contrôlé (par une autre unité cachée), l'échelle de temps de l'intégration peut être changée dynamiquement. Dans ce cas, nous voulons dire que même pour un LSTM avec des paramètres fixes, l'échelle de temps de l'intégration peut changer en fonction de la séquence d'entrée, parce que les constantes de temps sont produites par le modèle lui-même. Le LSTM a été trouvé extrêmement efficace dans de nombreuses applications, comme la reconnaissance d'écriture manuscrite sans contraintes (Graves & Schmidhuber, 2009), la reconnaissance vocale (Graves, 2013), (Graves & Jaitly, 2014), la génération d'écriture manuscrite (Graves, 2013), la traduction automatique (Sutskever, Vinyals, &

Le, 2014), le sous-titrage d'images (Kiros, Salakhutdinov, & Zemel, 2014), (Vinyals, Toshev, Bengio, & Erhan, 2015), (Xu, et al., 2015) et l'analyse syntaxique (Vinyals, Kaiser, Koo, Petrov, Sutskever, & Hinton, 2015), (Goodfellow, Bengio, Courville & Bengio, 2016).

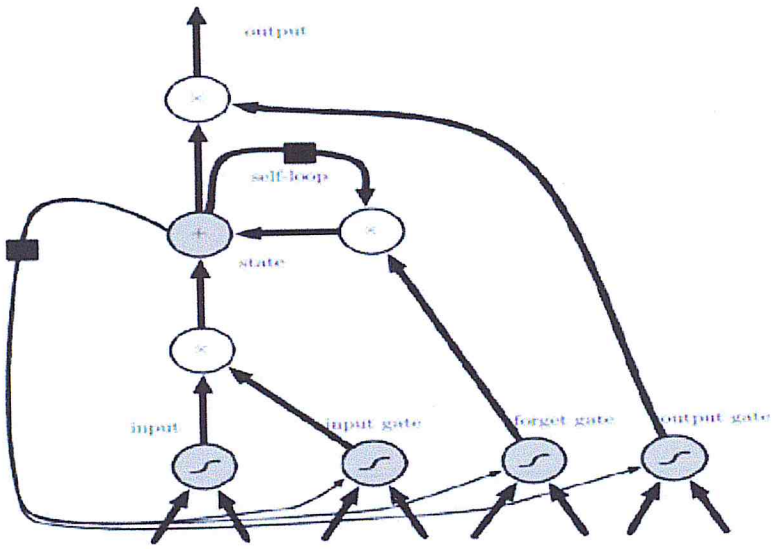

FIGURE 3.7 – Schéma fonctionnel de la "cellule" récurrente du réseau LSTM. Les cellules sont connectées de manière récurrente les unes aux autres, remplaçant les unités cachées habituelles des réseaux récurrents ordinaires. Une entité en entrée est calculée avec une unité de neurones artificielle régulière. Sa valeur peut être cumulée dans l'état si la porte d'entrée sigmoïdale le permet. L'unité d'état a une auto-boucle linéaire dont le poids est contrôlé par la porte d'oubli. La sortie de la cellule peut être coupée par la porte de sortie. Toutes les unités des portes ont une non-linéarité sigmoïde, tandis que l'unité d'entrée peut avoir une non-linéarité d'écrasement. L'unité d'état peut également être utilisée comme entrée supplémentaire pour les unités des portes. Le carré noir indique un retard d'un pas de temps unique (Goodfellow, Bengio, Courville & Bengio, 2016).

Quelles parties de l'architecture LSTM sont réellement nécessaires ? Quelles autres architectures réussies ont pu être conçues pour permettre au réseau de contrôler dynamiquement l'échelle de temps et le comportement d'oubli des différentes unités ?

Pour répondre ces questions, on passe à la cellule Gated recurrent unit « GRU ».

2. Cellule Gated recurrent unit « GRU » :

   Certaines réponses au questions précédentes sont données avec les travaux récents sur les gated RNN, dont les unités sont également connues sous le nom gated recurrent unit ou GRU (Cho, Van Merriënboer, Bahdanau, & Bengio, 2014), (Cho, Van Merriënboer, Gulcehre, Bahdanau, Bougares, Schwenk & Bengio, 2014), (Chung, Gülçehre, Cho, &

Bengio, 2015), (Jozefowicz, Zaremba, & Sutskever, 2015), (Chrupała, Kádár & Alishahi, 2015). La principale différence avec le LSTM est qu'une seule unité de contrôle qui permet de contrôler simultanément le facteur d'oubli et la décision de mettre à jour l'unité d'état (Goodfellow, Bengio, Courville & Bengio, 2016).

Les portes de réinitialisation et de mise à jour peuvent "ignorer" individuellement des parties du vecteur d'état. Les portes de mise à jour agissent comme des intégrateurs conditionnels qui peuvent linéairement ignorer n'importe quelle dimension, en choisissant de les copier (à l'extrémité du sigmoïde) ou de les ignorer complètement (à l'autre extrême) en les remplaçant par la nouvelle valeur vers lequel l'intégrateur qui fuit veut converger. Les portes de réinitialisation contrôlent quelles parties de l'état sont utilisées pour calculer l'état cible suivant, en introduisant un effet non linéaire supplémentaire dans la relation entre l'état passé et l'état futur. Beaucoup plus de variantes autour de ce thème peuvent être conçues. Par exemple, la sortie de réinitialisation (ou porte d'oubli) peut être partagée entre plusieurs unités cachées. Alternativement, le produit d'une porte globale (couvrant un groupe entier d'unités, comme une couche entière) et une porte locale (par unité) pourraient être utilisés pour combiner le contrôle global et le contrôle local. Cependant, plusieurs études sur les variations architecturales du LSTM et du GRU n'ont trouvé aucune variante qui les battrait clairement dans un large éventail de tâches (Greff, Srivastava, Koutník, Steunebrink, & Schmidhuber, 2015), (Jozefowicz, Zaremba, & Sutskever, 2015). (Greff, Srivastava, Koutník, Steunebrink, & Schmidhuber, 2015) ont trouvé qu'un ingrédient crucial est la porte d'oubli, alors que (Jozefowicz, Zaremba, & Sutskever, 2015) ont trouvé que l'ajout d'un biais de 1 à la porte LSTM oubli, une pratique préconisée par (Gers, Eck & Schmidhuber, 2000), rend le LSTM aussi fort que la meilleure des variantes architecturales explorées (Goodfellow, Bengio, Courville & Bengio, 2016).

3.4.3 GENERATIVE ADVERSARIAL NETWORK « GAN » :

Dans ce réseau, le modèle génératif est opposé à un adversaire : un modèle discriminant qui apprend à déterminer si un échantillon provient de la distribution du modèle ou de la distribution des données. Le modèle génératif peut être considéré comme analogue à une équipe de contrefacteurs, essayant de produire de fausses devises et de l'utiliser sans détection, tandis que le modèle discriminatif est analogue à la police, essayant de détecter la fausse monnaie. La concurrence dans ce jeu pousse les deux équipes à améliorer leurs méthodes jusqu'à ce que les contrefaçons soient indissociables des articles authentiques (voir figure 3.8) (Goodfellow, et al.,

2014).

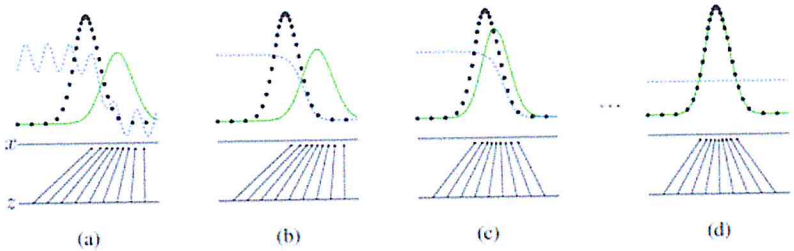

FIGURE 3.8 – Les réseaux accusatoires génératifs sont formés en mettant à jour simultanément la distribution discriminative (D, bleu, ligne pointillée) de manière à discriminer les échantillons de la distribution génératrice de données (ligne noire pointillée) de ceux de la distribution générative (vert, ligne continue). La ligne horizontale inférieure est le domaine à partir duquel z est échantillonné, dans ce cas de manière uniforme (Goodfellow, et al., 2014).

3.4.4 AUTOENCODERS :

Un auto-encoder est un réseau neuronal qui est formé pour tenter de copier son entrée à sa sortie. En interne, il possède une couche cachée h qui décrit un code utilisé pour représenter l'entrée. Le réseau peut être considéré comme constitué de deux parties : une fonction de codeur $h = f(x)$ et un décodeur qui produit une reconstruction $= g(h)$ (voir figure 3.9). Si un auto-encoder réussit simplement à apprendre à mettre $g(f(x)) = x$ partout, alors ce n'est pas particulièrement utile. Au lieu de cela, les auto-encodeurs sont conçus pour être incapables d'apprendre à copier parfaitement. Habituellement, ils sont restreints de manière à ne copier que de manière approximative et à ne copier que les données ressemblant aux données d'entraînement. Parce que le modèle est forcé de prioriser quels aspects de l'entrée doivent être copiés, il apprend souvent des propriétés utiles des données (Goodfellow, Bengio, Courville & Bengio, 2016). Traditionnellement, les auto-encodeurs étaient utilisés pour la réduction de la dimensionnalité ou l'apprentissage de fonctionnalités. Récemment, les connexions théoriques entre les auto-encodeurs et les modèles à variables latentes ont amené les auto-encodeurs à l'avant-garde de la modélisation générative. Les auto-encodeurs peuvent être considérés comme un cas particulier de réseaux feedforward, et peuvent être entraînés avec toutes les mêmes techniques, typiquement la descente de gradient mini-batch suivant des gradients calculés par rétropropagation (Goodfellow, Bengio, Courville & Bengio, 2016).

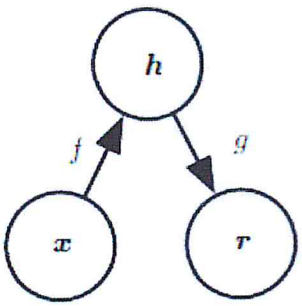

FIGURE 3.9 – La structure générale d'un auto-encoder, mappant une entrée x à une sortie (appelée reconstruction) r à travers une représentation interne ou un code h. L'auto-encoder a deux composantes : le codeur f (mappage x à h) et le décodeur g (mappage h à r) (Goodfellow, Bengio, Courville & Bengio, 2016).

3.5 CONCLUSION :

L'apprentissage automatique, est un domaine qui consiste à utiliser des données pour apprendre une solution aux problèmes que nous voulons confier à la machine, le modèle des Réseaux de Neurones Artificiels (RNA) est un outil précieux. Il a été inventé voilà maintenant près de soixante ans, et pourtant, il est encore de nos jours le sujet d'une recherche active. Récemment, avec l'apprentissage profond, il a en effet permis d'améliorer l'état de l'art dans de nombreux champs d'applications comme la vision par ordinateur, le traitement de la parole et le traitement des langues naturelles (Xavier, 2014).
Dans ce chapitre, nous avons défini l'apprentissage automatique, les réseaux de neurones, les réseaux de neurones profonds , avec les différentes architectures.

CHAPITRE 4

CONCEPTION

4.1 INTRODUCTION :

Le résumé multidocuments de textes est l'une des thématiques de recherche récentes à laquelle les chercheurs en Traitement Automatique des Langues se sont intéressés (Mnasri, 2015). Dans ce chapitre nous allons définir la problématique initiale de génération des résumés automatiques multidocuments, décrire les étapes de génération des résumés pour notre système (les prétraitements, les traitements et les post-traitements), et nous terminons ce chapitre avec une conclusion récapitulative de ce que nous avons fait.

4.2 PROBLÉMATIQUE :

Résumer un texte consiste à réduire ce texte en un nombre limité de mots. Le texte ainsi réduit doit rester fidèle aux informations et idées du texte original. Que ce soit pour des professionnels qui doivent prendre connaissance du contenu de documents en un temps limité ou pour un particulier désireux de se renseigner sur un sujet donné sans disposer du temps nécessaire pour lire l'intégralité des textes qui en traitent, le résumé est une aide contextuelle importante. Avec l'augmentation de la masse documentaire disponible électroniquement, résumer des textes automatiquement est devenu un axe de recherche important dans le domaine du traitement automatique de la langue. La production automatique de résumés pose le problème de la détection et de la modélisation des informations contenues dans les textes. Elle suppose également la hiérarchisation de ces informations afin d'intégrer au résumé les plus importantes (Bossard, 2010). Dans notre projet nous visons à améliorer le résumé automatique multidocuments de

textes de type abstractif, en se basant sur l'apprentissage automatique, avec un modèle hybride qui est composé de deux modèles de génération de textes, et qui sont déjà définis dans le chapitre 3, le « GAN » et l'« Auto-Encoder» nommé aussi «Variational Auto-Encoder (VAE)» (voir figure 4.1).

