Université Paris Dauphine

Master 2 IASD
Intelligence Artificielle, Systèmes, Données

Rapport de Mémoire

Extraction automatique du sens de documents textuels

Année universitaire 2021-2022

Rédigé par Mathilde DA CRUZ et encadré par

Monsieur Maixent CHENEBAUX, tuteur enseignant.

Les données textuelles sont de plus en plus abondantes et vouloir transmettre les informations d'un document est une tâche très commune, dans de nombreux métiers. Afin d'assister automatiquement et faciliter cette tâche, nous proposons d'extraire de chaque document trois éléments : un résumé, des mots clés et un thème. À cette fin, nous étudions et adaptons à notre contexte - c'est à dire des données de quelques pages, en français, avec du vocabulaire technique, et pas de supervision - différentes méthodes de synthèse extractive et d'extraction de mots clés et thème, notamment en utilisant des graphes. Nous discutons également des méthodes d'évaluation de la synthèse automatique, et proposons des améliorations, notamment pour évaluer la pertinence et la consistance d'un résumé automatique.

Mémoire rédigé lors d'une année en apprentissage au
Secrétariat Général des Ministères Économiques et Financiers
encadrée par Monsieur Quentin DESROUSSEAUX.

Remerciements

Je tiens à remercier toutes les personnes qui m'ont apporté leur soutien, leurs idées et conseils pour l'élaboration et la rédaction de ce mémoire.

Je remercie tout d'abord Monsieur Maixent Chenebaux, mon tuteur enseignant, pour l'encadrement de ce travail, et pour les réponses qu'il a pu m'apporter. Je le remercie également pour avoir fait grandir mon intérêt pour la théorie des graphes, en particulier pour ses applications au domaine du *NLP*.

Je remercie aussi Monsieur Quentin Desrousseaux, mon tuteur au Bercy Hub lors de cette enrichissante année d'apprentissage au Ministère de l'Économie, pour la proposition d'un sujet aussi intéréssant, ainsi que pour sa disponibilité et ses conseils avisés durant toute cette année.

Bien sûr, je remercie toute l'équipe pédagogique du master IASD de l'université Paris Dauphine, en particulier Monsieur Tristan Cazenave, responsable de formation, et Madame Patricia Lavagna, responsable du CFA.

Enfin, je remercie mes amis pour leur relecture patiente, leur soutien, et leurs encouragements.

Table des matières

Première partie

Introduction

1 Contexte

Dans de nombreux domaines, et notamment dans l'administration publique, les données textuelles sont abondantes. L'utilisation d'un document textuel consiste habituellement en sa lecture par une personne afin d'en comprendre le sens et, enfin, d'utiliser ce sens pour diverses tâches. Une tâche classique est la transmission d'informations : il s'agit de faire comprendre à une autre personne le sens d'un ou plusieurs documents, mais de manière beaucoup plus rapide et efficace qu'une simple lecture complète du ou des documents. Évidemment, afin d'effectuer cette tâche, une personne doit déjà avoir elle même saisit le sens des documents. Néanmoins, l'exercice de la transmission de l'information issue de données textuelles ne nécessite ici aucune analyse du texte : l'objectif est de rapporter de la manière la plus objective, claire et concise possible les informations contenues dans le ou les textes.

Selon la quantité et la longueur des documents, il peut être fastidieux pour une personne d'effectuer cette tâche. Nous pourrions alors tenter d'automatiser complètement cette tâche, c'est-à-dire :

— (1) À partir d'un corpus de documents, produire un ou plusieurs documents qui permettent une appréhension rapide et efficace du sens du corpus de documents.

ou bien de faciliter cette tâche pour une personne, c'est à dire :

— (2) À partir d'un corpus de documents, produire un ou plusieurs documents qui facilitent à une personne l'étude et la compréhension du corpus, dans le but d'en transmettre l'information.

Ces deux solutions sont similaires mais néanmoins différentes. La solution (1) aurait pour but de totalement remplacer une personne et ainsi de ne garder que les informations essentielles à transmettre. Cette solution permet une compréhension plus superficielle que la suivante (ou alors nécéssiterait de rapporter une grande quantité d'informations, ce qui donnerait finalement presque autant de travail que si la personne n'avait pas d'outil à sa disposition). La solution (2) a pour objectif de permettre une compréhension un peu plus en détail des documents, afin que la personne qui effectuera la tâche de la transmission d'informations puisse, par exemple, créer un document de synthèse ou aussi bien effectuer une présentation à d'autres personnes et répondre à des questions sur le corpus.

Nous pouvons nous demander ce qui faciliterait pour une personne la tâche de transmission d'informations. Nous avons posé la question suivante à plusieurs personnes de notre entourage professionel et personnel :

"Vous disposez d'une cinquantaine de textes de deux pages environ. Vous souhaitez transmettre l'information de ces textes, c'est à dire faire comprendre le sens de ces documents à une autre personne de la manière la plus rapide et efficace possible. Vous devrez également être capable de répondre à des questions sur le sens global de ces textes. Qu'est ce qui pourrait vous permettre de vous préparer le plus rapidement et efficacement possible à cette tâche, et sans avoir à lire tous les documents dans le détail ?"

Les réponses qui revenaient le plus souvent étaient :

- Un résumé de chaque texte ;
- Un résumé global ;
- Des mots clés pour chaque texte ;
- Des mots clés globaux ;
- Des aides de lecture (des indications pour repérer les parties importantes d'un texte) ;
- Les thèmes de chaque texte.

Nous nous intéresserons ici à la seconde solution. En effet, cette problématique nous a été introduite par le bureau du dialogue social du ministère de l'Économie, dont un

des rôles principaux est d'assurer la communication entre les syndicats et la secrétaire générale. Ce bureau effectue chaque jour une veille sociale, en collectant tous les nouveaux tracts parus, les lisant, et les synthétisant. Les personnes du bureau du dialogue social doivent pouvoir effectuer diverses tâches telles que :

— Synthétiser un tract ;
— Synthétiser un ensemble de tracts sur une certaine durée ;
— Synthétiser un ensemble de tracts selon un certain sujet ;
— Faire une présentation sur un sujet et éventuellement répondre à des questions ;
— Participer à des groupes de travail sur le sujet des tracts et leur contenu.

Nous allons donc chercher à faciliter la tâche de transmission d'informations dans le but d'aider une personne à effectuer les tâches ci-dessus. Dans une volonté de précision, nous traiterons cette tâche par document (et non pas pour un corpus dans sa globalité). Il est très important de ne pas transmettre de mauvaises informations, ou des informations incomplètes, c'est également pourquoi nous préfèrerons la solution (2).

2 Problématique et motivations

Notre problème sera de réussir à extraire du sens d'un corpus de documents dans le but d'aider une personne à transmettre de l'information concernant ces documents mais en lui permettant de s'adapter à différentes granularités de précision. Nous nous concentrerons principalement sur des documents de quelques pages (une dizaine tout au plus).

Ce mémoire aura pour objectif de répondre aux problématiques suivantes :

— **Par quel moyen et sous quelle forme extraire le sens de documents ?**

— **Comment évaluer une synthèse de document ?**

Le problème de la synthèse de documents, ou plus généralement de la transmission d'informations et de sens issus d'un document est un problème très classique et récurrent dans le milieu professionnel. Il s'agit en effet d'une tâche effectuée chaque jour par

de nombreuses personnes. Afin d'automatiser complètement cette tâche, il nous faudrait développer un modèle non seulement capable de synthétiser efficacement, mais également de saisir différents enjeux liés à un domaine en particulier mais aussi de s'adapter à plusieurs situations, plusieurs granularités, et capable de répondre à des questions sur ces sujets spécifiques. Cette tâche semble possible à réaliser et serait également intéréssante à étudier, mais nous avons ici décidé de laisser celle-ci à l'expertise humaine des agents du ministère et de ne nous concentrer pour le moment que sur l'aide à la transmission de sens.

Notons que, bien que la synthèse automatique de texte représente une grosse partie de la tâche, nous parlerons ici bien d'extraction de sens de documents. En effet, en plus de la synthèse automatique, nous tâcherons de trouver d'autre méthodes pour extraire du sens d'un texte. Nous nous intéresserons notamment, en plus de la synthèse automatique, à l'extraction de mots clés, et à l'attribution de thèmes.

Plus concrètement, à partir d'une base de données constituée de milliers de documents de chacun une à quelques pages (dans notre cas d'application il s'agira de tracts) nous allons réfléchir à des solutions pour aider des personnes à produire efficacement des synthèses, faire des compte-rendus, ou participer à des groupes de travail. Il conviendra donc de réfléchir à la forme que prendront les résultats en plus de la méthode pour les obtenir, mais également à une manière de les évaluer. Dans ce mémoire, nous essaierons tout particulièrement de prendre en compte les considérations pratiques d'un tel problème d'intelligence artificielle. En effet, nous tâcherons de discuter d'aspects qui passent souvent après les questions de performance (produire un modèle *state of the art*[1], avoir la meilleure *accuracy*, etc.) mais qui ont pourtant une grande importante, particulièrement lors d'une tâche aussi subjective que la nôtre.

Cette problématique est clairement en lien avec une de mes missions en tant qu'apprentie au ministère. J'ai choisi ce sujet précis pour mon mémoire pour diverses raisons. Tout d'abord, il s'agit de traiter une tâche rencontrée très régulièrement, autant dans le monde professionnel que personnel. Il m'a donc semblé particulièrement motivant de mener des recherches pour un problème que je rencontre souvent dans ma vie, et que je

1. Un modèle *state of the art* est supposé être le meilleur actuellement. Mais l'évaluation des modèles proposés dans les papiers concerne un jeu de données d'entrainement en particulier et une méthode d'évaluation en particulier.

rencontrerai encore de nombreuses fois. Ensuite, il m'a semblé qu'il n'existait pas pour ce problème une unique solution universelle bien meilleure que toutes les autres, dans toutes les situations. J'ai donc trouvé qu'il serait intéréssant d'explorer ce domaine et de mieux comprendre les diverses possibilités afin de trouver une solution personelle la plus adaptée possible à un problème en particulier, et de débattre de la subjectivité des résultats. Enfin, j'ai un intérêt personnel particulier pour le domaine du *NLP* (*Natural Language Processing*, ou traitement du language naturel) et explorer un problème de synthèse et de transmission d'informations me permet également de me pencher sur de nombreux autres sujets classiques du *NLP*, tels que les *embeddings*[2] de mots.