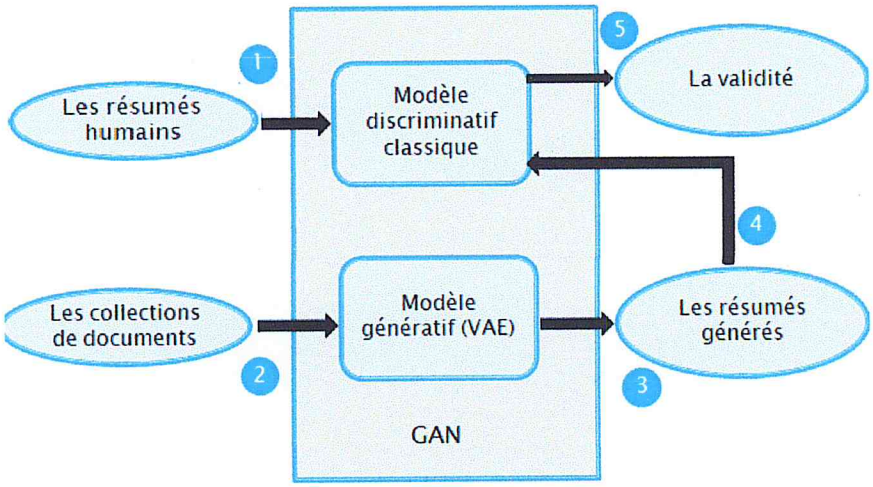

FIGURE 4.1 – Schéma global de l'architecture de notre système.

4.3 DESCRIPTION DES ÉTAPES DE NOTRE SYSTÈME :

Dans cette partie, nous allons définir les étapes de notre système en détail, en commençant par les prétraitements, les traitements (qui représente le modèle hybride) et à la fin les post-traitements.

4.3.1 LES PRÉTRAITEMENTS :

Les documents (articles) sont regroupés en collections par thématique. Chaque thématique lui est associée un résumé. Pour chacune de ces collections :

1. Nous récupérons les articles un par un, et nous fusionnons leurs contenus.
2. Nous faisons l'élimination des mots vides pour le « contenu global » (pour le cas des articles) et le « le contenu unique » (pour le cas des résumés), en utilisant une liste qui contient les mots vides pour la langue anglaise.
3. Nous faisons la segmentation de texte en phrases, puis en mots.

4. Nous générons un dictionnaire qui associe à chaque mot un indice, qui va nous aider à générer les résumés par la suite.
5. Nous calculons la mesure TF (Term Frequency) pour chaque mot, en utilisant la formule de calcul qui est déjà définie dans le chapitre 1. Cette mesure va être utilisée comme une caractéristique pour l'entrainement de notre modèle.
6. Nous générons des fichiers prétraités de format P. pour les documents d'entrainement, ses résumés, et les documents de test. Voici l'utilité du choix des fichiers de format P : Un fichier P est un module Python utilisé pour convertir des objets Python en une représentation en octets pour le stockage sur disque ou le transfert sur un réseau. Il permet aux objets d'être stockés ou transmis facilement sans devoir d'abord convertir les données dans un autre format. Les fichiers P peuvent être "décomposés" et rechargés dans la mémoire du programme pendant l'exécution.

La figure 4.2 représente un schéma récapitulatif de prétraitements :

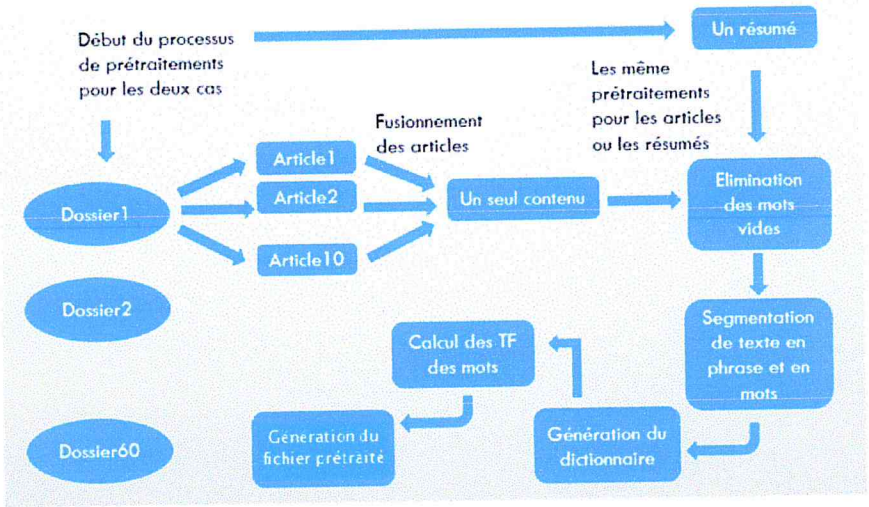

FIGURE 4.2 – Schéma récapitulatif de prétraitements.

4.3.2 LES TRAITEMENTS :

Dans cette partie nous allons définir les traitements de notre système, qui permettent la génération des résumés, pour cela nous allons commencer par l'algorithmes d'apprentissage utilisé, les fonctions du calcul d'erreur utilisées, les fonctions d'activation utilisées, et ensuite l'architecture neuronale de notre modèle.

4.3.2.1 ALGORITHME D'APPRENTISSAGE UTILISÉ :

Pour l'entrainement de notre système nous avons utilisé Adam (Adaptive Moment Estimation) qui est défini au chapitre 3.

4.3.2.2 LE CALCUL D'ERREUR :

Dans la plupart des réseaux d'apprentissage, l'erreur est calculée comme la différence entre la sortie réelle et la sortie prévue. Différentes fonctions de perte donneront différentes erreurs pour la même prédiction, et auront donc un effet considérable sur la performance du modèle. Voici les fonctions d'erreur utilisées dans notre système :

1. Binary Cross Entropy Error : Cette mesure est couramment utilisée pour quantifier la différence entre deux distributions de probabilité p (la vraie distribution) et q (la distribution prédite), elle permet de faire la classification qu'en deux classes[1].
   Sa formule est la suivante : $H(p,q) = -\sum_i p_i log_2(q_i)$
2. Kullback-Leibler Divergence Error (KL divergence) : Cette mesure s'interprète comme la différence moyenne du nombre de bits nécessaires au codage d'échantillons de p. p représente les données, les observations, ou une distribution de probabilités calculée avec précision. La distribution q représente typiquement une théorie, un modèle, une description ou une approximation de P [1].
   Sa formule est la suivante : $D_{KL} = \sum_i p(i) log \frac{p(i)}{q(i)}$
3. Bayes Error : Etant donné deux évènements A et B, Bayes Error permet de déterminer la probabilité de A sachant B, si l'on connait les probabilités de A, de B et de B sachant A[2].

   Sa formule est la suivante : $P(A \mid B) = \frac{P(B|A)\ P(A)}{P(B)}$
   Nous avons implémenté notre propre fonction bayésienne de perte pour le modèle de

1. https ://isaacchanghau.github.io/post/lossfunctions/
2. https ://fr.wikipedia.org/wiki/Th/C3/A9or/C3/A8medeBayesInf/C3/A9rencebay/C3/A9sienne

VAE (Variational Auto-Encoder) qui est défini ci-dessous (Après le GAN), l'idée de VAE est de déduire la distribution de probabilité de la variable latente, car nous ne le savons pas encore. Dans le VAE, nous l'inférons en utilisant une méthode appelée Inférence Variationnelle (VI). VI est l'un des choix de méthode populaire dans l'inférence bayésienne. L'idée principale de VI est de poser l'inférence en l'abordant comme un problème d'optimisation. En modélisant la distribution vraie en utilisant une distribution plus simple qui est facile à évaluer (Gaussien), et minimiser la différence entre ces deux distributions en utilisant la métrique de KL divergence, ce qui nous indique la différence.

4. Mean Squared Error (MSE) : Elle permet de trouver la moyenne d'un ensemble d'erreurs entre deux distributions de probabilité p (la vraie distribution) et q (la distribution prédite), elle est utilisée pour mesurer la performance d'un estimateur, la quadrature est nécessaire pour supprimer tous les signes négatifs [1] .
   Sa formule est la suivante : $MSE = \frac{1}{n}\sum_{i=1}^{n}(q_i - p_i)^2$

4.3.2.3 LES FONCTIONS D'ACTIVATION :

Les fonctions d'activation permettent de mapper une sortie particulière à un ensemble particulier d'entrées. Dans le cadre du choix de la fonction d'activation que nous devons utiliser dans notre modèle, nous devrons essayer différentes fonctions et choisir celle qui correspond le mieux à notre modèle. Voici l'ensemble des fonctions d'activation utilisées :

1. LeakyReLU : Le ReLU est la fonction d'activation la plus utilisée au monde en ce moment, elle est utilisée dans presque tous les réseaux de neurones. Mais le problème est que toutes les valeurs négatives deviennent immédiatement nulles, ce qui diminue la capacité du modèle à s'adapter ou à s'entraîner correctement à partir des données. Cela signifie que toute entrée négative donnée à la fonction d'activation ReLU fait immédiatement passer la valeur à zéro, ce qui ne permet pas de mapper correctement les valeurs négatives [2].

1. https ://isaacchanghau.github.io/post/lossfunctions/
2. https ://towardsdatascience.com/activation-functions-neural-networks-1cbd9f8d91d6

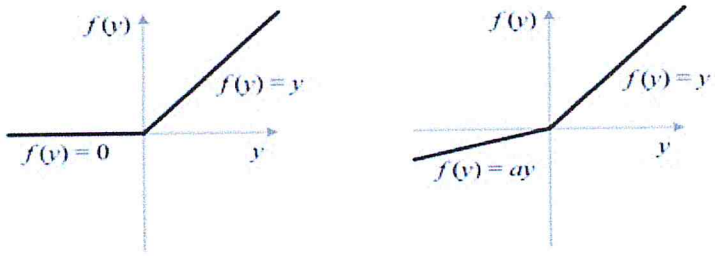

FIGURE 4.3 – Le graphe de la fonction ReLU à gauche et LeakyReLU à droite.

La fonction LeakyReLU est une tentative pour résoudre le problème de ReLU, elle permet d'augmenter la portée de la fonction ReLU. Habituellement, la valeur de a (voir figure 4.3) est 0,01. Par conséquent, la plage du Leaky ReLU est de moins l'infini à l'infini. Il est déconseillé d'utiliser cette fonction dans la dernière couche d'un réseau de neurone, elle est utilisée dans les couches intermédiaires [1].

2. Sigmoid : La raison principale pour laquelle nous utilisons la fonction sigmoïde est parce qu'elle est entre 0 et 1 (voir figure 4.4). Par conséquent, elle est particulièrement utilisée pour les modèles où nous devons prédire la probabilité comme une sortie. Puisque la probabilité de n'importe quoi existe seulement entre 0 et 1. elle fonctionne efficacement lorsqu'il n'y a que deux catégories d'étiquettes ciblent. Par exemple, oui ou non [1].

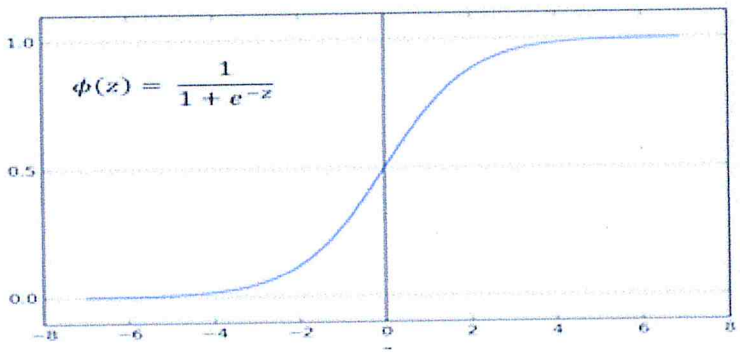

FIGURE 4.4 – Le graphe de la fonction sigmoïde.

1. https ://towardsdatascience.com/activation-functions-neural-networks-1cbd9f8d91d6

3. Tanh : Cette fonction est aussi comme la fonction sigmoïde mais mieux. La plage de la fonction tanh va de -1 à 1 (voir figure 4.5). L'avantage est que les entrées négatives seront cartographiées fortement négatives et les entrées nulles seront mappées près de zéro dans le graphe.La fonction tanh est principalement utilisée pour la classification entre deux classes [1].

   Sa formule est la suivante : $\tanh(x) = \frac{\sinh(x)}{\cosh(x)} = (e^x - e^{-x})/(e^x + e^{-x})$

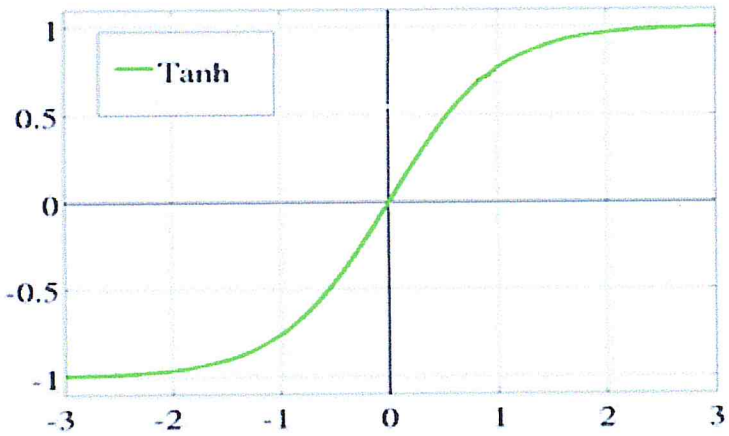

FIGURE 4.5 – Le graphe de la fonction tanh.

4. Softmax : La fonction Softmax calcule la distribution des probabilités de l'événement sur 'n' différents événements. En général, cette fonction calcule les probabilités de chaque classe cible sur toutes les classes cibles possibles. Les probabilités calculées seront utiles pour déterminer la classe cible pour les entrées données. L'avantage principal de l'utilisation de Softmax est la plage des probabilités de sortie qui est entre 0 et 1 (voir figure 4.6), et la somme de toutes les probabilités sera égale à 1. Si la fonction softmax est utilisée pour le modèle multi-classification, elle renvoie les probabilités de chaque classe et la classe cible aura la forte probabilité [2].

   Sa formule est la suivante : pour $j = 1, 2, \ldots, K$

1. https ://towardsdatascience.com/activation-functions-neural-networks-1cbd9f8d91d6
2. https ://en.wikipedia.org/wiki/Softmaxfunction

$$\sigma(z)_j = \frac{e^{z_j}}{\sum_{k=1}^{K} e^{z_k}}$$

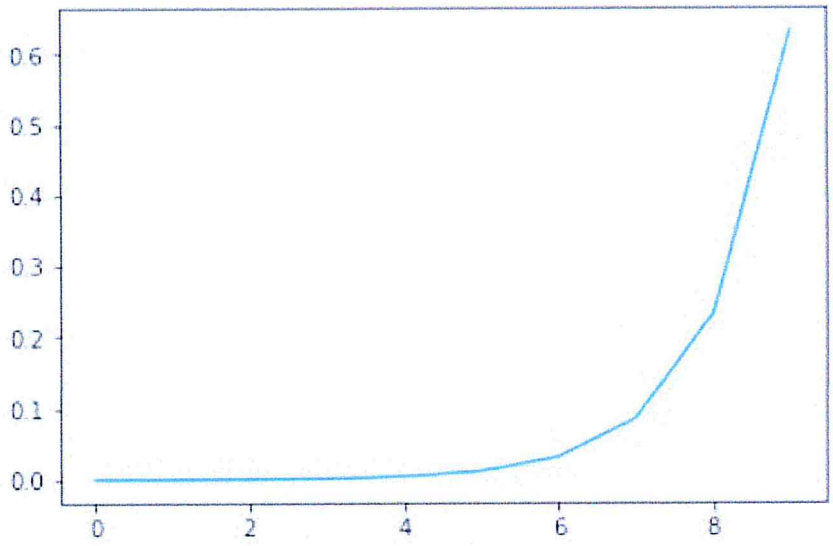

FIGURE 4.6 – Le graphe de la fonction softmax.

4.3.2.4 L'ARCHITECTURE NEURONALE :

Notre modèle est un GAN, et comme nous avons défini dans le chapitre 3 : le GAN est composé de deux modèles :

1. Le modèle génératif : qui permet de générer de données (pour notre cas les résumés multidocuments des articles), d'habitude le générateur c'est un réseau de neurone avec un nombre défini de couches, et des fonctions d'activation. Mais pour notre modèle, nous avons mis un autre modèle génératif à la place du modèle génératif classique qui est le VAE, qui va être détaillé en dessous (après le GAN).
2. le modèle discriminatif : permet de déterminer si un échantillon (résumé) provient du modèle génératif, ou bien c'est un résumé de référence (généré par un expert). Dans notre modèle, le modèle discriminatif contient une couche d'entrée de taille 100, car nous voulons générer des résumés de 100 mots, et une couche cachée de taille 100 avec une fonction d'activation « LeakyReLU », et une couche de sortie de taille 100 aussi avec une fonction d'activation « Sigmoid ».

La figure 4.7 est un schéma représentatif du GAN :

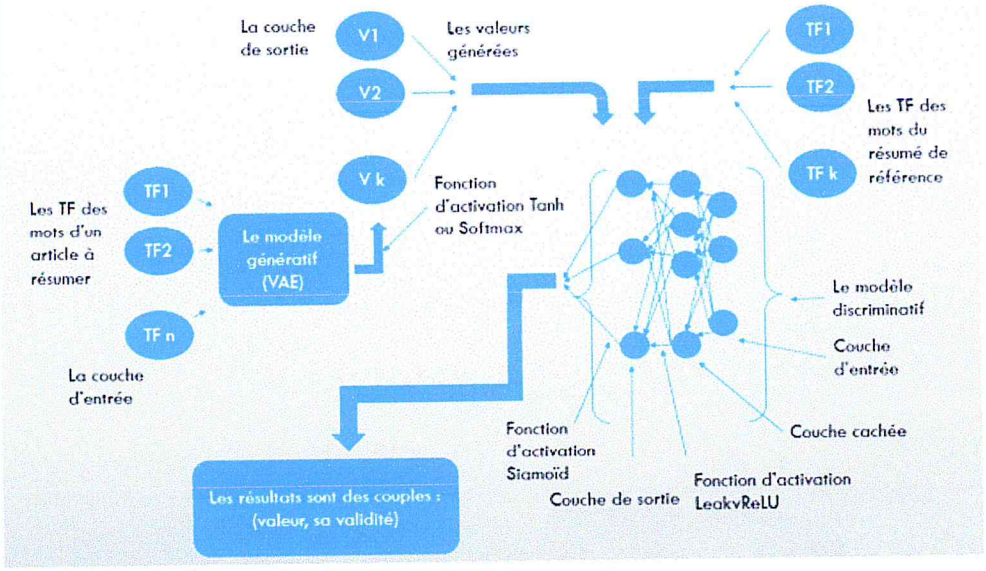

FIGURE 4.7 – Schéma représentatif du GAN.

Le modèle génératif est un VAE, un VAE à la même définition qu'un Auto-Encoder (définit dans le chapitre 3). Notre VAE est composé de deux composantes importantes :

1. Encoder : Permet d'encoder l'entrée en une représentation latente de taille 100. Il prend l'entrée X (qui représente les TF des mots) et sort deux choses : μ (X) et Σ (X), les paramètres de la distribution Gaussienne. C'est un réseau de neurone récurent bidirectionnel, il a une couche d'entrée de taille 100, trois couches cachées de taille 318 (pour redimensionner l'entrée dans un espace plus large), avec des cellules de type GRU (qui sont définit dans le chapitre 3), et l'utilisation de la technique Dropout [1] avec un pourcentage de 20 % qui permet d'améliorer l'entrainement en ignorant chaque fois 20% des nœuds pour améliorer le modèle génératif, et à la fin une fonction d'activation LeakyReLU avec une taille de 100 nœuds.
2. Decoder : Permet de décoder la représentation latente en une sortie équivalente à l'entrée mais avec une certaine variation. C'est un réseau de neurone récurent bidirectionnel. Il dispose d'une couche d'entrée de taille 100, trois couches cachées de taille 100, avec

1. https ://leonardoaraujosantos.gitbooks.io/artificial-inteligence/content/dropout-layer.html

des cellules de type GRU (qui sont définies dans le chapitre 3), et l'utilisation de la technique Dropout [1] avec un pourcentage de 20 % qui permet d'améliorer l'entrainement en ignorant chaque fois 20% des nœuds pour améliorer le modèle génératif, et à la fin une fonction d'activation (Tanh ou bien Softmax) avec une taille de sortie de 100 nœuds.

La figure 4.8 est un schéma représentatif du VAE :

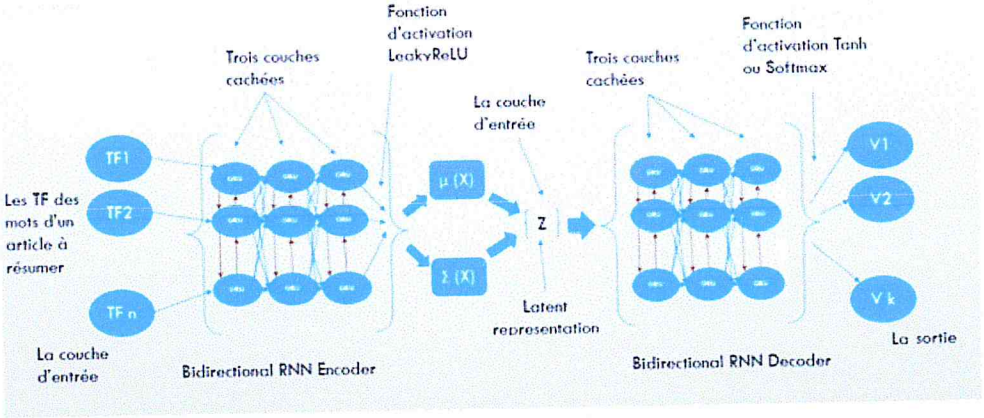

FIGURE 4.8 – Schéma représentatif du VAE.

4.3.2.5 GÉNÉRATION DES RÉSUMÉS :

Pour la génération des résumés (voir figure 4.9) :

1. Nous récupérons les fichiers prétraités.
2. L'entrainement de modèles discriminatif et génératif se fait en parallèle, pour le discriminateur l'entrainement se fait en deux phases : pour les TF de mots de résumés réels, et les valeurs générées par le générateur. Pour le générateur nous avons une seule phase pour les TF de mots des articles.
3. Les résumés sont générés chaque 5 époques, en utilisant un dictionnaire de mots cibles issu des articles donnés en entrée avec leur indices (chaque mot a un indice particulier).
4. Pour la génération des résumés :
   a) Nous faisons appel au générateur pour nous prédire des résultats, en lui donnant les TF des mots de chaque article pour générer son résumé.
   b) Nous redimensionnons les valeurs prédites selon le dictionnaire des indices.
   c) Nous récupérons les mots selon les indices, et nous les utilisons pour générer des fichiers de format txt de 100 mots.

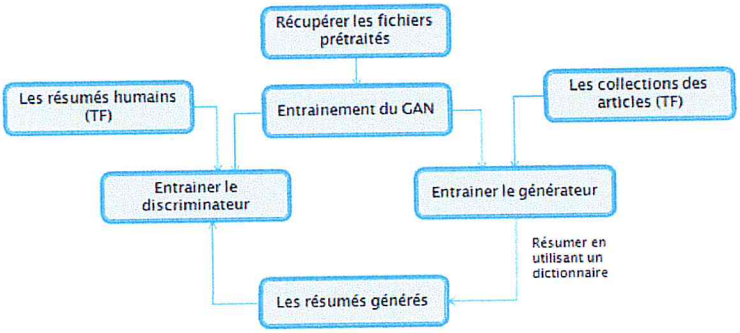

FIGURE 4.9 – Organigramme qui représente les étapes de génération des résumés.