Par ailleurs, la synthèse de texte et la transmission efficace des informations me semblent être des sujets particulièrement importants de nos jours. En effet, avec la baisse des coûts de production et de stockage (et également la digitalisation de documents papiers), la quantité de textes disponibles en ligne augmente de jour en jour. La synthèse devient alors presque indispensable pour gérer cette immense quantité de textes et tenter de tirer parti de toute cette information, alors qu'il est impossible - notamment par manque de temps - de se tenir informé de toutes les informations publiées chaque jour (même concernant seulement un domaine particulier). Aussi, dans un monde où les réseaux sociaux ont une place prépondérante, la désinformation et les *fake news*[3] sont une menace permanente. Personne n'a le temps d'apprendre toutes les complexités de notre monde et de devenir expert sur tous les sujets. C'est ainsi que nous pouvons tous en venir un jour à croire un gros titre racolleur ("*clickbait*") ou de fausses informations. Dans ce cadre, la synthèse automatique permettrait de gagner du temps et ainsi saisir les informations principales d'un article au titre sensationaliste, afin de se faire notre propre avis dessus, avant de le repartager. Aussi, l'extraction de sens (de manière plus générale que seulement la synthèse automatique), permettrait de simplifier nos recherches - également pour aider à sélectionner ce que l'on souhaitera lire - pour se tenir informés sur nos sujets d'intérêt, ou pour vérifier la crédibilité de faits lus (notamment sur les réseaux). L'objectif serait de rendre une grande quantité d'informations beaucoup plus digeste mais avec le moins possible de perte d'informations.

On peut penser deux types de synthèses : La synthèse manuelle et la synthèse automa-

2. Plongement, ou vectorisation
3. Fausse information ou désinformation.

tique [30]. De manière générale, pour synthétiser un document, une personne devra lire le document dans son entièreté, se concentrer sur les phrases et éléments importants, et enfin, réécrire ces morceaux en un résumé cohérent. La synthèse manuelle est faite par un expert et est un procédé long et compliqué pour un humain (et donc cher à appliquer à un usage à grande échelle). Il semble donc tout à fait pertinent de souhaiter automatiser cette tâche, et c'est dans ce cas que nous parlons de synthèse automatique. L'objectif étant de rendre cette tâche moins chère et plus rapide (et pourquoi pas plus efficace !). Par ailleurs, l'automatisation d'une telle tâche pourrait également avoir la vertue de produire un résumé moins subjectif et moins biaisé du texte.

Deuxième partie

État de l'art

Dans cette partie, nous présenterons un état de l'art des solutions qui ont déjà pu être apportées à notre problème. La tâche que nous avons appelée "extraction du sens" n'est pas vraiment une tâche classique de *NLP* (ou tout du moins cette tâche est rarement appelée ainsi, mais on parle souvent seulement de synthèse automatique), mais il est possible de la décomposer en plusieurs sous-tâches classiques, afin d'en étudier les différentes méthodes qui peuvent permettre de résoudre notre problème. Afin d'extraire le sens d'un texte ou d'un corpus de texte, avant de chercher une solution personnalisée, nous pouvons penser notamment à la synthèse automatique, méthode qui aura effectivement une place centrale dans notre travail.

3 Présentation et débuts de la synthèse automatique

La synthèse automatique de texte, consiste à produire automatiquement un texte, plus court que le texte original, qui contiendra les informations les plus importantes de ce dernier. Il s'agit de créer un résumé qui permettrait aux personnes le lisant de comprendre l'essentiel du texte sans le lire. Il existe principalement deux méthodes de synthèse automatique : La synthèse extractive et la synthèse abstractive. Nous développerons ces deux méthodes. Enfin, il est possible de faire de la synthèse multi-documents ou de la synthèse mono-document. Nous nous concentrerons sur le cas mono-document, mais la synthèse multi-document pourrait être une extension de notre travail.

Dans la majorité des cas pratiques, la synthèse automatique est non-supervisée, c'est à dire qui n'utilise pas d'exemples déjà labellisés, et en l'occurence, qui n'utilise pas d'exemples de textes résumés pour apprendre à faire de même. En effet, comme évoqué

précédemment, synthétiser un texte est une tâche fastidieuse, très subjective et difficile même pour des humains. Bien qu'il existe de petites bases de textes avec leurs résumés, il serait beaucoup trop cher de faire labelliser une grande quantité de textes. Nous nous concentrerons ici sur les méthodes non supervisées. Les méthodes supervisées sont plus utilisées sur de courts textes et de très courts résumés, pour lesquels on dispose déjà de données labellisées facilement, par exemple pour des commentaires suite à des achats avec comme cible le titre du commentaire.

La synthèse automatique reste un sujet complexe, pour lequel il n'existe pas de solution miracle. En effet, certaines methodes sont *state of the art* sur certains jeux de données d'entrainement mais pas sur d'autres. Les performances d'un modèle peuvent dépendre notamment de la complexité du vocabulaire utilisé ou tout simplement de la taille des textes. Pour comparer différent modèles, la méthode d'évaluation joue également beaucoup. Un modèle peut être meilleur selon une certaine métrique mais pas selon une autre. Il est donc important de définir une métrique d'évaluation qui mesure efficacement les performances d'un modèle.

C'est dès la fin des années 1950 (on situe souvent le début du *NLP* dans les années 1940) que la recherche pour la synthèse automatique a commencé à attirer l'attention de la communauté scientifique [52]. Un des plus vieux papiers parus à ce sujet [36] présentait une méthode pour extraire les phrases les plus importantes d'un texte en utilisant des caractéristiques statistiques telles que la fréquence des mots, et ainsi construire ce que Luhn appelait "*auto-abstract*". La proposition consistait à attribuer aux phrases un poids en fonction de la fréquence des mots qu'elle contient, tout en ignorant les mots avec une trop grande fréquence, qu'on appelle couramment les *stop-words*[4] aujourd'hui. La Figure 1 montre cette approche concernant l'importance des mots selon leur fréquence. Cette approche est aujourd'hui largement utilisée dans le *NLP*, et de nombreuses méthodes actuelles de synthèse automatique reposent sur ces travaux.

En 1969, Harold Edmundson [13] définit trois méthodes pour repérer les phrases importantes dans un texte.

4. Appelés également mots vides en français. Il s'agit des mots qui ne contiennent que peu de sens, et servent principalement à la syntaxe.

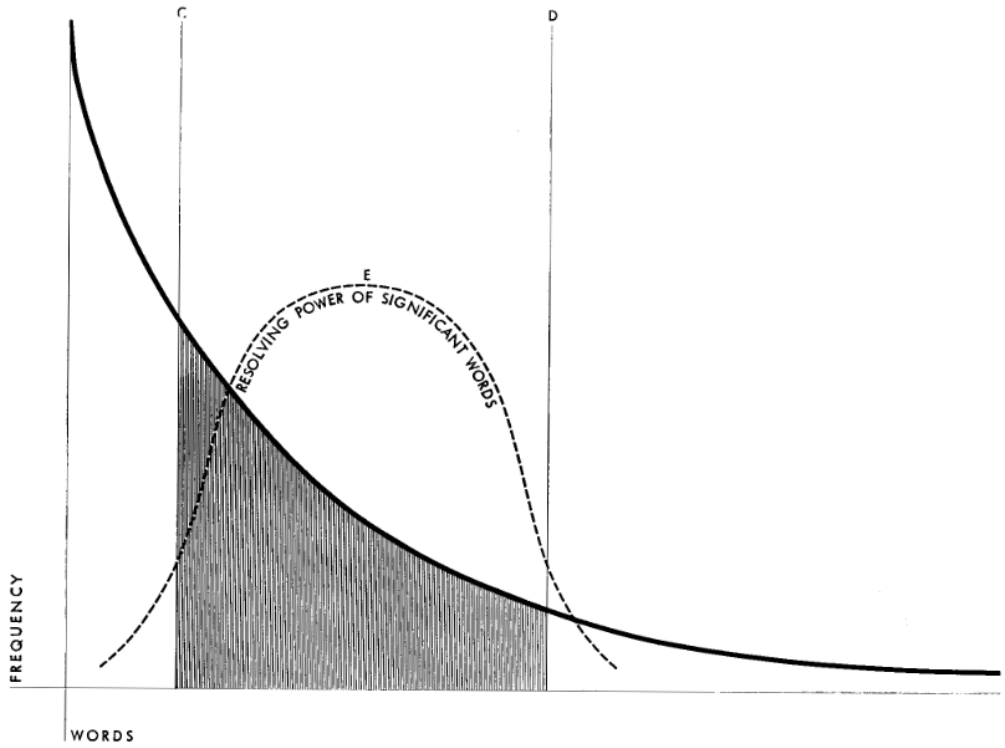


FIGURE 1 – Diagramme de l'importance des mots selon leur fréquence.[36] En abscisse, les mots sont triés par fréquence. Les mots trop fréquents sont des *stopwords*. Les mots les plus importants sont ceux avec une fréquence intermédiaire.

— "*Cue Method*" : L'importance d'une phrase est calculée selon l'absence ou la présence de certains mots indices qui indiqueraient que la suite est importante, tels que "on en déduit que", "en résumé", "il est important de noter que" ;
— "*Title Method*" : Le poids d'une phrase est calculé comme la somme de tous ses mots qui apparaissent également dans des titres ou sous titres du textes ;
— "*Location Method*" : Cette méthode suppose que les phrases apparaissant au début du document ou au début de paragraphes ont une plus haute probabilité d'être importantes.

Il s'agit de méthodes assez primaires mais pouvant néanmoins être couplées à d'autres méthodes ou a du *machine learning*.

Plus tard, en 1984, Elaine Marsch décrit la "*Production rule system for summarization*" [5] [37], qui consiste en trois étapes : "(i) inférence (déduire des liens entre des termes/phrases, ou des indicateurs d'importance), (ii) attribution de scores selon l'importance, (iii) sélection des meilleures phrases comme résumé". [6] Dans de nombreuses

5. Règle de production pour la synthèse.
6. *"(i) inferencing, (ii) scoring the format rows for their importance and finally (iii) selecting the*

méthodes de synthèse extractive, on observe des étapes très similaires. L'intéret pour la synthèse automatique a été renforcé par la suite avec le développement de l'intelligence artificielle.

Pendant longtemps, les recherches ont été concentrées sur la synthèse extractive. Dans les dernières années, on ne note pas d'avancées majeures dans ce domaine ou de nouvelles techniques revolutionnaires, mais plutôt de petites améliorations ou des combinaisons de méthodes existantes. Récemment, les nouvelles découvertes concernent plutôt la synthèse abstractive, en particulier depuis l'emergence du *deep learning*. Jusqu'à récemment, on pensait d'ailleurs impossible de pouvoir automatiser la tâche de synthèse de texte comme le ferait un humain. Les avancées récentes concernant les *word embeddings*, les réseaux de neurones récurrents, ou encore les *Transformers* ont rendu cet objectif bien plus proche.

4 La synthèse extractive

Le principe de la synthèse extractive est de produire un résumé mais en sélectionnant les phrases les plus importantes d'un texte ou d'un corpus de texte, sans les modifier. Très souvent, il est question d'élaborer un classement des phrases d'un texte, de la plus importante à la moins importante, grâce à un score qui aura été attribué à chaque phrase, et de ne garder que les n premières. Une phrase sera considérée comme importante si elle porte une quantité importante de l'information d'un texte. Le problème peut aussi être vu comme un problème de classification : est ce qu'une phrase doit apparaître dans le résumé ou non ? Ce problème est parfois également appelé "extraction de phrases clés".