4.3.3 LES POST-TRAITEMENTS :

La figure 4.10 représente les post-traitements du traitement précédent :

1. Pour chaque époque (itération) nous avons trois valeurs générées pendant l'entrainement, et qui sont la valeur de la fonction de perte du modèle discriminatif, la valeur de la fonction de perte du modèle génératif, et la valeur de la précision.
2. Nous avons implémenté un code qui permet d'ordonner les phrases et d'ajouter les mots vides pour les résumés initiaux générés, selon les étapes suivantes :
   a) Nous récupérons le contenu d'un résumé généré mot par mot.
   b) Nous associons à chaque mot son étiquette grammaticale (verbe, nom ou autre).
   c) Nous utilisons une liste qui contient les mots vides les plus utilisés en anglais.
   d) Nous re-ordonnons les phrases en choisissant aléatoirement des noms, des verbes, et les concaténons dans l'ordre (nom, verbe, autre).
   e) Nous mettons chaque fois un mot vide (choisi aléatoirement) pour séparer les phrases.

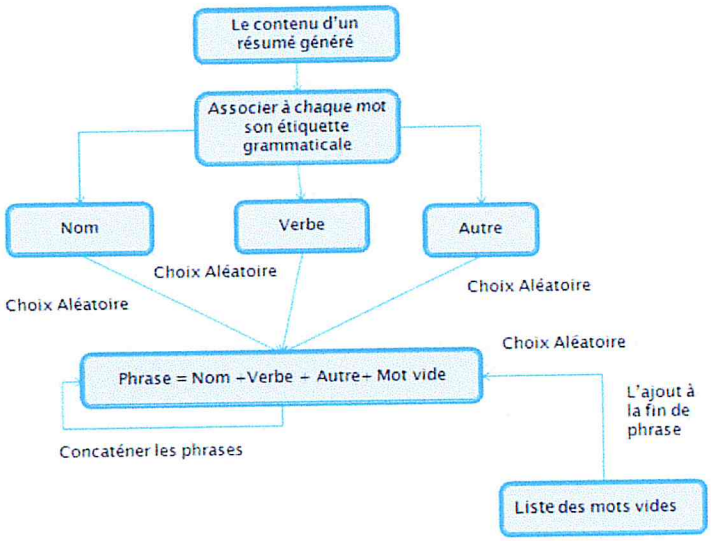

FIGURE 4.10 – Organigramme qui représente les étapes de post-traitements.

4.4 CONCLUSION :

Dans ce chapitre nous avons défini les prétraitements, les traitements de notre système de génération des résumés automatiques multidocuments de texte, en détaillant notre modèle hybride et son fonctionnement, et nous avons enfin détailler les post-traitements effectué. Dans le chapitre qui suit nous allons parler de la mise en œuvre ainsi que de l'évaluation de notre système.

CHAPITRE 5

RÉALISATION

5.1 INTRODUCTION :

Il est évident que les méthodes et les outils choisis pour concevoir et développer une application ou un système informatique doivent être en fonction de l'environnement et du domaine d'application de ces derniers.
Ce chapitre nous permet de montrer les résultats de notre système du résumé automatique multidocuments de texte dans sa phase de tests et d'essais, ce qui nous permettra d'envisager les améliorations possibles. Donc, dans ce chapitre nous allons présenter les outils utilisés pour la réalisation de notre système du résumé automatique multidocuments (hardware et software), les outils de l'évaluation des résumés générés, les benchmarks qui existent et en particulier ceux utilisés comme corpus pour l'entrainement et le test de notre système. Nous allons présenter aussi l'évaluation de notre système comme une étude paramétrique et comparative en testant le système avec différentes fonctions d'erreur et fonctions d'activation, et à la fin nous allons conclure ce chapitre par une conclusion récapitulative.

5.2 L'ENVIRONNEMENT DU 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 10 Professionnel 64 BITS afin de mener à bien ce projet, nous avons utilisé un ensemble de matériels dont les principales caractéristiques sont les suivantes :
CPU : Intel® Core™ i7-5500U CPU @ 2.40 GHz.

RAM : 4.00 GO.

5.3 LANGAGE DE PROGRAMMATION :

Nous avons développé notre système du résumé automatique multidocuments de texte, en utilisant le langage python à cause de ses caractéristiques et ses bibliothèques qui sont dédiées pour l'apprentissage automatique et les réseaux de neurone artificiels. Python est un langage de programmation de haut niveau à usage général largement utilisé (voir figure 5.1). Il a été principalement développé pour mettre l'accent sur la lisibilité du code, et sa syntaxe permet aux programmeurs d'exprimer des concepts dans moins de lignes de code [1].

FIGURE 5.1 – Le logo du langage de programmation Python.

IDE signifie Integrated Development Environment, c'est un outil de codage qui vous permet d'écrire, de tester et de déboguer votre code plus facilement, car ils offrent généralement des compléments de code ou de code en mettant en évidence, la gestion des ressources, les outils de débogage, et même si l'IDE est un concept strictement défini, Il commence à être redéfini à mesure que d'autres outils tels que les ordinateurs portables commencent à gagner de plus en plus de fonctionnalités qui appartiennent traditionnellement aux IDE. L'IDE utilisé pour le développement de notre système est Jupyter Notebook, c'est une application web basée sur la structure serveur-client et qui permet de créer et de manipuler des documents de « notebook ». Jupyter Notebook fournit un environnement de science des données interactif et facile à utiliser dans de nombreux langages de programmation qui fonctionne non seulement comme un IDE, mais aussi comme un outil de présentation ou d'éducation. Il prend en charge les démarques, ce qui permet d'ajouter des composants HTML d'images à des vidéos. Il donne la possibilité de voir et modifier votre code facilement afin de créer des présentations convaincantes, en utilisant des bibliothèques de visualisation de données et afficher vos graphiques dans le même document où votre code est. Outre tout cela, il permet d'exporter le travail final en fichiers PDF et HTML, ou simplement l'exporter en tant que fichier .py. En outre, il permet également de créer des blogs et des présentations (voir figure 5.2) [2].

1. https ://fr.wikipedia.org/wiki/Python(langage)
2. https ://www.oreilly.com/ideas/what-is-jupyter

FIGURE 5.2 – Le logo de l'IDE Jupyter Notebook.

5.4 DESCRIPTION DE NOTRE SYSTÈME :

Dans cette partie nous allons présenter l'architecture du fonctionnement du notre système et les bibliothèques utilisées pour l'apprentissage automatique du python.

5.4.1 ARCHITECTURE DU FONCTIONNEMENT :

Nous avons résumé le fonctionnement de notre application dans le schéma de la figure 5.3 :

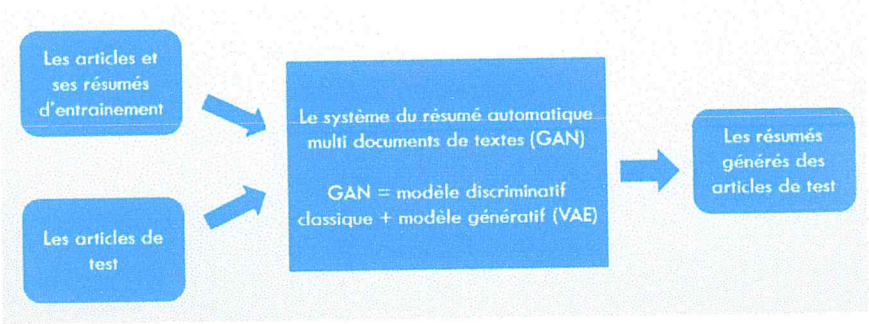

FIGURE 5.3 – Schéma représentatif de l'architecture du fonctionnement du système.

5.4.2 LES BIBLIOTHÈQUES UTILISÉES :

Dans cette partie nous allons présenter les bibliothèques du langage python utilisées pour le développement de notre système :

1. Tensorflow : TensorFlow (TF) est la bibliothèque la plus célèbre utilisée dans la production pour les modèles d'apprentissage en profondeur, elle a une communauté très grande et impressionnante (voir figure 5.4) [1].

1. https ://opensource.com/article/17/11/intro-tensorflow

FIGURE 5.4 – Le logo de la bibliothèque Tensorflow.

2. Keras : Keras est une interface de programation applicative de réseaux de neurones de haut niveau construite sur TensorFlow (TF), Keras est plus convivial et facile à utiliser par rapport à TF, il est capable de fonctionner sur TensorFlow, CNTK ou Theano. il a été développé dans le but de permettre une expérimentation rapide. Il peut passer de l'idée au résultat avec le moins de retard possible qui est la clé pour faire de bonnes recherches. Il permet un prototypage facile et rapide (grâce à la convivialité, la modularité et l'extensibilité). Il prend en charge les réseaux convolutifs et les réseaux récurrents, ainsi que les combinaisons des deux. Il fonctionne de manière transparente sur le processeur et le GPU, et il permet de construire et tester rapidement un réseau de neurones avec un minimum de lignes de code, il permet même de construire des réseaux de neurones simples ou très complexes en quelques minutes à cause du modèle et les API séquentielles qui sont si puissants (voir figure 5.5) [1].

FIGURE 5.5 – Le logo de la bibliothèque Keras.

3. NumPy : NumPy est le package fondamental pour le calcul scientifique avec Python. Il contient entre autres choses : un puissant objet tableau N-dimensional, fonctions sophistiquées (diffusion), outils pour intégrer le code C / C ++ et Fortran, algèbre linéaire utile, transformée de Fourier et capacités de nombres aléatoires, En plus de ses utilisations scientifiques évidentes, NumPy peut également être utilisé comme un conteneur multidimensionnel efficace de données génériques. Des types de données arbitraires peuvent être définis. Cela permet à NumPy de s'intégrer facilement et rapidement à une grande variété de bases de données (voir figure 5.6) [2].

1. https ://medium.com/implodinggradients/tensorflow-or-keras-which-one-should-i-learn-5dd7fa3f9ca0
2. https ://docs.scipy.org/doc/numpy-1.13.0/user/whatisnumpy.html

FIGURE 5.6 – Le logo de la bibliothèque NumPy.

5.5 LES OUTILS DE L'ÉVALUATION :

5.5.1 ROUGE :

ROUGE évalue les résumés en les comparants à des résumés modèles. Cette comparaison est automatique et ne nécessite pas un prétraitement particulier. Elle est déduite à partir du recouvrement entre les N-grammes des deux textes. Cette méthode a montré une forte corrélation avec les jugements humains. 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. Comme l'indique ses premières lettres, ROUGE est orienté rappel (Recall Oriented). 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 (Lin, 2004).

5.5.1.1 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 5.7.

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

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

FIGURE 5.7 – 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* (Lin, 2004)», montre à quel point les scores ROUGE-*n* peuvent être décorrélés de la réalité :

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

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

5.5.1.2 ROUGE-L :

Afin de pallier en partie au défaut de ROUGE-*n*, à savoir le fait que l'ordre des mots dans les phrases n'est pas pris en compte, (Lin, 2004) 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 :

1. phrase2 = (« Dr Jekyll Hide ») = 3/4
2. 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 ».

5.5.1.3 ROUGE-SUn :

ROUGE-SU, pour *skip-bigram and unigram ROUGE* prend en compte des bi-grammes à trou ainsi que les uni-grammes des résumés de référence et du résumé cible. Les bi-grammes à trou (*skip-bigram*) sont définis dans (Lin, 2004) 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 comprend 6 bi-grammes et 4 uni-grammes :
« Dr Jekyll », « Dr tua », « Dr Hide », « Jekyll tua », « Jekyll Hide », « tua Hide » et les 4 uni-grammes 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.

5.5.2 PYRAMID :

Cette méthode permet de comparer un résumé candidat à un ensemble de résumés de référence(Passonneau, Chen, Guo & Perin, 2013). 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. Malheureusement, l'annotation des résumés modèles reste difficile à automatiser.