Souvent, le processus de synthèse extractive suit les étapes de la figure 2.

Les recherches sur la synthèse extractive ayant débuté dans les années 1950, et au vu de l'importance du sujet, une gigantesque quantité de papiers a été publiée à ce sujet. De très nombreuses méthodes existent donc, mais en réalité, la plupart s'appuient sur les mêmes concepts fondamentaux et ne proposent que de petites modifications, ou

appropriate ones as summary."

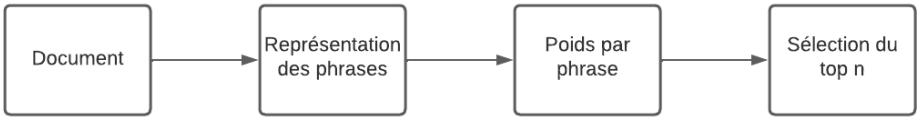


FIGURE 2 – Etapes classiques de la synthèse extractive.

des combinaisons de méthodes. Dans cette section, nous tâcherons de présenter les méthodes principales.

4.1 Représentation des phrases et attribution des poids

La synthèse extractive commence souvent par une phase de représentation des phrases. Les méthodes les plus classiques utilisent la fréquence des mots.

4.1.1 *Topic Words*

La technique des "*Topic Words*" est celle décrite dans le papier de Luhn [36] que nous avons présenté comme le premier papier concernant la synthèse automatique. Un peu plus tard d'autres méthodes modifiant légèrement cette première idée à l'aide d'un ratio de *log-vraissemblance* sont parues [11].

Une fois les "*Topic Words*" identifiés, l'importance d'une phrase pourra être mesurée ou bien par le nombre qu'elle en contient (mais cette méthode peut favoriser les longues phrases) ou par la proportion qu'elle en contient.

4.1.2 Probabilités de mots

Une autre manière simple serait d'utiliser la probabilité des mots. La probabilité d'un mot t correspond au nombre d'occurences de ce mot divisé par le nombre de mots du document [57]. Une phrase est alors représentée par le vecteurs des probabilités de mots. La papier propose l'algorithme de la figure 3.

Cet algorithme assure de toujours choisir une phrase avec le mot qui aura la plus

Step 1 Compute the probability distribution over the words w_i appearing in the input, $p(w_i)$ for every i; $p(w_i) = \frac{n}{N}$, where n is the number of times the word appeared in the input, and N is the total number of content word tokens in the input.

Step 2 For each sentence S_j in the input, assign a weight equal to the average probability of the words in the sentence, i.e.,

$$Weight(S_j) = \sum_{w_i \in S_j} \frac{p(w_i)}{|\{w_i | w_i \in S_j\}|}$$

Step 3 Pick the best scoring sentence that contains the highest probability word.

Step 4 For each word w_i in the sentence chosen at step 3, update their probability:

$$p_{new}(w_i) = p_{old}(w_i) \cdot p_{old}(w_i)$$

Step 5 If the desired summary length has not been reached, go back to Step 2

FIGURE 3 – Agorithme SumBasic de synthèse extractive de [57].

haute probabilité grâce à **l'étape 3**. Les poids sont ensuite mis à jour pour tenter de diversifier le résumé. D'autres méthodes se basent sur ce principe mais utilisent une fonction à optimiser pour maximiser l'apparition des mots importants dans le résumé [56].

4.1.3 *TF-IDF*

Les méthodes précédentes donneront de hauts scores aux mots très fréquents, c'est pourquoi il est important de préciser une liste de *stopwords*. La méthode suivante palie ce problème en pénalisant automatiquement les mots qui apparaissent trop souvent dans tout le corpus. Il s'agit d'une méthode qui utilise le score *TF-IDF* (*Term Frequency Inverse Document Frequency*)[53]. Ce score mesure l'importance d'un mot relativement à un document (on parlera de manière générale de document mais il peut également s'agir d'une phrase). En effet, un mot est considéré comme important s'il apparaît relativement souvent dans le document, et relativement peu dans le corpus. Un mot apparaissant trop fréquemment dans le corpus serait considéré comme un *stopword*, et donc peu important, comme nous avons pu le voir dans la figure 1. Le score *TF-IDF* est calculé comme suit :

$$TF\text{-}IDF(t) = TF(t, d) \times IDF(t) \tag{1}$$

t étant un mot et d un document du corpus. TF est la fréquence du mot par rapport

au document, et *IDF* est la fréquence inverse du mot par rapport au corpus. On a donc $TF(t, d)$ = Nombre d'occurences de t dans d / Nombre de mots dans d, et $IDF(t)$ = Nombre de documents dans le corpus (N) / Nombre de documents où t apparaît (df). On note que TF correspond à la proabilité du terme définie dans la partie précédente. En pratique, pour éviter que la valeur d'IDF n'explose pour de grands corpus, et pour éviter de d'avoir un zéro au dénominateur lorsque le mot n'apparaît pas, on utilisera plutôt $IDF = log(N/(df + 1))$. Ce score discrimine naturellement les *stopwords*, puisque les mots apparaissant très souvent, voire dans presque tous les textes, tels que "et" ou "être" auront un faible IDF et donc un faible *TF-IDF*.

Les poids *TF-IDF* ont le gros avantage d'être rapides et faciles à calculer, en plus de donner de bons résultats. Ce score est utilisé dans de nombreuses techniques et reste un incontournable du *NLP*.

On peut calculer une représentation du document grâce au score *TF-IDF* dans une matrice de taille (Nombre de phrases × Nombre de mots du vocablaire). Les scores *TF-IDF* des mots du vocabulaire pour une phrase représenteront les caractéristiques de la phrase. Autrement dit, une phrase est représentée par un vecteur de poids de mots. À partir de cela, il est possible d'utiliser divers algorithmes pour sélectionner les phrases du résumé. Une possibilité est, comme avec les *topic words*, de prendre les phrases avec le meilleur score cumulé, ou bien avec le meilleur score cumulé divisé par le nombre de mots de la phrase. Il est également possible d'utiliser un algorithme de *clustering* en utilisant les phrases comme *items* à classer[26]. Chaque *cluster* représenterait alors un thème du texte, et il suffirait ensuite de sélectionner la phrase la plus proche de chaque centroïde. Cela permet de diversifier les sujets du résumé. Cependant, selon la méthode de *clustering* utilisée, il peut être nécéssaire de devoir préciser un nombre de *clusters*, ce qui est souvent un inconvénient. Par ailleurs, cette méthode présente également l'inconvénient de ne sélectionner qu'une phrase par thème, sans hiérarchiser les thèmes. Si un document comporte un thème principal, et quelques thèmes annexes peu importants, l'algorithme ressortira une phrase pour chaque thème, y compris les peu importants, au détriment du thème principal qui aurait peut-être mérité plus d'une phrase pour le décrire. Une solution pourrait être d'adapter le nombre de *cluster* (le thème principal serait ainsi séparé en plusieurs sous-thèmes), ou bien de pondérer le

nombre de phrases à sélectionner par la taille du *cluster*.

Une autre méthode répandue qui utilise cette matrice de *features TF-IDF*, est la "*Latent Semantic Analysis*"[24]. Cela consiste en l'utilisation de la décomposition en valeurs singulières de la matrice *TF-IDF* pour extraire trois matrices. La première est la matrice termes/thèmes, la seconde, la matrice diagonale, est la matrice des thèmes et leurs poids, et la dernière la matrice des phrases et leurs thèmes. À partir de là, il est possible d'extraire les poids des thèmes dans chaque phrase ou encore les mots qui représentent le mieux un thème. Il est alors possible de sélectionner une phrase par thème. Il existe plusieurs variations sur cette méthode [44] [45]. Cette méthode présente le même inconvénient que la méthode par *cluster*, c'est à dire que des thèmes peu importants seront placés au même niveau qu'un thème important, et on en sélectionnera donc le même nombre de phrases.

Les méthodes basées sur la fréquence ont l'inconvénient de ne pas prendre en compte les synonymes et périphrases. Ainsi, un sujet souvent répété, mais de manières différentes, ne sera pas forcément considéré comme important.

4.2 Sélection des phrases

Méthode classique Une fois que les phrases ont des poids, il convient de choisir les phrases à garder. Une manière très répandue, sans doute la plus simple et intuitive, est de simplement garder les n phrases avec le meilleur score (n étant la taille du résumé désirée). Nous avons également vu plus haut une méthode de sélection grâce au *clustering*. Il est également possible de sélectionner les phrases du top jusqu'à avoir dépassé un certain nombre de mots.

Sélection des phrases par *Maximal Margin Relevance* Une autre possibilité est d'utiliser l'algorithme de *Maximal Marginal Relevance* (*MMR*) [8] pour éviter les redondances. Le principe est de parcourir les unités textuelles pré-sélectionnées (par exemple les n premières), et de construire le résumé de telle sorte que chaque ajout soit le plus important, mais aussi le plus éloigné des informations déjà présentes. Cependant, cet algorithme ne remet pas en question les choix déjà effectués : une fois qu'une phrase

a été sélectionnée, elle ne sera jamais remplacée par une autre, qui pourrait être plus complète ou pourrait fusionner deux idées.

Méthodes par Optimisation Linéaire Enfin, il est possible de voir la construction du résumé comme un problème d'optimisation linéaire [23]. L'idée est de maximiser la présence des concepts du texte dans les phrases du résumé en ajoutant des contraintes, notamment de longueur du texte.

Pour finir, on pourra organiser les phrases par ordre d'importance ou par ordre d'apparition dans le texte (c'est souvent cette méthode qui est préférée, dans un souci de lisibilité).

4.3 Méthodes par graphe

Une toute autre manière de voir le problème de la synthèse automatique est d'utiliser les graphes. Une première méthode assez répandue et que nous développerons plus tard dans notre travail est appelée *TextRank* [40]. Cette méthode se base sur l'algorithme *Pagerank* de Google [46], qui peut ressortir les pages *web* les plus importantes d'un réseau de pages. L'idée est simplement de faire la même chose avec des phrases, le réseau de pages étant alors tout simplement le texte. Pour construire un graphe à partir d'un texte, on considère que l'ensemble des sommets est l'ensemble des phrases et que le lien entre deux sommets et le score de similarité des deux phrases. Il ne reste plus qu'à appliquer l'algorithme *PageRank*, et sélectionner les phrases de la manière de notre choix, selon leurs scores d'importance.

4.4 Apprentissage supervisé

Il existe encore d'autres modèles très efficaces mais qui relèvent de l'apprentissage supervisé. Le problème est traité ou bien comme un problème de classification binaire ("Telle phrase doit elle apparaître dans le résumé ?") ou bien cherchera à optimiser le score *ROUGE* (qui est une métrique d'évaluation des résumés automatiques et dont

nous reparlerons plus tard).

Plusieurs d'entre eux sont basés sur une modification de *BERT*. *BERT* (*Bidirectionnel Encoder Representations from Transformers*) [10] est un modèle de language pour représenter sous forme vectorielle un élement de language naturel. Un mot est alors représenté comme un vecteur continu dans un espace de faible dimension[7]. Certains modèles l'utilisent, puisque cela a de nombreux avantages, notamment celui de permettre aux modèle de comprendre les liens entre les mots (et de rapprocher notamment des synonymes). C'est le cas de *BertSumExt* [34], *HiBert* [60], et *MatchSum* [62]. Ces méthodes donnent en sortie pour chaque phrase la probabilité d'être dans le résumé.

D'autres méthodes supervisées n'utilisent pas *BERT*, mais utilisent des *Graph Convolutional Networks* (*GCN*) [29] c'est le cas de celle proposée par Yasunaga et al. : une méthode de synthèse multidocuments qui utilise un *GCN* prenant en entrée des *embeddings* de phrases issues de Réseaux de Neurones Récurrents (*RNN*).

L'utilisation des *RNN* est elle même récurrente c'est le cas notamment dans la méthode *SummaRuNNer* [43]. L'idée est d'identifier un ensemble de phrases qui maximiseront le score *ROUGE* (relativement donc au résumé de référence). Un avantage de ce modèle est qu'il permet diverses visualisations selon plusieurs caractéristiques (telles que la nouveauté ou la saillance).

Dans la grande majorité des cas d'applications, il est presque impossible d'avoir recours à des labels, c'est à dire à des textes pour lesquels une personne aurait défini le "résumé extractif parfait" (souvent appelé *golden summary*). Bien que ces méthodes fonctionnent très bien, devant la difficulté d'obtention et la subjectivité de ces "*golden summaries*", nous préfèrerons nous concentrer sur les méthodes non supervisées, c'est pourquoi nous ne développerons pas plus les méthodes classiques de synthèse supervisée.

7. Par exemple 300. On parle de "faible" dimension par rapport aux méthodes telles que *TF-IDF* ou *one-hot encoding* (un vecteur rempli de 0 et un 1 à la position du mot d'intérêt) dans lesquelles la représentation vectorielle est de la taille du vocabulaire, donc plusieurs dizaines de milliers en général.

5 La synthèse abstractive

La synthèse extractive peut se confronter à quelques inconvénients, tels que la cohérence, la cohésion, la manque de contexte, ou les répétitions. Cela est notamment du au fait que le modèle n'est pas capable de combiner des informations qu'il pourrait repérer à divers endroits d'un texte, ou encore qu'il n'est pas capable de comprendre à quoi font référence des pronoms. La synthèse abstractive est censée palier ces problèmes, en essayant d'imiter ce que pourrait faire un humain. Ce processus pourrait être représenté comme suit dans la figure 4.

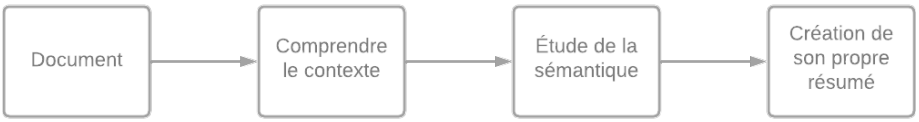


FIGURE 4 – Etapes classiques de la synthèse pour imiter le travail d'un humain.

La synthèse abstractive poursuit l'objectif de créer un résumé sémantiquement et syntaxiquement correct, en reformulant le texte du document à résumer. La synthèse abstractive ressemble plus à ce que ferait un humain si on lui demandait de résumer un texte. Le fait de pouvoir reformuler des phrases permet théoriquement de produire une synthèse plus courte et plus efficace qu'une synthèse extractive. Cette méthode est en général plus compliquée à mettre en place que la synthèse extractive, puisqu'elle nécessite une compréhension en profondeur de la sémantique et du sens du texte, dans le but de créer un résumé cohérent qui contiendra notamment des mots et phrases qui n'apparaissent pas dans le document initial.

Dans plusieurs méthodes, la synthèse abstractive consiste en une extraction des phrases clés puis une compression pour reformuler le résumé [6] [31].

5.1 Approches basées sur les ontologies

Une autre approche totalement différente est celle des ontologies. Une ontologie est une base structurée de connaissances et semble donc particulièrement pertinente pour représenter les informations présentes dans un texte. Une des principales méthodes par

ontologies est la "*fuzzy ontology method*" [32]. Cette méthode est efficace mais nécessite néanmoins des experts pour déterminer le domaine de l'ontologie, ce qui peut être, dans la plupart des cas, trop coûteux.

5.2 Approches basées sur la sémantique

5.2.1 *Encoder-Decoder*

De nombreuses recherches concernant la synthèse abstractive se reposent sur des modèles "*Seq2Seq*" (*Sequence to sequence*) [16] puis sur des méchanismes d'attention [2]. L'encodeur encode le texte initial en un "vecteur de contexte", qui contiendrait toutes les informations sémantiques et syntaxiques importantes, et le décodeur décode le vecteur en une séquence *target*[8] selon l'objectif donnée (dans notre cas la synthèse). Le méchanisme d'attention servira à localiser les morceaux ayant le plus d'intérêt lors du décodage. Ces méthodes ont initialement été pensées pour de la traduction automatique et donnent donc de bons résultats quant à la génération de texte.

5.2.2 *Pegasus*

Le modèle *Pegasus* (*Pre-training with Extracted Gap-sentences for Abstractive Summarization) [17]* est considéré comme un des meilleurs modèles pour la synthèse abstractive à ce jour. Il s'agit d'un réseau de neurones pré-entrainé à la tâche de synthèse automatique. Pour cela, *Pegasus* est entrainé sur une autre tâche, celle du *Gap Sentence Generation (GSG)*, c'est à dire devoir remplir des trous dans le texte. En particulier, il sera demandé au modèle de prédire des phrases ou morceaux de phrases particulièrement importants choisis en utilisant la méthode *ROUGE*. Il s'agit là d'une tâche auto-supervisée. Le fait que cette tâche soit très difficile (également pour des humains), force le modèle à apprendre en profondeur les aspects sémantiques et syntaxiques du texte fourni, ce qui est exactement ce qui est souhaité pour la tâche de synthèse automatique. Le modèle *Pegasus* est basé sur une architecture *Transformer* et combine également un encodeur et un décodeur. Le modèle *Pegasus* est particulièrement re-

8. cible

marquable, du fait qu'il s'agisse d'un modèle non-supervisé (ou tout du moins *self-supervised*) et donc qui ne nécéssite pas de base labellisée.

La plupart de ces méthodes abstractives restent cependant des méthodes supervisées. Il existe aujourd'hui plusieurs datasets qui permettent un entrainement efficace de ces méthodes, mais puisqu'il s'agit de méthodes très récentes, les meilleurs datasets sont en anglais. Il existe quelques datasets en français, mais plutôt pour générer des titres d'articles. [54]

Comme nous le verrons plus loin, la synthèse abstractive est prometteuse mais possède également de gros inconvénients. Ce ne sera donc pas la méthode que nous choisirons pour résoudre notre problème et nous ne développerons donc pas plus cette partie sur la synthèse abstractive.

Troisième partie

Étude du problème

Dans cette partie nous proposerons une approche personnelle, adaptée à notre contexte, pour résoudre notre problème d'extraction de sens. Rappelons que nous nous plaçons ici dans une problématique de transmission de sens, à partir d'un corpus de documents, mais que nous souhaitons pouvoir extraire une synthèse par document. Nous sommes également dans le cadre de documents en français, utilisant du vocabulaire parfois assez technique, chacun d'une longueur de quelques pages. Nous ne disposons pas de données d'entrainements semblables à celles de notre jeu de données.

6 Choix des pistes à explorer

6.1 Cadre du problème

Nous disposons d'un corpus de quelques milliers de tracts syndicaux, tous entre une et dix pages, et en français. Nous souhaitons mettre à disposition du bureau du dialogue social un outil pour l'aide à la transmission de sens. À la suite des réponses que nous avons pu avoir au questionnement que nous avons rapporté en introduction, nous souhaitons pouvoir fournir, pour chaque document, un résumé, des mots clés, et un thème. Nous pouvons voir cet exercice comme de la synthèse à plusieurs granularités : le résumé extrait les idées clés du documents (sous forme de phrases), les mots clés seront les mots les plus importants parmi ces phrases et le thème peut être vu comme un mot clé principal.

Se pose alors la question du type de résumé : extractif ou abstractif ? Comme nous l'avons vu précédemment, les méthodes abstractives sont beaucoup plus récentes et plus complexes, puisqu'elles nécéssitent non seulement une compréhension en profondeur des idées du texte mais également une rigueur syntaxique pour produire des phrases correctes et cohérentes. Les limites de temps et de puissance de calcul auxquelles nous

devons nous soumettre nous poussent à pencher plutôt pour la synthèse extractive. Par ailleurs, étant donné qu'il s'agit là seulement d'aide à la transmission d'informations, les résumés seront repris et probablement reformulés à la manière des personnes concernées et selon différentes orientations (qu'il s'agisse d'un résumé abstractif ou extractif). Nous préfèrerons donc donner les phrases et informations les plus objectives possibles et nous concentrer uniquement sur la sélection des phrases plutôt que sur la reformulation. Le fait que nos textes soient en français pourrait également être un obstacle à l'utilisation de modèles de synthèse abstractive pré-entrainés, la plupart étant des modèles très récents et donc entrainés sur des données en anglais. Par ailleurs, les documents que nous utiliserons sont, certes, en français, mais utilisent un vocabulaire très particulier et assez complexe, beaucoup de mots faisant parti du jargon de l'administration publique française. Un modèle de synthèse abstractive pourrait se heurter a des "OOV", c'est à dire des mots "*Out Of Vocabulary*", donc en dehors du vocabulaire qu'il connait. Cela n'est pas toujours important, en particulier lorsque les mots OOV sont rares. Dans notre cas ils risquent d'être fréquents, en plus de porter beaucoup de sens.

Notons que la synthèse extractive possède également des inconvénients, dont certains que nous avons déjà cités, comme le manque de contexte. Nous devrons donc trouver un moyen de palier ce problème. En outre, la synthèse extractive peut souvent être moins efficace sur les textes courts : si les informations du texte sont bien réparties entre cinq phrases d'un court texte, un résumé extractif d'une ou deux phrases aura de grandes chances d'être de mauvaise qualité. Cependant, nos textes sont assez longs pour contourner cet inconvénient.

6.2 Limites de la synthèse abstractive

Dans cette sous-partie, nous justfierons plus en détail notre choix de ne pas utiliser la synthèse abstractive.

Un premier point à noter est que la plupart des méthodes abstractives sont des méthodes supervisées, ce qui est également un frein au choix de cette technique.