5.6 LES BENCHMARKS :

5.6.1 DOCUMENT UNDERSTANDING CONFERENCE (DUC) :

Depuis 2001, le National Institute of Standards and Technology 8 (NIST) organise la campagne d'évaluation Document Understanding Conference (DUC) (Favre & al., 2006). Son but est de promouvoir les progrès réalisés dans le domaine du résumé automatique de textes mais surtout de permettre aux chercheurs de participer à des expérimentations de grande envergure tant au point de vue du développement que de l'évaluation de leurs systèmes. Les campagnes DUC ont successivement introduit des tâches visant à produire des résumés génériques mono et multi documents (2001 - 03), des résumés courts mono et multi documents (2003 - 04), des résumés orientés multi documents (2003 - 07) et des résumés mis à jours orientés multi documents (2007 - 08) (Favre & al., 2006), (Boudin, Favre, Bechet, El-Beze, Gillard & Torres-Moreno, 2007). La tâche principale est de produire un résumé d'une taille maximum, tout en tenant compte d'un besoin utilisateur. Ce besoin utilisateur, appelé topic en anglais, s'exprime sous la forme d'un titre et d'un ensemble de questions. Dans le cadre de ces campagnes, l'évaluation des systèmes est réalisée de manière intrinsèque sur le fond ainsi que sur la forme des résumés produits. Pour la forme, une note qualitative est attribuée par des juges selon les critères linguistiques suivants :

1. Q1 Grammaticality (grammaticalité) : le résumé ne doit pas contenir de problèmes de formatage, d'erreurs de capitalisation ou de phrases clairement agrammaticales (e.g., parties manquantes, fragmentation) qui rendent difficile la lecture du texte.
2. Q2 Non-redundancy (non redondance) : il ne doit pas y avoir de répétitions non nécessaires dans le résumé.
3. Q3 Referential clarity (clarté des références) : on doit pouvoir identifier à qui ou à quoi les pronoms et les groupes nominaux se réfèrent dans le résumé.
4. Q4 Focus (cible) : le résumé ne doit pas contenir d'informations sortant de la thématique désirée.
5. Q5 Structure and Coherence (structure et cohérence) : le résumé doit être structuré, les informations doivent être disposées de manière cohérente.

Une note comprise entre 1 et 5 est donnée pour chacun des critères linguistiques : 1 correspondant à très mauvais et 5 à très bon (équivalent à une production humaine). Cette évaluation

manuelle est primordiale et justifie le succès des campagnes DUC. Elle est à notre connaissance la seule et unique façon de juger la forme d'un résumé.
Maintenant nous allons définir le détail de prétraitements faits avant de commencer la tâche du résumé de textes. Pour cela nous avons exploité le corpus DUC (Document Understanding Conference) (Favre & al., 2006), (Harabagiu & Lacatusu, 2002), et en particulier duc2003, et parce que nous sommes intéressées par le résumé multidocuments de textes. Nous disposons d'un ensemble des articles en anglais qui sont distribués en 60 dossiers. Chaque dossier contient en moyenne 10 articles de format xml qui traitent de la même thématique. Nous avons également un ensemble de résumés de format html pour ces derniers qui veut dire, pour chaque dossier nous avons un résumé en 100 mots qui est généré manuellement (par un expert), et qui représente la référence pour le résumé qui doit être généré.
Pour pouvoir exploiter les fichiers de format xml, nous devons les rendre valide en ajoutant leur DTD (Définition de Type de Document). Un document xml valide est un document bien formé conforme à une définition. Cela signifie que le document xml respecte toutes les règles qui lui sont imposées dans les fameuses définitions (pour notre cas selon les fameuses définitions de duc2003).
Pour générer des résumés de type abstractif multidocuments de textes, nous sommes obligées de prendre 70% de ce corpus pour l'entrainement de notre modèle, et 30% pour le test. Et donc 42 dossiers avec ses résumés pour l'entrainement, et 18 dossiers sans ses résumés pour le test. Apres, pour chaque dossier, nous récupérons les contenus de tous ses articles, et nous appliquons les prétraitements présentés dans le chapitre 4.

5.6.2 GIGAWORD :

Gigaword est actuellement le plus grand corpus statique de documents d'information en anglais disponibles. L'ajout le plus récent, Gigaword v.5 (Parker, Gra, Kong, Chen & Maeda, 2011), contient près de 10 millions de documents provenant de sept médias, avec un total de plus de 4 milliards de mots. Cette ressource

1. Fournit un ensemble cohérent d'annotations de pointe, sur lesquelles les chercheurs peuvent comparer les résultats.
2. Empêche le redoublement des efforts d'annotation par différents groupes de recherche.
3. Permet à ceux qui n'ont pas la capacité de calcul de générer de telles annotations à grande échelle (Napoles, Gormley & Van Durme, 2012).

5.7 ETUDE PARAMÉTRIQUE ET ÉVALUATION :

5.7.1 NOMBRE IDÉAL D'ÉPOQUE :

Dans cette partie nous allons présenter le choix de l'époque idéale où notre système se stabilise. Pour cela nous avons entrainé le code implémenté de notre modèle avec plusieurs fonctions d'activation et fonctions d'erreur de VAE, pour la fonction d'erreur et la fonction d'activation de GAN sont respectivement « Binary Cross Entropy » et « Sigmoid » pour toute l'étude car nous visons à améliorer le modèle génératif alors que le discriminateur est un modèle classique et simple, nous avons remarqué que pour toutes les variantes utilisées l'époque 100 est l'époque moyenne où notre modèle se stabilise, nous présentant en dessous les résultats des codes pour un ensemble de fonctions d'activation et de fonctions d'erreur utilisées.
Les résultats avec la fonction d'activation tanh et la fonction d'erreur bayes de VAE :

Intervalle des époques	De 0 à 3	De 4 à 10	De 11 à 31	De 32 à 35	De 36 à 171
La précision moyenne	47.28%	49.23%	49.63%	50.09%	49.28%

TABLE 5.1 – les valeurs de la précision en fonction des intervalles des époques.

Les intervalles des époques sont choisis selon les résultats, car nous avons remarqué que :

1. De l'époque 0 à l'époque 3 les valeurs de la précision sont proches (entre 46.23% et 47.34%).
2. De l'époque 4 à l'époque 10 les valeurs de la précision sont entre 48.52% et 49.53%.
3. De l'époque 11 à l'époque 31 les valeurs de la précision sont entre 49.12% et 50.22%.
4. De l'époque 32 à l'époque 35 les valeurs de la précision sont entre 50.02% et 50.59%.
5. De l'époque 36 à l'époque 171 les valeurs de la précision sont un mélange aléatoire des valeurs entre 48%, 49%, 50% et 51%.

Pour ces fonctions d'erreur et d'activation, lors de l'entrainement la fonction d'erreur du discriminateur et la fonction d'erreur du générateur varient entre 0.717399 et 0.692993.
Les résultats avec la fonction d'activation softmax et la fonction d'erreur binary cross entropy de VAE :

Intervalle des époques	De 0 à 8	De 9 à 11	12	13	14	De 15 à 16	De 17 à 150
La précision moyenne	51.25%	52.11%	56.64%	70.08%	82.25%	93.62%	100%

TABLE 5.2 – les valeurs de la précision en fonction des intervalles des époques.

Les intervalles des époques sont choisis selon les résultats, car nous avons remarqué que :

1. De l'époque 0 à l'époque 8 les valeurs de la précision sont entre 50.48% et 52.09%.
2. De l'époque 9 à l'époque 11 les valeurs de la précision sont entre 51.31% et 53.56%.
3. De l'époque 15 à l'époque 16 les valeurs de la précision sont entre 91.14% et 96.10%.
4. De l'époque 17 à l'époque 150 la valeur de la précision se stabilise à 100%.

Pour ces fonctions d'erreur et d'activation, lors de l'entrainement la fonction d'erreur du discriminateur tend vers 0 et la fonction d'erreur du générateur tend vers l'infini.
Les résultats avec la fonction d'activation tanh et la fonction d'erreur mean squared error de VAE :

Intervalle des époques	De 0 à 1	De 2 à 3	De 4 à 6	De 7 à 12	De 13 à 17	De 18 à 24	De 25 à 154
La précision moyenne	47.80%	45.13%	47.52%	48.57%	49.18%	50.44%	49.68%

TABLE 5.3 – les valeurs de la précision en fonction des intervalles des époques.

Les intervalles des époques sont choisis selon les résultats, car nous avons remarqué que :

1. De l'époque 0 à l'époque 1 les valeurs de la précision sont entre 46.47% et 49.14%.
2. De l'époque 2 à l'époque 3 les valeurs de la précision sont entre 45.02% et 45.25%.
3. De l'époque 4 à l'époque 6 les valeurs de la précision sont entre 47.25% et 47.77%.
4. De l'époque 7 à l'époque 12 les valeurs de la précision sont entre 48.32% et 48.95%.
5. De l'époque 13 à l'époque 17 les valeurs de la précision sont entre 49.16% et 49.59%
6. De l'époque 18 à l'époque 24 les valeurs de la précision sont entre 50.21% et 52.02%.

7. De l'époque 25 à l'époque 154 la valeur de la précision est un mélange aléatoire des valeurs entre 48%, 49%, 50%, 51% et 52%.

Pour ces fonctions d'erreur et d'activation, lors de l'entrainement la fonction d'erreur du discriminateur et la fonction d'erreur du générateur varient entre 0.714367 et 0.689749.
Donc selon les trois tableaux, nous remarquons que toutes les valeurs des tableaux sont stables entre les époques 36 et 150, pour cela nous choisissons l'époque 100 comme l'époque idéale pour l'entrainement de notre système de résumé automatique multi-documents.

5.7.2 MEILLEURE FONCTION D'ACTIVATION ET FONCTION D'ERREUR :

Dans cette partie, nous allons présenter les tableaux qui contiennent les évaluations de notre système en utilisant ROUGE et selon :

1. Les époques 0, 5, 10, 20 et 100.
2. Les deux mesures ROUGE-1 et ROUGE-SU4 de ROUGE.
3. Pour les fonctions d'activation Tanh et softmax de la dernière couche de VAE.
4. Pour les fonctions d'erreur Binary Cross Entropy, Bayes et Mean Squared Error du modèle génératif (VAE).

Les valeurs de mesures de ROUGE sont entre 0 (mauvaise valeur) et 1 (bonne valeur) :

5.7.2.1 LA FONCTION D'ACTIVATION SOFTMAX ET LA FONCTION D'ERREUR BAYES :

Voici un tableau qui montre les résultats de l'évaluation avec ROUGE pour ce cas :

	Epoch 0	Epoch 5	Epoch 10	Epoch 20	Epoch 100
ROUGE_1	0,65322667	0,63481167	0,66031167	0,63401333	0,61623222
ROUGE_SU4	0,20594056	0,20336111	0,2373464	0,19958667	0,19857944

TABLE 5.4 – Les valeurs de mesures de ROUGE en fonction des époques.

Nous commençons par démontrer les résultats de l'évaluation pour chaque époque, et pour chaque mesure de ROUGE :

1. Pour l'époque 0 : la valeur de ROUGE_1 est 0.65322667, qui veut dire que notre système arrive à générer des mots existant dans les résumés de référence, et avec une bonne précision qui est égal à 78.79%, car la valeur de cette mesure est supérieure à 0.5. Pour la valeur de ROUGE_SU4 est égal à 0.20594056, c'est une valeur pas vraiment bonne, car cette mesure prend chaque fois une paire de mots dans l'ordre d'une phrase séparée par une distance maximale arbitraire pour les résumés générés et les résumés de référence et faire une comparaison entre les deux, ça veut dire que les résumés générés contiennent des mots qui existe dans les résumés de référence mais pas exactement avec le même ordre, ni exactement la même distance entre chaque paire de mots.
2. Pour l'époque 5 : nous remarquons que les valeurs de ROUGE sont inférieures à celles de l'époque précédente, et ça à cause de la précision qui a diminué de 78.79% à 50.00% lors de l'entrainement.
3. Pour l'époque 10 : nous remarquons que les valeurs de ROUGE ont atteint les valeurs maximales pour cette évaluation, avec ROUGE_1 = 0,66031167 et ROUGE_SU4 = 0,2373464, et ça à cause de la précision qui a atteint la valeur 100.00%.
4. Pour l'époque 20 : nous remarquons dans cette époque que les valeurs de ROUGE sont inférieures à celles de l'époque précédente, avec ROUGE_1 = 0,63401333 et ROUGE_SU4 = 0,19958667, malgré que la valeur de la précision est toujours 100.00%, et ça à cause de la valeur de la fonction d'erreur du modèle génératif qui est en train d'augmenter lors de l'entrainement de 0.725107 à 3.110015.
5. Pour l'époque 100 : dans cette époque les valeurs de ROUGE sont inférieures à celles de l'époque précédente, avec ROUGE_1 = 0,61623222 et ROUGE_SU4 = 0,19857944, malgré que la valeur de la précision est toujours 100.00%, et ça à cause de la valeur de la fonction d'erreur du modèle génératif qui a atteint la valeur 8.910643.