6.2.1 Abstractif meilleur qu'extractif ?

Un second point à ne pas perdre de vue, est que plus complexe ne veut pas dire meilleur. Il est clair que les modèles abstractifs sont bien plus complexes que la majorité des modèles extractifs. Mais cette complexité vient en grande partie des efforts fournis pour produire un texte correct et cohérent. Dans la plupart des modèles abstractifs, les idées clés concernant la synthèse en elle-même sont rarement très novatrices, relativement aux modèles de synthèse extractive. Dans notre cas, nous n'avons pas besoin d'un modèle extrèmement impressionnant du point de vue de la génération de phrases. Nous n'avons pas besoin non plus d'une synthèse parfaitement correcte syntaxiquement, puisque la synthèse produite sera utilisée comme aide à la transmission de sens, et non pas restituée telle quelle. Nous préfèrons donc nous concentrer sur l'efficacité de la synthèse, d'autant que nous n'avons pas les ressources pour entraîner un modèle de language correctement dans les temps impartis. Il serait cependant possible que la synthèse abstractive soit, de toute manière, meilleure que la synthèse extractive, et cela pourrait remettre en question notre choix de la synthèse extractive. Il est vrai que, dans certains cas - pas dans tous, mais cela est vrai notamment pour la synthèse de courts articles d'actualités - la synthèse abstractive parvient à de meilleurs résultats que la synthèse extractive. Cependant, il est important de prendre en compte les méthodes d'évaluation, et selon quelle mesure il est possible de conclure que les résultats sont meilleurs. Nous discuterons en effet, dans la section 11 Évaluation, de la pertinence du score *ROUGE*[9], et de la subjectivité dans l'évaluation des résumés.

6.2.2 L'effet "*fake news*"

Il convient de se demander si, avec une méthode abstractive, le résumé est gramatical-lement et factuellement correct. L'aspect grammatical n'est pas primordial dans notre cas, mais l'aspect factuel, dans le sens où on ne veut que de vraies informations dans le résumé, l'est.

Il semblerait que la génération de texte ait tendance à "improviser". En effet, il n'est pas rare que les modèles de synthèse abstractive soient inconsistants, c'est à dire qu'ils

9. Métrique d'évaluation de la synthèse la plus utilisée.

génèrent des faits incorrects relativement au texte initial (ou qui n'y apparaissent simplement pas). En particulier depuis 2020, plusieurs études [39] [21] observent que malgré un score *ROUGE* élevé, les systèmes générant des résumés abstractifs sont souvent factuellement inconsistants par rapport au texte d'entrée. Cela est valable également pour les méthodes de synthèse abstractive les plus récentes, celles qui sont supposées être les plus performantes. Dans [19], plusieurs exemples sont donnés dont notamment des exemples avec le modèle *Bottom-Up Summary* [22], ou *BertAbs* (issu du même papier que *BertSumExt*) [34] que nous avons évoqué lors de l'état de l'art.

Un exemple assez parlant est donné dans [21] (voir Figure 5). Il s'agit d'un résumé généré automatiquement grâce au modèle *Pointer Generator* [55]. L'erreur produite est même assez grossière et semble évidente pour un humain.

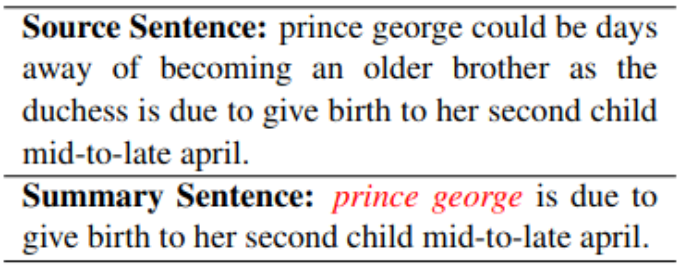


FIGURE 5 – Exemple d'inconsistance relevé dans le papier [21].

Nous souhaiterions alors essayer avec un de nos propres exemples en français, en utilisant un modèle pré-entrainé. Nous testons donc le modèle *BARThez* [12], un modèle *Seq2Seq* très récent (2021) entrainé en apprenant à reconstruire des phrases modifiées. Ce modèle a été entrainé sur des données françaises dans l'objectif d'accomplir des tâches de génération de texte. Il est possible d'utiliser ce modèle pré-entrainé en quelques lignes, comme on peut le voir dans la figure 6. Nous avons demandé au modèle de synthétiser le texte suivant :

"Le Bayern-Munich a fait un score de 3-0 contre l'ASSE ce qui est tout à fait historique en 2012. C'est une année très importante pour ce club qui continue de faire des scores impressionnants, au détriment d'autres équipes."

Comme on peut le constater dans le résultat renvoyé par le code de la figure 6, le résumé généré est le suivant :

```python
import torch

from transformers import (AutoTokenizer, AutoModelForSeq2SeqLM)

barthez_tokenizer = AutoTokenizer.from_pretrained("moussaKam/barthez")
barthez_model = AutoModelForSeq2SeqLM.from_pretrained("moussaKam/barthez-orangesum-abstract")

f = open('test.txt', 'r')
text_sentence = f.read()

input_ids = torch.tensor(
    [barthez_tokenizer.encode(text_sentence, add_special_tokens=True)])

barthez_model.eval()
predict = barthez_model.generate(input_ids, max_length=1000)[0]

barthez_tokenizer.decode(predict, skip_special_tokens=True)
```

```
'Le Bayern Munich a réalisé un triplé en 2012 et 2013 face à l'équipe de football de l'Olympique de Marseille.'
```

FIGURE 6 – Exemple d'inconsistance en français avec le modèle BARThez.

"Le Bayern Munich a réalisé un triplé en 2012 et 2013 face à l'équipe de football de l'Olympique de Marseille."

Ce résultat est effectivement inconsistant avec le texte de base puisqu'il comporte de grossières erreurs. En effectuant ce test, il était volontaire de donner des nombres dans le texte d'entrée, en espérant constater des erreurs puisque les modèles génératifs en commettent souvent sur ces points. Mais le résultat peut également paraître surprenant puisqu'il mentionne même une équipe de football qui n'était pas dans le texte original.

Nous testons également un second modèle entrainé sur des données en français disponible sur *Hugging Face*[10]. Il s'agit du modèle *French RoBERTa2RoBERTa* qui est un modèle basé sur *CamemBERT* [38], lui même étant un modèle de language français - décrit comme *state of the art* pour le français - basé sur le modèle *RoBERTa* [35]. *French RoBERTa2RoBERTa* a été ré-entrainé sur un dataset français dans le but d'effectuer des tâches de synthèse automatique. Comme on peut le voir dans la figure 7, les résultats ne sont pas moins inconsistants. On note cependant que, du point de vue de la génération de texte, le résultat est de bonne qualité. La phrase ressortie est totalement correcte, et le modèle a utilisé une périphrase tout à fait pertinente : "le club bavarois" pour "le Bayern-Munich".

Les papiers relevant ces erreurs proposent en général des méthodes pour mesurer l'in-

10. *Hugging Face* est une plateforme de *Data Science* permettant notamment d'avoir accès en ligne à des modèles de *NLP*, et les essayer très facilement grâce à des *API*. Le modèle essayé dans la figure 7 est disponible à l'adresse suivante : https://huggingface.co/mrm8488/camembert2camembert_shared-finetuned-french-summarization.

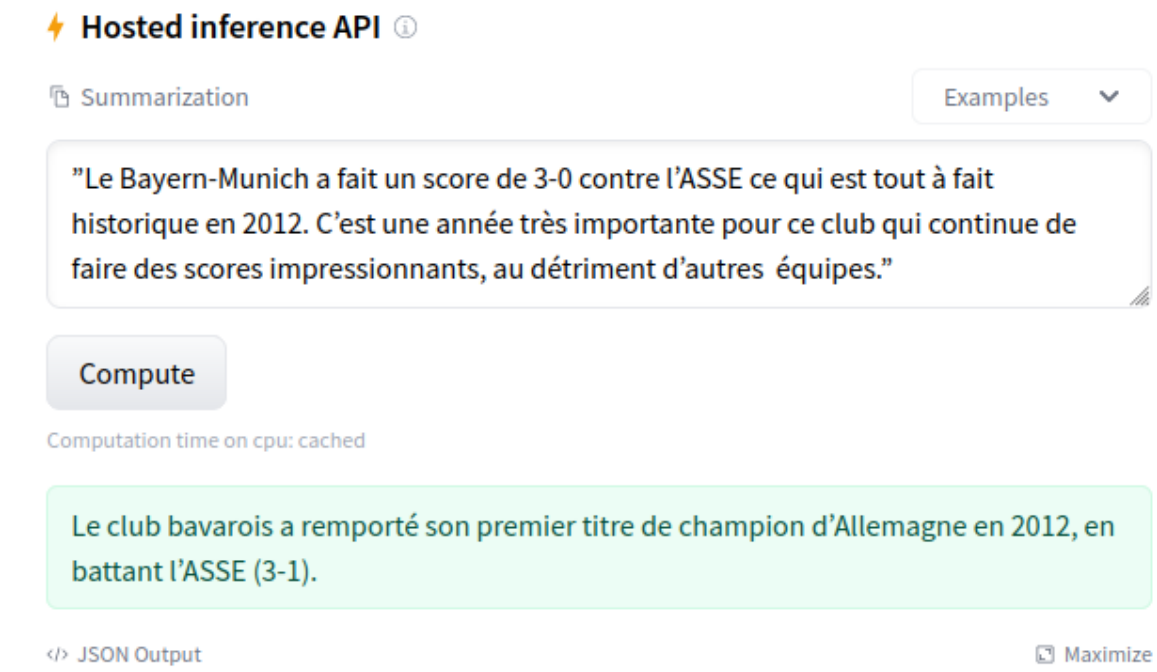


FIGURE 7 – Un second exemple de l'effet "*fake news*". Le résumé généré est inconsistant avec le texte d'entrée, ce qui peut se révéler très problématique.

consistance et la corriger, notamment en utilisant des questions-réponses [15]. L'idée dans le papier cité est que, pour une même question, les deux textes doivent fournir la même réponse. Si ce n'est pas le cas, c'est qu'il y a une incohérence quelque part.

6.3 Discussion sur la forme

Les principales difficultés de la synthèse automatique sont la gestion de la redondance, de la cohérence, de la cohésion, la perte d'informations et le manque de contexte. Le fait d'utiliser la synthèse extractive devrait déjà nous prémunir des problèmes de cohérence et de cohésion. Cependant, il faudra trouver à palier un des défauts majeurs de la synthèse extractive, c'est à dire le manque de contexte. Pour cela, il nous semble pertinent de ne pas présenter le résumé extractif comme simplement une suite de phrases tirées du texte initial, mais plutôt de désigner les phrases importantes directement à l'intérieur du texte. Ainsi, cela permettra aux lecteurs de repérer très rapidement les phrases les plus importantes et qui résument le texte à l'intérieur de celui-ci, de pouvoir éventuellement ne lire que ces phrases, mais aussi de pouvoir les lire dans leur contexte,

et éventuellement retrouver très rapidement un contexte manquant. Cette solution est d'autant plus pertinente dans notre contexte, puisque nous travaillons avec des tracts, c'est à dire des documents avec beaucoup de mise en page, voire d'images. Il semble donc important de garder cette mise en page lors de la présentation des résultats de la synthèse, puisqu'elle peut contenir des informations toutes aussi importantes que celles contenues dans le texte.

Nous souhaitons faire de la synthèse mono-document, mais nous disposons d'une grande quantité de documents. Il serait donc intéréssant de fournir une interface d'aide à la transmission de sens, notamment pour pouvoir trier et filtrer.

7 Préparation des données

Avant d'extraire du sens des textes, il convient de bien préparer les données.

Tout d'abord, puisque nous travaillons avec des *PDF*, nous utiliserons une librairie pour extraire le texte des *PDF*. Notons qu'en faisant ce choix, nous ne gardons plus aucune mise en page et nous perdons également la notion de titre et paragraphe. Puisque les tracts sont tous très différents, il n'est pas possible de déterminer directement où sont les titres et paragraphes. Une possibilité serait de repérer dans le document la taille d'écriture des différents textes. Ainsi, on considèrerait que la taille d'écriture la plus répandue correspond à celle des paragraphes, et donc que toutes les phrases de cette taille font parties des paragraphes. A partir de là, on peut déterminer hiérarchiquement que les morceaux de textes écrits plus gros sont des titres, sous-titres et ainsi de suite. Et de la même manière que les morceaux de textes écrits plus petits sont moins importants et corresepondent par exemple à des notes de bas de page.

7.1 Premier nettoyage

Pour appliquer nos algorithmes d'extraction de sens, nous souhaitons avoir les textes les plus propres possibles. Il est donc nécessaire de passer par une première phase de nettoyage, afin de supprimer au mieux ce qui n'est pas du texte, ou ce qui pourrait

gêner son traitement.

Grâce aux expressions régulières, nous effectuons les étapes suivantes :

— Retrait des coupures au milieu des mots en fin de ligne du document ;
— Retrait des liens ;
— Retrait des sauts de lignes parasites (ceux en fin de lignes, alors que la phrase n'est pas terminée) ;
— Retrait des images.

7.2 *Tokenization*

La *tokenization* consiste à séparer un texte en plusieurs unités selon certains séparateurs. Dans notre cas, nous aurons besoin de deux types de *tokenizers* : un qui sépare un texte en phrases, et un second qui sépare un texte en mots. En effet, notre objectif principal étant d'effectuer de la synthèse extractive il est important de séparer notre texte en phrases. Lorsque nous devrons calculer des fréquences de mots, nous aurons besoin d'une séparation en mots.

Il est important de garder, d'un côté, les phrases intactes, afin de produire un résumé lisible à la fin, et de faire les modifications sur une copie seulement. Lors de la *tokenization* par mots, nous supprimerons également la ponctuation, les caractères spéciaux, et mettrons le texte en minuscules.

En annexe A se trouve un exemple de texte *tokenizé* par phrase. Il s'agit du monologue de l'inconstance, tiré de *Dom Juan* de Molière [42]. Nous avons volontairement choisi pour cet exemple un texte assez court, et assez célèbre, pour pouvoir juger manuellement les résultats de nos modèles. Nous réutiliserons cet exemple.

7.3 *Stemming* et lemmatisation

Le *stemming* et la lemmatisation sont deux méthodes pour "normaliser un texte". L'objectif en utilisant ces méthodes est de réduire la taille du vocabulaire et de rapprocher

Mot	Lemmatisation	Stemming
informations	information	inform
informatrice	informateur	inform
ordinateur	ordinateur	ordin
ordinaux	ordinal	ordin
nicolas	nicolas	nicol

TABLE 1 – Exemples de normalisation de mots

deux mots identiques qui apparaissent sous des formes différentes (dans le sens où "aimera" et "aimaient" sont le même mot mais conjugués différemment). Ces techniques sont particulièrement utiles lors de l'utilisation de modèles utilisant les fréquences. A priori, un modèle de language tel que *BERT* est supposé rapprocher de lui même plusieurs formes d'un même mot.

La lemmatisation consiste à ramener les mots à leur forme "basique". C'est à dire, notamment, ramener les verbes conjugués à leur indicatif, et les noms et adjectifs à la forme masculin singulier. Le processus de lemmatisation ne produira que des mots corrects (des mots de la langue en question). Les librairies de lemmatisation récentes utilisent souvent un modèle de language pré-entrainé.

Le *stemming* consiste en général à couper la fin des mots pour ne garder à priori que la racine grammaticale. L'exemple du tableau 1 montre le résultat du *stemming* et de la lemmatisation sur différents mots.

Nous considèrerons que le stemming applique une modification trop brutale au texte, qui risquerait d'en changer le sens. Dans le tableau 1 par exemple, les mots "ordinateur" et "ordinaux" ont la même forme après *stemming*, alors qu'ils ont un sens très différent. Nous préfèrerons donc n'utiliser que la lemmatisation.

7.4 Construction du dictionnaire

L'étape suivante est la construction d'un dictionnaire. Ainsi, nous pourrons spécifier notre vocabulaire, et éliminer les *stopwords*. Il est possible d'utiliser une liste pré-faite de *stopwords*, comme il existe dans certaines librairies telles que *NLTK* ou *Spacy*. Il est également possible de spécifier manuellement sa propre liste de *stopwords*. Dans

notre cas, nous utiliserons une liste pré-faite et grâce au type de dictionnaire utilisé, nous pourrons également éliminer les mots qui apparaissent dans une trop grande proportion des documents. En effet, il est légitime de penser qu'un mot qui apparaît dans une phrase sur trois ou quatre (par exemple) ne sera pas très porteur de sens. C'est la même chose pour un mot qui apparaîtrait dans plus de la moitié des documents. Il est également possible de supprimer les nombres puisqu'ils ne comporteraient que peu d'information pour effectuer un résumé. Ces étapes sont facultatives mais permettent de réduire la taille du dictionnaire, qui peut très vite devenir très grande et ainsi augmenter les temps de calcul. Une autre manière de réduire la taille du dictionnaire, serait de supprimer de celui-ci les mots qui apparaissent moins d'un certain nombre de fois dans tout le corpus. En effet, il semble, encore une fois, légitime de penser qu'un mot qui n'apparaît qu'une ou deux fois dans un corpus de plusieurs milliers de documents n'aura pas une grande importance, voire qu'il s'agit d'une faute de frappe.

8 La synthèse automatique de texte

Nous commencerons notre tâche de transmission de sens par un travail sur la synthèse automatique. L'idée serait donc d'effectuer un résumé extractif mono-document dont la taille serait de 20% du texte initial (en nombre de phrases). Nos textes étant de longueur variable, il paraissait plus pertinent de choisir une proportion qu'un nombre de phrases fixe.

Nous allons étudier plus en détail l'algorithme *TextRank* [40]. *TextRank* est un modèle basé sur les graphes, pour traiter du texte. Le papier présente la méthode, puis donne deux exemples d'application : les phrases et mots clés. Le but de *TextRank* est plus précisément de classer des éléments de texte par ordre d'importance. Pour reprendre une phrase du papier de recherche, "un algorithme de classement basé sur les graphes est un moyen de décider de l'importance d'un sommet dans un graphe, en tenant compte des informations globales calculées de manière récursive à partir de l'ensemble du graphe, plutôt que de s'appuyer uniquement sur des informations locales spécifiques au sommet" [11].

11. "a graph-based ranking algorithm is a way of deciding on the importance of a vertex within a

Nous verrons d'abord comment traiter nos phrases pour qu'elles puissent entrer dans l'algorithme de graphe.

8.1 Représentation des phrases

Afin de rendre les phrases "compréhensibles" par notre ordinateur, il faut trouver une représentation numérique. Il existe de nombreuses manières de représenter les phrases, dont certaines que nous avons vues au sein de l'état de l'art. Ci-dessous, nous en détaillerons quelques unes qui nous semblent pertinentes pour notre algorithme. Dans le papier original, cette étape n'existe pas, et la similarité entre deux phrases est directement calculée selon le chevauchement des mots entre les phrases (nous détaillerons ce score plus tard). Nous préfèrerons cependant tirer parti d'autres méthodes, notamment des méthodes plus récentes.

8.1.1 Représentation par la fréquence des mots.

Nous rappelons une première possibilité, évoquée lors de l'état de l'art pour représenter le texte grâce au score *TF-IDF* : On construit, pour un texte, une matrice de la taille (nombre de phrases x nombre de mots du voabulaire), dont la case $[i][j]$ correspond au score *TF-IDF* du mot j par rapport à la phrase i. La représentation de la phrase i est donc la ligne i en entier. Cette méthode a l'avantage d'être très simple, très rapide, mais cependant efficace. Elle a néanmoins pour inconvénient de produire une matrice de caractéristiques très grande à cause de la taille du vocabulaire. La matrice sera cependant très creuse, et il existe des types d'objets adaptés en Python (dans la classe *scipy.sparse* par exemple).

8.1.2 Vectoriser un mot

Une seconde méthode consiste à tirer parti des modèles de language et représenter une phrase sous forme de vecteur. Il existe alors plusieurs manières et plusieurs modèles

graph, by taking into account global information recursively computed from the entire graph, rather than relying only on local vertex-specific information."[40]

pour le faire. Un modèle de language tel que *BERT* [10] est très long et compliqué à entraîner, l'idée est donc d'utiliser un modèle pré-entraîné.

L'avantage des *embeddings* de mots est de permettre l'attribution d'un sens sémantique à un mot. Ainsi deux mots ayant un sens totalement opposé auront des vecteurs éloignés, tandis que deux mots très proches auront des vecteurs proches (dans le sens d'une distance). Ainsi, les mots "pêche" et "abricot" seront proches dans l'espace vectoriel des mots, et le mot "pleuvoir" en sera probablement assez éloigné.

***Word2Vec* (2013)** Un des modèles de *word-embedding* les plus populaires est très certainement *word2vec*, issu du papier *Efficient Estimation of Word Representations in Vector Space* [41] publié en 2013 par des chercheurs de Google. Le modèle *Word2Vec* permet également d'effectuer des opérations algébriques sur les vecteurs de mots. L'exemple le plus connu de cette application est le suivant, donné dans le papier :

$$\text{vector("King")} - \text{vector("Man")} + \text{vector("Woman")} = \text{vector("Queen")}$$

Word2Vec est basé sur deux modèles : *CBOW*, à gauche de la figure 8, et *Skip-Gram*, à droite. Les deux sont des modèles *self-supervised* (auto-supervisés) c'est à dire qui trouvent leurs labels dans leurs propres *inputs*[12]. Plus clairement, *CBOW* s'entraine à effectuer la tâche suivante : En prenant une phrase ou un morceau de phrase, un mot est caché. En prenant le contexte de ce mot caché (donc n mots à gauche et n mots à droite), le modèle devra retrouver le mot caché. C'est en quelques sortes la tâche inverse que va effectuer *Skip-Gram*, puisque selon un mot, le modèle devra retrouver le contexte.