5.7.2.2 LA FONCTION D'ACTIVATION SOFTMAX ET LA FONCTION D'ERREUR BINARY CROSS ENTROPY :

Voici un tableau qui montre les résultats de l'évaluation avec ROUGE pour ce cas :

	Epoch 0	Epoch 5	Epoch 10	Epoch 20	Epoch 100
ROUGE_1	0,52118167	0,54014	0,51365222	0,50597389	0,52142611
ROUGE_SU4	0,16625722	0,19204167	0,17862	0,178645	0,18152778

TABLE 5.5 – Les valeurs de mesures de ROUGE en fonction des époques.

Nous commençons par démontrer les résultats de l'évaluation pour chaque époque, et pour chaque mesure de ROUGE :

1. Pour l'époque 0 : la valeur de ROUGE_1 est 0,52118167, qui veut dire que notre système arrive à générer des mots existant dans les résumés de référence, et avec une bonne précision qui est égal à 50.90% car la valeur de cette mesure est supérieure à 0.5. Pour la valeur de ROUGE_SU4 est égal à 0,16625722, c'est une valeur pas vraiment bonne.
2. Pour l'époque 5 : nous remarquons que les valeurs de ROUGE sont supérieures à celles de l'époque précédente, avec ROUGE_1 = 0,54014 et ROUGE_SU4 = 0,19204167, et ça à cause de la précision qui a augmenté de 50.90% à 52.02% lors de l'entrainement.
3. Pour l'époque 10 : nous remarquons que les valeurs de ROUGE sont inférieures à celles de l'époque précédente, avec ROUGE_1 = 0,51365222 et ROUGE_SU4 = 0,17862, car la précision a diminué de 52.02% à 51.48%.
4. Pour l'époque 20 : nous remarquons dans cette époque que les valeurs de ROUGE sont inférieures à celles de l'époque précédente, avec ROUGE_1 = 0,50597389 et ROUGE_SU4 = 0,178645, malgré que la valeur de la précision est devenue 100.00%, et ça à cause de la valeur de la fonction d'erreur du modèle génératif qui est en train d'augmenter lors de l'entrainement de 0.689254 à 0.787993.
5. Pour l'époque 100 : dans cette époque les valeurs de ROUGE sont supérieures à celles de l'époque précédente, avec ROUGE_1 = 0,52142611 et ROUGE_SU4 = 0,18152778, malgré le fait que la valeur de la précision soit toujours 100.00%. La fonction d'erreur du modèle génératif augmente également, et cela à cause de la fonction d'erreur du modèle génératif utilisée qui est binary cross entropy, qui est malgré en train d'augmenter mais avec un intervalle réduit par rapport à la fonction d'erreur bayésienne, car avec la fonction bayésienne nous avons trouvé l'erreur à l'époque 100 égal à 8.910643, et avec la fonction d'erreur binary cross entropy la valeur à l'époque 100 est égal à 7.830611.

5.7.2.3 La fonction d'activation Softmax et la fonction d'erreur Mean Squared Error :

Voici un tableau qui montre les résultats de l'évaluation avec ROUGE pour ce cas :

	Epoch 0	Epoch 5	Epoch 10	Epoch 20	Epoch 100
ROUGE_1	0,51257783	0,52080222	0,51991056	0,50832	0,52274833
ROUGE_SU4	0,16861444	0,16430889	0,16295778	0,16064778	0,16904278

TABLE 5.6 – Les valeurs de mesures de ROUGE en fonction des époques.

Nous commençons par démontrer les résultats de l'évaluation pour chaque époque, et pour chaque mesure de ROUGE :

1. Pour l'époque 0 : la valeur de ROUGE_1 est 0,51257783, qui veut dire que notre système arrive à générer des mots existant dans les résumés de référence, et avec une bonne précision car sa valeur est supérieure à 0.5. Pour la valeur de ROUGE_SU4 est égal à 0,16861444, c'est une valeur pas vraiment bonne.
2. Pour l'époque 5 : nous remarquons que les valeurs de ROUGE sont presque les mêmes que celles de l'époque précédente, avec ROUGE_1 = 0,52080222 qui est un petit peu supérieur et ROUGE_SU4 = 0,16430889 qui est presque la même, et ça à cause de la précision qui a augmenté de 47.95% à 51.07% lors de l'entrainement.
3. Pour l'époque 10 : nous remarquons que les valeurs de ROUGE sont presque les mêmes que celles de l'époque précédente, avec ROUGE_1 = 0,51991056 qui est un petit peu inférieur et ROUGE_SU4 = 0,16295778 qui est presque la même, car la précision a diminué de 51.07% à 50.50%.
4. Pour l'époque 20 : nous remarquons dans cette époque que les valeurs de ROUGE sont presque les mêmes à celles de l'époque précédente, avec ROUGE_1 = 0,50832 qui un petit peu inférieur et ROUGE_SU4 = 0,16064778 qui est presque la même, malgré que la valeur de la précision a devenu 100.00%, et ça à cause de la valeur de la fonction d'erreur du modèle génératif qui est en train d'augmenter lors de l'entrainement de 0.683520 à 0.773299.
5. Pour l'époque 100 : dans cette époque la valeur de ROUGE-1 est supérieure à celle de l'époque précédente, avec ROUGE_1 = 0,52274833 et ROUGE_SU4 = 0,16904278 est presque la même, et ça à cause de la valeur de la précision qui est égale à 100.00%.

5.7.2.4 LA FONCTION D'ACTIVATION TANH ET LA FONCTION D'ERREUR BAYES :

Voici un tableau qui montre les résultats de l'évaluation avec ROUGE pour ce cas :

	Epoch 0	Epoch 5	Epoch 10	Epoch 20	Epoch 100
ROUGE_1	0.33707722	0.50372167	0.519865	0.50844333	0.47733
ROUGE_SU4	0.10309278	0.15017	0.174914	0.158824	0.162168

TABLE 5.7 – Les valeurs de mesures de ROUGE en fonction des époques.

Nous commençons par démontrer les résultats de l'évaluation pour chaque époque, et pour chaque mesure de ROUGE :

1. Pour l'époque 0 : la valeur de ROUGE_1 est 0.33707722, qui veut dire que notre système arrive à générer des mots existant dans les résumés de référence, et avec une précision égale à 46.23% et qui n'est pas vraiment bonne, car la valeur de cette mesure est inférieure à 0.5. La valeur de ROUGE_SU4 est égale à 0.10309278. C'est une valeur non satisfaisante.
2. Pour l'époque 5 : nous remarquons que les valeurs de ROUGE sont supérieures par rapport à celles de l'époque précédente, avec ROUGE_1 = 0.50372167 et ROUGE_SU4 = 0.15017. Ceci est dû au fait que la précision qui ait augmenté de 46.23% à 49.53% lors de l'entrainement.
3. Pour l'époque 10 : nous remarquons que les valeurs de ROUGE sont supérieures à celles de l'époque précédente, avec ROUGE_1 = 0.519865 et ROUGE_SU4 = 0.174914, malgré la diminution de la précision de 49.53% à 48.74%. Cette augmentation de valeurs est expliquée par la diminution de la valeur de la fonction d'erreur du modèle génératif de 0.693349 à 0.693088.
4. Pour l'époque 20 : nous remarquons dans cette époque que les valeurs de ROUGE sont inférieures à celles de l'époque précédente, avec ROUGE_1 = 0.50844333 et ROUGE_SU4 = 0.158824, et ça à cause des valeurs de la précision et de la fonction d'erreur du modèle génératif qui sont presque les mêmes.
5. Pour l'époque 100 : dans cette époque la valeur de ROUGE-1 est inférieure à celle de l'époque précédente, avec ROUGE_1 = 0.47733 et ROUGE_SU4 est supérieure à celle de l'époque précédente, ROUGE_SU4 =0.162168, et ça à cause de la précision qui a augmenté de 49.75% à 50.23%, et la valeur de la fonction d'erreur du modèle génératif qui a augmenté de 0.693010 à 0.693247.

5.7.2.5 LA FONCTION D'ACTIVATION TANH ET LA FONCTION D'ERREUR BINARY CROSS ENTROPY :

Voici un tableau qui montre les résultats de l'évaluation avec ROUGE pour ce cas :

	Epoch 0	Epoch 5	Epoch 10	Epoch 20	Epoch 100
ROUGE_1	0,34165111	0,5614	0,4386	0,4386	0,45614
ROUGE_SU4	0,13826278	0,15951	0,11656	0,13804	0,1319

TABLE 5.8 – Les valeurs de mesures de ROUGE en fonction des époques.

Nous commençons par démontrer les résultats de l'évaluation pour chaque époque, et pour chaque mesure de ROUGE :

1. Pour l'époque 0 : la valeur de ROUGE_1 est 0,34165111, qui veut dire que notre système arrive à générer des mots existant dans les résumés de référence, et avec une précision égal à 47.56% et qui n'est pas vraiment bonne, car la valeur de cette mesure est inférieure à 0.5. Pour la valeur de ROUGE_SU4 qui est égale à 0,13826278, c'est une valeur non satisfaisante.
2. Pour l'époque 5 : nous remarquons que les valeurs de ROUGE sont supérieures par rapport à celles de l'époque précédente, avec ROUGE_1 = 0,5614 et ROUGE_SU4 = 0,15951, et ça à cause de la précision qui a augmenté de 47.56% à 48.33%lors de l'entrainement.
3. Pour l'époque 10 : nous remarquons que les valeurs de ROUGE sont inférieures à celles de l'époque précédente, avec ROUGE_1 = 0,4386 et ROUGE_SU4 = 0,11656, malgré que la précision ait augmenté de 48.33% à 49.27%, et ça à cause de la valeur de la précision qui n'est pas stable entre l'époque 0 et l'époque 10 (elle variée arbitrairement entre 47%, 48%, 49% et 50%).
4. Pour l'époque 20 : nous remarquons dans cette époque que les valeurs de ROUGE sont presque les mêmes que celles de l'époque précédente, avec ROUGE_1 = 0,4386 exactement la même, et ROUGE_SU4 = 0,13804 qui est supérieure à celle de l'époque précédente, et ça à cause de la valeur de la précision qui s'est stabilisé de l'époque 10 à l'époque 20 dans les environs de 49%, avec une valeur égal à 49.43% pour cette époque.
5. Pour l'époque 100 : dans cette époque la valeur de ROUGE-1 est supérieure à celle de l'époque précédente, avec ROUGE_1 = 0,45614 et ROUGE_SU4 est inférieure à celle

de l'époque précédente, ROUGE_SU4 =0,1319, et ça à cause de la précision qui n'est pas stable (entre 47%, 48%, 49%, 50%) de l'époque 20 à l'époque 100, et la valeur de la fonction d'erreur du modèle génératif qui a augmenté de 0.692947 à 0.693205.

5.7.2.6 LA FONCTION D'ACTIVATION TANH ET LA FONCTION D'ERREUR MEAN SQUARED ERROR :

Voici un tableau qui montre les résultats de l'évaluation avec ROUGE pour ce cas :

	Epoch 0	Epoch 5	Epoch 10	Epoch 20	Epoch 100
ROUGE_1	0,26453611	0,45614	0,52632	0,4386	0,5263
ROUGE_SU4	0,115478	0,14417	0,16564	0,1227	0,1411

TABLE 5.9 – Les valeurs de mesures de ROUGE en fonction des époques.