***Glove*(2014)** *Glove* est un modèle publié par des chercheurs de Stanford. La principale différence en pratique par rapport à *Word2Vec* est que ce modèle prend en compte les fréquences de co-occurence de mots.

FastText(2016) Dans le même principe que *Word2Vec* et *Glove*, *FastText*[48], modèle développé par Facebook en 2016 apprend des représentations vectorielles pour des mots.

12. Données d'entrée. En l'occurence des textes.

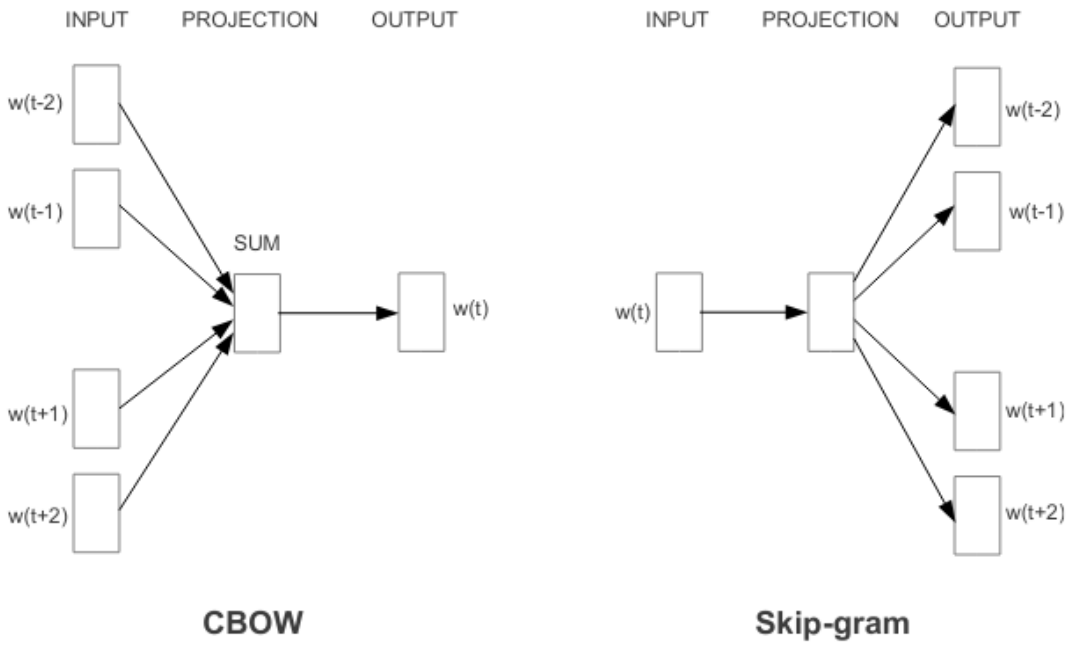


FIGURE 8 – Architecture des modèles *CBOW* et *Skip-Gram*.

Cependant, *FastText* est entrainé à une granularité différente : à l'échelle des lettres plutôt que des mots. Cela peut permettre de rapprocher plus facilement des mots avec une racine commune, ce qui peut rendre le modèle plus efficace pour certaines tâches, notamment dans certaines langues où les mots sont souvent composés d'autres mots (d'où le titre du papier de recherche : *Enriching Word Vectors with Subwords Information*).

Un gros avantage de ce modèle par rapport à *Word2Vec* est que, grâce à sa prise en compte des mots à l'echelle de la lettre, le modèle sera capable de gérer des mots qu'il n'a jamais vu auparavant, c'est à dire des mots *OOV* ("*Out of Vocabulary*"). Pour cela, le modèle prend le vecteur moyen des *n-grams* du nouveau mot.

***BERT* (2018)** Le modèle *BERT*[10] (*Bidirectional Encoder Representations from Transformers*), publié en 2018 par Google, est plus complexe et utilise notamment un modèle d'attention [58]. En particulier, le modèle utilise une nouvelle technique d'entrainement bidirectionnel : *Masked Language Modelling* (*MLM*) [13], et le *Next Sentence*

13. Le *MLM* consiste à donner au modèle des phrases à trou et lui demander de prédire les mots manquants.

Prediction (*NSP*) [14], permettant une meilleure prise en considération du contexte.

C'est un modèle plus avancé et c'est celui que nous utiliserons (ou tout du moins, nous utiliserons un modèle basé sur *BERT*).

8.1.3 Vectoriser une phrase

Avec le succès des modèles pour calculer les *embeddings* de mots, sont apparues des méthodes pour calculer des *embeddings* de morceaux de textes plus longs, que ce soient des phrases ou des paragraphes. Il existe ensuite plusieurs moyens pour vectoriser une phrase. L'objectif est d'attribuer à chaque phrase une représentation vectorielle de longueur fixe.

Somme pondérée des *word embeddings* Comme nous avons pu en discuter plus tôt, plus complexe ne veut pas dire meilleur. Il existe de nombreux modèles très complexes qui calculent les *embeddings* de phrases directement (sans forcément passer par les mots), mais certains papiers, notamment "*Towards Universal Paraphrastic Sentence Embeddings*" [59] en 2016, soulignent le fait que pour certaines applications "*out-of-domain*", telles que la similarité de phrases (et donc ce qui nous intéresse ici), des modèles complexes étaient en réalité moins performants que des modèles bien plus simples. Ce papier est également venu avec la proposition d'une méthode plus simple, mais pas moins efficace, basée sur une moyenne des *embeddings* de mots..

Une méthode basique serait de prendre une sorte de somme pondérée des vecteurs des mots qui composent la phrase. Il est possible d'utiliser tout simplement la moyenne, mais une autre idée serait de pondérer les vecteurs de mots par un facteur inversement proportionnel à leur fréquence [1] (et ainsi, donner moins de poids dans le calcul aux mots vides (*stopwords*) qu'au mots un peu moins fréquents mais bien plus porteurs de sens). Cette méthode est issue d'un papier qui vise à proposer une méthode encore plus simple est efficace que [59] (elle même se voulant être une méthode simple et efficace). Les auteurs ont d'ailleurs voulu traduire cette idée dans le titre du papier : "*A simple but Tough-To-Beat Baseline For Sentence Embeddings*". Ils vont jusqu'à décrire

14. Prédire la phrase suivante

Algorithm 1 Sentence Embedding

Input: Word embeddings $\{v_w : w \in \mathcal{V}\}$, a set of sentences $\mathcal{S}$, parameter a and estimated probabilities $\{p(w) : w \in \mathcal{V}\}$ of the words.
Output: Sentence embeddings $\{v_s : s \in \mathcal{S}\}$

for all sentence s in $\mathcal{S}$ **do**
  $v_s \leftarrow \frac{1}{|s|} \sum_{w \in s} \frac{a}{a+p(w)} v_w$
end for
Form a matrix X whose columns are $\{v_s : s \in \mathcal{S}\}$, and let u be its first singular vector
for all sentence s in $\mathcal{S}$ **do**
  $v_s \leftarrow v_s - uu^\top v_s$
end for

FIGURE 9 – Algorithme de *sentence embedding* [1]

leur méthode comme "*embarassingly simple*". La première méthode simple citée [59] nécéssitait cependant un ré-entrainement avec un *dataset* labellisé pour la tâche. La méthode "*Tough-To-Beat*" va plus loin en proposant une méthode qui ne nécéssite aucune donnée labellisée (*unsupervised*).

La méthode consiste en les étapes suivantes :

— 1. Utiliser le modèle de notre choix pour calculer les *word embeddings* d'un texte ;
— 2. Calculer, pour chaque phrase, la somme pondérée des vecteurs de mots de la phrase : Le poids d'un mot w, le *SIF* (*Smooth Inverse Frequency*), vaut $a/(a+p(w))$, $p(w)$ étant la fréquence estimée du mot, et a un paramètre entre 0 et 1 (chaque vecteur de phrase est ensuite normalisé par la taille de la phrase) ;
— 3. Retirer à la somme la projection de la moyenne des vecteurs sur leur premier vecteur singulier ("*common component removal*").

L'algorithme écrit par les auteurs du papier est donné en Figure 9.

Un inconvénient de cette méthode est qu'il ne prend pas l'ordre des mots en compte. Il est donc possible que deux phrases, ayant approximativement les mêmes mots, mais un sens différent, soient très proches dans l'espace des phrases. On peut prendre l'exemple des phrases "Il n'est pas certain que ce projet ait de l'avenir." et "Ce projet a un avenir certain.", qui ont des sens opposés mais des mots très similaires, dans un ordre différent. Cependant, notre objectif étant de trouver les morceaux de texte les plus importants, cet inconvénient sera moins important que s'il s'agissait d'une tâche de

question-answering[15].

Notons que la papier a été publié en 2017, soit à peu près au même moment que *BERT*, et n'a donc pas été testé avec ce modèle (il a été testé avec *GloVe* [14] et *PSL* [27]). Cependant, il est bien précisé que la méthode doit pouvoir s'appliquer avec n'importe quel type d'*embedding*.

L'idée sera donc pour nous d'essayer cette méthode de vectorisation de phrase avec *CamemBERT* [38], dont nous avons déjà parlé précédemment et qui conviendra bien à nos données en français pour faire de la synthèse.

8.2 Score de similarité

Dans le but d'utiliser *TextRank*, nous avons besoin d'un score de similarité entre les phrases afin d'attribuer des poids aux arêtes. Pour cela nous avons encore une fois plusieurs possibilités. Une possibilité serait d'utiliser directement une mesure telle que la plus grande sous-sequence commune (mesure qui peut revenir dans certaines méthodes d'évaluation) mais nous préfèrerons tirer parti des représentations de phrases que nous venons de décrire.

8.2.1 Similarité *TextRank*

Avant de voir comment calculer la similarité grâce aux représentations vectorielles de phrases, voyons d'abord la méthode présentée dans le papier *TextRank*, qui est une mesure directe (sans passer par la vectorization). Il est question du score de similarité suivant entre une phrase S_i et une phrase S_j :

$$Similarity(S_i, S_j) = \frac{|\{w_k | w_k \in S_i, w_k \in S_j\}|}{log(|S_i|) + log(|S_j|)} \tag{2}$$

Ce score représente le chevauchement normalisé des deux phrases, c'est à dire le nombre de leurs mots en communs, normalisé par le logarithme de leurs longueurs (en nombre

15. La tâche de *Question-Answering* consiste à répondre automatiquement à une question à partir d'un texte.

de mots).