Nous commençons par démontrer les résultats de l'évaluation pour chaque époque, et pour chaque mesure de ROUGE :

1. Pour l'époque 0 : la valeur de ROUGE_1 est 0,26453611, qui veut dire que notre système arrive à générer des mots existant dans les résumés de référence, et avec une précision qui n'est pas bonne, car sa valeur est inférieure à 0.5. Pour la valeur de ROUGE_SU4 est égal à 0,115478.
2. Pour l'époque 5 : nous remarquons que les valeurs de ROUGE sont supérieures par rapport à celles de l'époque précédente, avec ROUGE_1 = 0,45614 et ROUGE_SU4 = 0,14417, et ça à cause de la précision qui a augmenté de 46.47% et 45.02% à 47.55% lors de l'entrainement.
3. Pour l'époque 10 : nous remarquons que les valeurs de ROUGE sont supérieures à celles de l'époque précédente, avec ROUGE_1 = 0,52632 et ROUGE_SU4 = 0,16564, et ça à cause de la précision a augmenté de 47.55% à 48.45%.
4. Pour l'époque 20 : nous remarquons dans cette époque que les valeurs de ROUGE sont inférieures à celles de l'époque précédente, avec ROUGE_1 = 0,4386 et ROUGE_SU4 = 0,1227, et ça à cause de la précision qui n'est pas stable dans l'intervalle de l'époque actuelle et l'époque précédente (entre 48%, 49%, 52% et 50%).
5. Pour l'époque 100 : dans cette époque les valeurs de ROUGE sont supérieures à celles de l'époque précédente, avec ROUGE_1 = 0,5263 et ROUGE_SU4 =0,1411 , et ça à cause de l'instabilité de la précision (parfois elle dépasse 50% dans cet intervalle d'époques).

Selon cette étude comparative la fonction d'activation Softmax avec la fonction d'erreur de Bayes ont donné de meilleurs résultats par rapport aux autres fonctions.

5.8 CONCLUSION :

Dans ce chapitre nous avons présenté la réalisation de notre système, en commençant par définir l'environnement du travail, le langage de programmation, les étapes de notre système (l'architecture du fonctionnement et les bibliothèques utilisées), et les résultats de l'évaluation de notre système. Nous avons trouvé que la fonction d'activation Softmax avec la fonction d'erreur de Bayes ont donné de meilleurs résultats par rapport aux autres fonctions.

CONCLUSION ET PERSPECTIVES

Dans ce travail notre objectif était d'améliorer le système du résumé automatique multi-documents de type abstractif. Pour ce faire nous avons basé sur l'approche structurelle, en choisissant la branche basée sur l'apprentissage profond, du domaine d'intelligence artificielle. Et nous avons proposé un modèle hybride, qui est basé sur deux modèles génératifs qui sont :

1. Generative Adversarial Networks "GAN".
2. Variational Auto-Encoder "VAE".

Les principales contributions de ce modèle sont :

1. La génération d'un résumé multi-documentaire à base d'apprentissage profond , alors qu'actuellement on ne peut générer que des headlines.
2. La limitations des architectures récurrentes et l'utilisation de modèles génératifs à la place.
3. La combinaison de deux modèles génératifs.

Nous avons effectué des tests sur notre modèle hybride pour les fonctions d'activation suivantes :

1. La fonction Softmax.
2. La fonction Tanh.

Et pour les fonctions d'erreur suivantes :

1. La fonction d'erreur Binary Cross Entropy.
2. La fonction d'erreur Bayes.
3. La fonction d'erreur Mean Squared Error.

En utilisant le système d'évaluation ROUGE sur le corpus DUC2003. Les expérimentations conduites ont montré que la fonction d'activation Softmax avec la fonction d'erreur Bayes ont donné de meilleurs résultats par rapport aux autres fonctions.

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. Il nous a donné l'occasion 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.
Comme perspectives, nous aimerons pour le futur d'améliorer notre système du résumé comme suit :

1. Donner la possibilité à notre système de filtrer les documents non pertinents, en gardant que les pertinents avant de commencer la tâche du résumé.
2. Améliorer les résumés générés en ajoutant une couche qui permet d'ordonner les phrases d'une façon plus efficace.
3. Améliorer le système en lui donnant la possibilité de choisir les caractéristiques pour l'entrainement selon son besoin, même de pouvoir combiner des caractéristiques pour donner de meilleurs résultats.

Références

Aberdeen, J., Burger, J., Day, D., Hirschman, L., Robinson, P., & Vilain, M. (1995). Mitre: description of the alembic system used for muc-6. In *Proceedings of the 6th conference on message understanding* (pp. 141–155).

Bahdanau, D., Cho, K., & Bengio, Y. (2014). Neural machine translation by jointly learning to align and translate. *arXivpreprint arXiv:1409.0473*.

Bär, D., Zesch, T., & Gurevych, I. (2015). Composing measures for computing text similarity.

Bartakke, D., Sawarkar, S. D., & Gulati, A. (2016). A semantic based approach for abstractive multi-document text summarization. *International Journal of Innovative Research in Computer and Communication Engineeringl*.

Barzilay, R., & McKeown, K. R. (2005). Sentence fusion for multidocument news summarization. *Computational Linguistics*, *31*(3), 297–328.

Barzilay, R., McKeown, K. R., & Elhadad, M. (1999). Information fusion in the context of multi-document summarization. In *Proceedings of the 37th annual meeting of the association for computational linguistics on computational linguistics* (pp. 550–557).

Bengio, Y., Simard, P., & Frasconi, P. (1994). Learning long-term dependencies with gradient descent is difficult. *IEEE transactions on neural networks*, *5*(2), 157–166.

Bossard, A. (2010). *Contribution au résumé automatique multi-documents* (Theses, Université Paris-Nord - Paris XIII). Retrieved from `https://tel.archives-ouvertes.fr/tel-00573567`

Boudin, F. (2008). *Exploration d'approches statistiques pour le résumé automatique de texte* (Unpublished doctoral dissertation). Université d'Avignon.

Boudin, F., Favre, B., Béchet, F., El-Beze, M., Gillard, L., & Torres-Moreno, J.-M. (2007). The lia-thales summarization system at duc-2007. *Actes de DUC*, *7*.

Bouzy, B. (2005). Descente de gradient.

Bowman, S. R., Vilnis, L., Vinyals, O., Dai, A. M., Jozefowicz, R., & Bengio, S. (2015). Generating sentences from a continuous space. *arXivpreprint arXiv:1511.06349*.

Brandow, R., Mitze, K., & Rau, L. F. (1995). Automatic condensation of electronic publications by sentence selection. *Information Processing & Management*, *31*(5), 675–685.

Brants, J. R., Ayoubi, T. A., Chada, K., Marchal, K., de Ven, W. J. V., & Petit, M. M. (2004). Differential regulation of the insulin-like growth factor ii mrna-binding protein genes by architectural transcription factor hmga2. *FEBS Letters*, *569*(1), 277 - 283. Retrieved from `http://www.sciencedirect.com/science/article/pii/S0014579304007239` doi: https://doi.org/10.1016/j.febslet.2004.05.075

Breiman, L., Friedman, J., Olshen, R., & Stone, C. C. (1984). Regression trees. wadsworth int. *Group*.

Carlberger, J., Dalianis, H., Duneld, M., & Knutsson, O. (2001). Improving precision in information retrieval for swedish using stemming. In *Proceedings of the 13th nordic conference of computational linguistics (nodalida 2001)*.

Cheung, J. C. K., Poon, H., & Vanderwende, L. (2013). Probabilistic frame induction. *arXivpreprint arXiv:1302.4813*.

Cho, K., Van Merriënboer, B., Bahdanau, D., & Bengio, Y. (2014). On the properties of neural machine translation: Encoder-decoder approaches. *arXivpreprint arXiv:1409.1259*.

Cho, K., Van Merriënboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., & Bengio, Y. (2014). Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXivpreprint arXiv:1406.1078*.

Chrupała, G., Kádár, A., & Alishahi, A. (2015). Learning language through pictures. *arXivpreprint arXiv:1506.03694*.

Chung, J., Gulcehre, C., Cho, K., & Bengio, Y. (2014). Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXivpreprint arXiv:1412.3555*.

Chung, J., Gulcehre, C., Cho, K., & Bengio, Y. (2015). Gated feedback recurrent neural networks. In *International conference on machine learning* (pp. 2067–2075).

Chung, J., Kastner, K., Dinh, L., Goel, K., Courville, A. C., & Bengio, Y. (2015). A recurrent latent variable model for sequential data. In *Advances in neural information processing systems* (pp. 2980–2988).

Collobert, R., Weston, J., Bottou, L., Karlen, M., Kavukcuoglu, K., & Kuksa, P. (2011). Natural language processing (almost) from scratch. *Journal of Machine Learning Research*, *12*(Aug), 2493–2537.

Crispino, G., & Couto, J. (2004). Construction de résumés automatiques: une approche dynamique.

Damasio, H., Tranel, D., Grabowski, T., Adolphs, R., & Damasio, A. (2004). Neural systems behind word and concept retrieval. *Cognition*, *92*(1-2), 179–229.

Dave, H., & Jaswal, S. (2015). Multiple text document summarization system using hybrid summarization technique. In *Next generation computing technologies (ngct), 2015 1st international conference on* (pp. 804–808).

Delalleau, O., & Bengio, Y. (2011). Shallow vs. deep sum-product networks. In *Advances in neural information processing systems* (pp. 666–674).

Edmundson, H. P. (1969). New methods in automatic extracting. *Journal of the ACM (JACM)*, *16*(2), 264–285.

EI-Ghannam, F., & EI-Shishtawy, T. (2013). Multi-topic multi-document summarizer international journal of computer science & information technology (ijcsit).

Ekmekcioglu, F. C., Lynch, M. F., & Willett, P. (1996). Stemming and n-gram matching for term conflation in turkish texts. *Information Research News*, *7*(1), 2–6.

Fathy, I., Fadl, D., & Aref, M. (2012). Rich semantic representation based approach for text generation. In *Informatics and systems (infos), 2012 8th international conference on* (pp. NLP–20).

Fattah, M. A., & Ren, F. (2009). Ga, mr, ffnn, pnn and gmm based models for automatic text summarization. *Computer Speech & Language*, *23*(1), 126–144.

Favre, B., Bechet, F., Bellot, P., Boudin, F., El-Beze, M., Gillard, L., ... Torres-Moreno, J.-M. (2006). The lia-thales summarization system at duc-

2006. In *Proceedings of document understanding conference (duc-2006), new york, usa.*

Filippova, K. (2010). *Dependency graph based sentence fusion and compression* (Unpublished doctoral dissertation). TechnischeUniversität.

Gatt, A., & Reiter, E. (2009). Simplenlg: A realisation engine for practical applications. In *Proceedings of the 12th european workshop on natural language generation* (pp. 90–93).

Genest, P.-E., & Lapalme, G. (2011). Framework for abstractive summarization using text-to-text generation. In *Proceedings of the workshop on monolingual text-to-text generation* (pp. 64–73).

Genest, P.-E., & Lapalme, G. (2012). Fully abstractive approach to guided summarization. In *Proceedings of the 50th annual meeting of the association for computational linguistics: Short papers-volume 2* (pp. 354–358).

Gers, F., Eck, D., & Schmidhuber, J. (2000). Applying lstm to time series predictable through time-window approaches.

Goodfellow, I., Bengio, Y., Courville, A., & Bengio, Y. (2016). *Deep learning* (Vol. 1). MIT press Cambridge.

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., ... Bengio, Y. (2014). Generative adversarial nets. In Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, & K. Q. Weinberger (Eds.), *Advances in neural information processing systems 27* (pp. 2672–2680). Curran Associates, Inc. Retrieved from `http://papers.nips.cc/paper/5423-generative-adversarial-nets.pdf`

Graves, A. (2013). Generating sequences with recurrent neural networks. *CoRR, abs/1308.0850*. Retrieved from `http://arxiv.org/abs/1308.0850`

Graves, A., & Jaitly, N. (2014). Towards end-to-end speech recognition with recurrent neural networks.

Graves, A., & Schmidhuber, J. (2009). Offline handwriting recognition with multidimensional recurrent neural networks. In *Advances in neural information processing systems* (pp. 545–552).

Greenbacker, C. F. (2011). Towards a framework for abstractive summarization of multimodal documents. In *Proceedings of the acl 2011 student session* (pp. 75–80).

Greengrass, M., Robertson, A. M., Robyn, S., & Willett, P. (1996). Processing morphological variants in searches of latin text. *Information research news, 6*(4), 2–5.

Greff, K., Srivastava, R. K., Koutník, J., Steunebrink, B. R., & Schmidhuber, J. (2015). Lstm: A search space odyssey. *arXivpreprint arXiv:1503.04069*.

Gregor, K., Danihelka, I., Graves, A., Rezende, D. J., & Wierstra, D. (2015). Draw: A recurrent neural network for image generation. *arXivpreprint arXiv:1502.04623*.

Gustavo, C., C.and Javier. (2004).
In *Construction de rsums automatiques: une approche dynamique.*

Hans, M., & Natalis, G. (1980). Word recognition as the basis of a universal method for automatically segmenting texts into sentences. a method for automatic sentence boundary determination in english.

Harabagiu, S. M., & Lacatusu, F. (2002). Generating single and multi-document summaries with gistexter. In *Document understanding conferences* (pp. 11–12).

Hassel, M. (2007). *Resource lean and portable automatic text summarization* (Unpublished doctoral dissertation). KTH.

Håstad, J., & Goldmann, M. (1991). On the power of small-depth threshold circuits. *Computational Complexity*, *1*(2), 113–129.

Heitz, T. (2006). Modélisation du prétraitement des textes. In *Jadt'06 (international conference on statistical analysis of textual data)* (Vol. 1, pp. 499–506).

Hirasawa, K., Ohbayashi, M., Koga, M., & Harada, M. (1996, Jun). Forward propagation universal learning network. In *Neural networks, 1996., ieee international conference on* (Vol. 1, p. 353-358 vol.1). doi: 10.1109/ICNN.1996.548917

Hochreiter, S. (1998). The vanishing gradient problem during learning recurrent neural nets and problem solutions. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, *6*(02), 107–116.

Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. *Neural computation*, *9*(8), 1735–1780.

Indurkhya, N., & Damerau, F. J. (2010). *Handbook of natural language processing* (Vol. 2). CRC Press.

Ishikawa, K., Ando, S., & Okumura, A. (2001). Hybrid text summarization method based on the tf method and the lead method. In *In proceedings of the 2nd national institute of informatics test collection information retrieval (ntcir) workshop* (pp. 5–219).

Jean, S., Cho, K., Memisevic, R., & Bengio, Y. (2014). On using very large target vocabulary for neural machine translation. *arXivpreprint arXiv:1412.2007*.

Jozefowicz, R., Zaremba, W., & Sutskever, I. (2015). An empirical exploration of recurrent network architectures. In *International conference on machine learning* (pp. 2342–2350).

Khan, A., & Salim, N. (2014). A review on abstractive summarization methods. *Journal of Theoretical and Applied Information Technology*, *59*(1), 64–72.

Khan, A., Salim, N., & Kumar, Y. J. (2015a). A framework for multi-document abstractive summarization based on semantic role labelling. *Applied Soft Computing*, *30*, 737–747.

Khan, A., Salim, N., & Kumar, Y. J. (2015b). Genetic semantic graph approach for multi-document abstractive summarization. In *Digital information processing and communications (icdipc), 2015 fifth international conference on* (pp. 173–181).

Kingma, D. P., & Ba, J. (2014). Adam: A method for stochastic optimization. *CoRR*, *abs/1412.6980*. Retrieved from `http://arxiv.org/abs/1412.6980`

Kingma, D. P., & Welling, M. (2013). Auto-encoding variationalbayes. *arXivpreprint arXiv:1312.6114*.

Kiros, R., Salakhutdinov, R., & Zemel, R. S. (2014). Unifying visual-semantic embeddings with multimodal neural language models. *arXivpreprint arXiv:1411.2539*.

Koehn, P. (2004). Pharaoh: a beam search decoder for phrase-based statistical machine translation models. In *Conference of the association for machine translation in the americas* (pp. 115–124).

Kraaij, W., & Pohlmann, R. (1996). Viewing stemming as recall enhancement.

In *Proceedings of the 19th annual international acm sigir conference on research and development in information retrieval* (pp. 40–48).

Kupiec, J., Pedersen, J., & Chen, F. (1995). A trainable document summarizer. In *Proceedings of the 18th annual international acm sigir conference on research and development in information retrieval* (pp. 68–73).

Larkey, L. S., Ballesteros, L., & Connell, M. E. (2002). 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* (pp. 275–282).

Le Cun, Y., & Bengio, Y. (1994). Word-level training of a handwritten word recognizer based on convolutional neural networks. In *Pattern recognition, 1994. vol. 2-conference b: Computer vision & image processing., proceedings of the 12th iapr international. conference on* (Vol. 2, pp. 88–92).

Lee, C.-S., Jian, Z.-W., & Huang, L.-K. (2005). A fuzzy ontology and its application to news summarization. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, *35*(5), 859–880.

Lee, H., Grosse, R., Ranganath, R., & Ng, A. Y. (2009). Convolutional deep belief networks for scalable unsupervised learning of hierarchical representations. In *Proceedings of the 26th annual international conference on machine learning* (pp. 609–616).

Li, P., Lam, W., Bing, L., & Wang, Z. (2017). Deep recurrent generative decoder for abstractive text summarization. *arXivpreprint arXiv:1708.00625*.

Lin, C.-Y. (2004). Rouge: A package for automatic evaluation of summaries. *Text Summarization Branches Out*.

Lloret, E., & Manuel, D. (2011). Analyzing the use of word graphs for abstractive text summarization. *In The First International Conference on Advances in Information Mining and Management*.

Lopyrev, K. (2015). Generating news headlines with recurrent neural networks. *arXiv preprint arXiv:1512.01712*.

Luhn, H. P. (1958). The automatic creation of literature abstracts. *IBM Journal of research and development*, *2*(2), 159–165.

Madnani, N., & Dorr, B. J. (2010). Generating phrasal and sentential paraphrases: A survey of data-driven methods. *Computational Linguistics*, *36*(3), 341–387.

Marc, P. (2006). Réseaux de neurones. *University Laval*, 27–51.

Miao, Y., & Blunsom, P. (2016). Language as a latent variable: Discrete generative models for sentence compression. *CoRR*, *abs/1609.07317*. Retrieved from `http://arxiv.org/abs/1609.07317`

Mikheev, A. (2002). Periods, capitalized words, etc. *Computational Linguistics*, *28*(3), 289–318.

Mnasri, M. (2015). Résumé automatique multi-document dynamique: État de lart.

Moawad, I. F., & Aref, M. (2012). Semantic graph reduction approach for abstractive text summarization. In *Computer engineering & systems (icces), 2012 seventh international conference on* (pp. 132–138).

Monz, C., & De Rijke, M. (2001). Shallow morphological analysis in monolingual information retrieval for dutch, german, and italian. In *Workshop of the cross-language evaluation forum for european languages* (pp. 262–277).

Moratanch, N., & Chitrakala, S. (2016). A survey on abstractive text sum-

marization. In *Circuit, power and computing technologies (iccpct), 2016 international conference on* (pp. 1–7).

Moulinier, I., McCulloh, J. A., & Lund, E. (2000). West group at clef 2000: Non-english monolingual retrieval. In *Workshop of the cross-language evaluation forum for european languages* (pp. 253–260).

Nallapati, R., Zhou, B., Gulcehre, C., Xiang, B., et al. (2016). Abstractive text summarization using sequence-to-sequence rnns and beyond. *arXivpreprint arXiv:1602.06023*.

Napoles, C., Gormley, M., & Van Durme, B. (2012). Annotated gigaword. In *Proceedings of the joint workshop on automatic knowledge base construction and web-scale knowledge extraction* (pp. 95–100).

Nenkova, A., & McKeown, K. (2012). A survey of text summarization techniques. In *Mining text data* (pp. 43–76). Springer.

Neto, J. L., Freitas, A. A., & Kaestner, C. A. (2002). Automatic text summarization using a machine learning approach. In *Brazilian symposium on artificial intelligence* (pp. 205–215).

Nobata, C., & Sekine, S. (2004). Crl/nyu summarization system at duc-2004. In *Proceedings of duc*.

Nunberg, G. (1990). The linguistics of punctuation (center for the study of language and information-lecture notes).

Paice, C. D. (1981). 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* (pp. 172–191). Kent, UK, UK: Butterworth & Co. Retrieved from `http://dl.acm.org/citation.cfm?id=636669.636680`

Palmer, D. D. (2010). Text preprocessing, handbook of natural language processing, machine learning and pattern recognition. *CRC Press, Taylor and Francis Group, Boca Raton, FL, deuxime dition*.

Palmer, D. D., & Hearst, M. A. (1997). Adaptive multilingual sentence boundary disambiguation. *Computational linguistics*, *23*(2), 241–267.

Parker, R., Graff, D., Kong, J., Chen, K., & Maeda, K. (2011). Linguistic data consortium.

Passonneau, R. J., Chen, E., Guo, W., & Perin, D. (2013). Automated pyramid scoring of summaries using distributional semantics. In *Proceedings of the 51st annual meeting of the association for computational linguistics (volume 2: Short papers)* (Vol. 2, pp. 143–147).

Popovic, M., & Willett, P. (1992). The effectiveness of stemming for natural-language access to slovene textual data. *Journal of the American Society for Information Science*, *43*(5), 384.

Porter, M. (1997). *Readings in information retrieval. chap. an algorithm for suffix stripping (pp. 313–316)*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc.

Reynar, J. C., & Ratnaparkhi, A. (1997). A maximum entropy approach to identifying sentence boundaries. In *Proceedings of the fifth conference on applied natural language processing* (pp. 16–19).

Rezende, D. J., Mohamed, S., & Wierstra, D. (2014). Stochastic backpropagation and approximate inference in deep generative models. *arXivpreprint arXiv:1401.4082*.

Riley, M. D. (1989). Some applications of tree-based modelling to speech and

language. In *Proceedings of the workshop on speech and natural language* (pp. 339–352).

Rumelhart, D. E., Hinton, G. E., & Williams, R. J. (1986). Learning representations by back-propagating errors. *nature*, *323*(6088), 533.

Rush, A. M., Chopra, S., & Weston, J. (2015). A neural attention model for abstractive sentence summarization. *arXivpreprint arXiv:1509.00685*.

Saggion, H., & Lapalme, G. (2002). Generating indicative-informative summaries with sumum. *Computational linguistics*, *28*(4), 497–526.

Salton, G., & Buckley, C. (1988). Term-weighting approaches in automatic text retrieval. *Information processing & management*, *24*(5), 513–523.

Salton, G., Singhal, A., Buckley, C., & Mitra, M. (1996). Automatic text decomposition using text segments and text themes. In *Proceedings of the the seventh acm conference on hypertext* (pp. 53–65).

SN, V. V., & Bressan, S. (2001). Indexing the indonesian web: Language identification and miscellaneous issues. *Recall*, *80*, 100.

Sparck Jones, K. (1972). A statistical interpretation of term specificity and its application in retrieval. *Journal of documentation*, *28*(1), 11–21.

Suanmali, L., Salim, N., & Binwahlan, M. S. (2009). Fuzzy logic based method for improving text summarization. *arXivpreprint arXiv:0906.4690*.

Sutskever, I., Vinyals, O., & Le, Q. V. (2014). Sequence to sequence learning with neural networks. In *Advances in neural information processing systems* (pp. 3104–3112).

Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., ... others (2015). Going deeper with convolutions..

Tai, S. Y., Ong, C. S., & Abullah, N. A. (2000). 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* (pp. 207–208).

Tanaka, H., Kinoshita, A., Kobayakawa, T., Kumano, T., & Kato, N. (2009). Syntax-driven sentence revision for broadcast news summarization. In *Proceedings of the 2009 workshop on language generation and summarisation* (pp. 39–47).

Taylor, G. W., Fergus, R., LeCun, Y., & Bregler, C. (2010). Convolutional learning of spatio-temporal features. In *European conference on computer vision* (pp. 140–153).

Touzet, C. (1992). *Les réseaux de neurones artificiels, introduction au connexionnisme.* EC2.

Trandabă, D. (2011). Using semantic roles to improve summaries. In *The 13th european workshop on natural language generation* (p. 164).

Vinyals, O., Kaiser, Ł., Koo, T., Petrov, S., Sutskever, I., & Hinton, G. (2015). Grammar as a foreign language. In *Advances in neural information processing systems* (pp. 2773–2781).

Vinyals, O., Toshev, A., Bengio, S., & Erhan, D. (2015). Show and tell: A neural image caption generator. In *Computer vision and pattern recognition (cvpr), 2015 ieee conference on* (pp. 3156–3164).

Xavier, G. (2014).
In *pprentissage des rseaux de neurones profonds et applications en traitement automatique de la langue naturelle.*

Xu, K., Ba, J. L., Kiros, R., Cho, K., Courville, A., Salakhutdinov, R., ...

Bengio, Y. (2015). Show, attend and tell: Neural image caption generation with visual attention.

Yeasmin, S., Tumpa, P. B., Nitu, A., & Palash, M. (2017). Study of abstractive text summarization techniques.