8.2.2 Similarité cosinus

Une deuxième méthode, celle que nous allons utiliser, est la similarité cosinus. Il s'agit d'une mesure de similarité entre deux vecteurs (non nuls et de même dimension). Quelle que soit la représentation que nous aurons choisi (en utilisant le score *TF-IDF*, en utilisant les *word embedding*, ou n'importe quelle autre méthode pour représenter les phrases sous forme de vecteur), nous aurons donc un vecteur par phrase. Pour calculer le score de similarité entre deux phrases, dont les vecteurs (de même taille) sont V_i et V_j, on utilise le produit scalaire des deux vecteurs divisé par le produit de leurs normes :

$$\frac{V_i \cdot V_j}{||V_i|| \times ||V_j||} \tag{3}$$

Il s'agit donc du cosinus de l'angle des deux vecteurs, et est compris entre -1 et 1. Afin d'obtenir un score entre 0 et 1, on peut tout simplement clipper la valeur dans cet intervalle, c'est à dire ramener toutes les valeurs négatives à 0. Ce qui nous intéresse sont en effet les phrases similaires, donc avec des scores relativement proches de 1, donc il n'est pas dérangeant de supprimer les valeurs négatives.

Quel que soit le calcul de similarité choisi, il faudra calculer la similarité pour chaque paire de phrase. Nous obtenons donc une matrice X de taille $(n \times n)$ avec n le nombre de phrases et dont $X[i, j]$ indique la similarité entre la phrase i et la phrase j.

8.3 Construction du graphe et score des phrases

Pour pouvoir utiliser l'algorithme *TextRank*, nous devons maintenant transformer notre texte en graphe. Il faut que les sommets du graphe repésentent les unités de texte que l'on veut classer, ici donc les phrases. La construction du graphe est très simple :

— On construit un sommet par phrase. Dans notre implémentation, un sommet portera le numéro de la phrase qu'il représente. Cela permettra de dessiner le graphe plus facilement.

— On construit un lien entre chaque paire de sommet dont le score de similarité est supérieur au seuil fixé (ou supérieur à 0 si aucun seuil n'est fixé).

Une des principales différences avec la modélisation du graphe pour *PageRank*, est que les arêtes sont non orientées (dans notre cas en tout cas, mais elles pourraient l'être), et ont des poids. En effet, dans l'algorithme initial, puisque les arêtes représentent des liens, une arête est soit existante soit inexistante, il n'existe pas de liens partiels. Cependant, cela ne pose aucun problème pour l'algorithme.

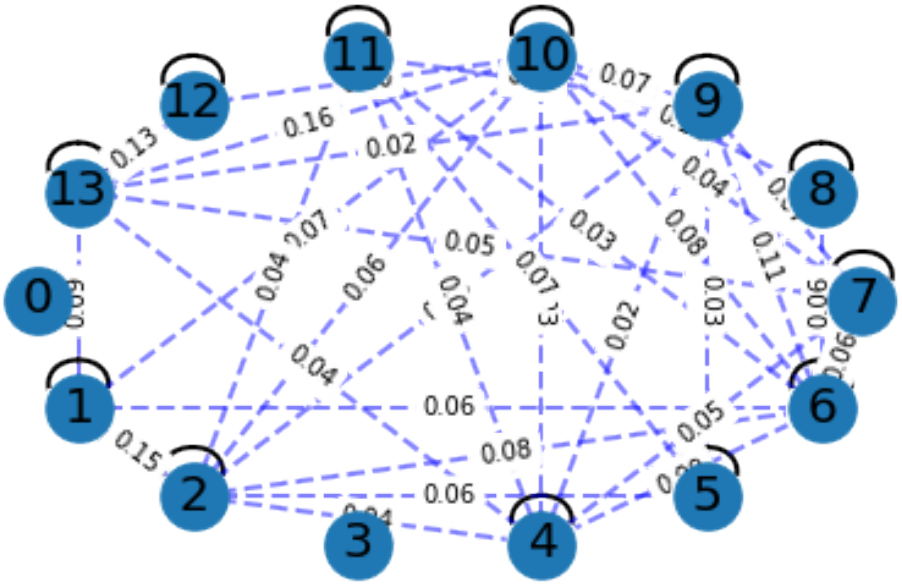


FIGURE 10 – Graphe du texte en Annexe B. Les numéros des sommets représentent les numéros de phrase du texte et les poids sur les arêtes représentent la similarité entre deux phrases ayant été représentée avec la méthode *TF-IDF*.

Dans la figure 10, on trouve un exemple d'un tel graphe, issu du texte en annexe A. On note que certains sommets, les sommets 0 et 3, ne sont reliés à aucun autre sommet. Les phrases 0 et 3 sont effectivement des phrases très courtes qui ne ressemblent à aucune autre. Pour cet exemple, nous avons utilisé la méthode utilisant le score *TF-IDF* pour représenter les phrases. Si deux phrases n'ont aucun mot en commun, la similarité cosinus sera nulle, ce qui explique que toutes les phrases ne soient pas reliées.

La figure 11 montre un graphe pour le même texte. Mais cette fois les similarités font suite à une vectorisation des phrases par la méthode "*Tough-To-Beat*" décrite précédemment [1]. Cet exemple est beaucoup moins lisible puisque presque toutes les

paires sont reliées. Cela est dû à la vectorisation choisie : En utilisant la méthode *TF-IDF*, la matrice de représentation est très creuse, c'est à dire que les vecteurs des phrases contiennent beaucoup de 0. En effet, Le vecteur d'une phrase est de la taille du vocabulaire, et ne contient des valeurs qu'aux indices des mots qui composent la phrase, et des 0 partout ailleurs. Lorsqu'on utilise la méthode "*tough-to-beat*" pour la vectorisation de phrases, il est presque impossible qu'une coordonnée soit nulle. Pour qu'un score de similarité soit nul, et donc qu'il n'y ait pas d'arête, il faudrait que le score soit négatif et donc clippé à 0, ce qui ne semble pas arriver dans notre exemple. Si on veut moins d'arêtes, il est également possible de ramener toutes les valeurs inférieures à un certain seuil (0.4 par exemple) à 0.
On note encore une fois que les sommets 0 et 3 ont des scores de similarité très faibles avec les autres sommets (les arêtes dont le poids est inférieur à 0.5 sont représentées en violet).

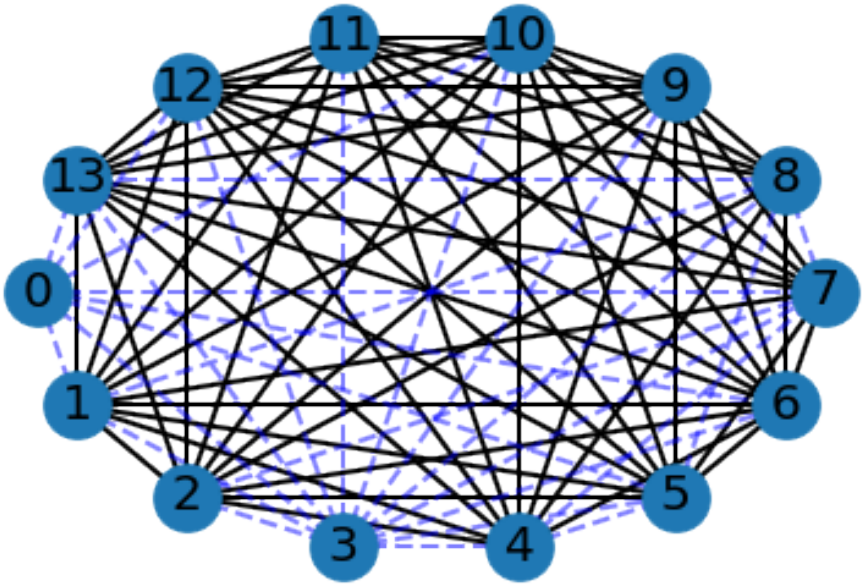


FIGURE 11 – Graphe du texte en Annexe A. Les phrases ont été vectorisée grâce à la méthode de *sentence embedding* décrite plus haut.[1] Les arêtes violettes ont un score inférieur à 0.5 et les noires un score supérieur à 0.5

En transformant le texte en graphe, on ne cherchera plus les phrases/mots les plus important(e)s, mais simplement les sommets du graphe les plus importants. L'idée gobale (issue notamment du papier *PageRank* [46]) est que les liens sont des votes/recommandations et donc plus une phrase ou un mot (ou une page à la base) a de votes, plus elle est

importante. Néanmoins les votes peuvent aussi avoir des importances différentes : Une phrase qui ressemble à (recommandée par) une autre phrase importante aura plus de chance d'être importante. Finalement, le score d'un sommet sera fonction du nombre et de l'importance des arêtes de ce sommet.

En modifiant la formule du papier *PageRank* pour intégrer les poids des arêtes, le score d'un sommet V_i est défini comme suit :

$$WS(V_i) = (1-d) + d \times \sum_{V_j \in In(V_i)} \frac{w_{ji}}{\sum_{V_k \in Out(V_j)} w_{jk}} WS(V_j) \tag{4}$$

d est un facteur compris entre 0 et 1 (souvent 0.85, c'était la valeur dans le papier original, et c'est celle qui est reprise la plupart du temps) qui représente la probabilité de sauter d'un sommet à un autre sommet aléatoire. Cette idée vient du papier original *PageRank* où l'algorithme implémentait le modèle du surfeur aléatoire, dans lequel un utilisateur cliquerait sur un lien aléatoire de sa page actuelle avec une probabilité d et serait téléporté vers une nouvelle page avec probabilité $1-d$.[16] $In(V)$ et $Out(V)$ représentent respectivement l'ensemble des prédécesseurs (sommets avec une arête allant vers V) et successeurs (sommets avec une arête provenant de V) de V. Notons que dans notre cas, les arêtes sont non-orientées. En effet, la relation de similarité etant une relation d'équivalence, si S_i est similaire à S_j, alors S_j est similaire de la même manière à S_i.

Pour n phrases, nous nous retrouvons alors avec n équations (4), formant un système récursif à n variables. En ajoutant certaines contraintes (notamment le fait que la somme des scores doit être égale à 1), on peut résoudre le système (par exemple grâce à la méthode de la puissance itérée) et obtenir une unique solution - en théorie tout du moins : en pratique la solution dépend de si l'algorithme de résolution a convergé, du nombre d'itération, et du seuil de convergence défini.

Après convergence, chaque phrase s'est vue attribuer un score. Ce score est supposé représenter à quel point une phrase représente les sujets les plus importants d'un texte.

16. L'idée de *PageRank* est d'appliquer leur modèle aux pages *web* et de déterminer lesquelles sont les plus importantes. L'ensemble des sommets est ainsi l'ensemble des pages Internet, et l'ensemble des arêtes orientées est l'ensemble des liens allant d'une page vers une autre.

8.4 Sélection des phrases

Il ne reste alors plus qu'à sélectionner les phrases. La méthode la plus intuitive et la plus répandue reste de sélectionner simplement les phrases ayant le meilleur score, c'est ce qu'on pourrait appeler la méthode gloutonne. Le nombre de phrases sélectionnées peut dépendre soit d'un nombre de phrases défini, un nombre de mots défini, d'un pourcentage du texte (en nombre de mots ou phrases). Il est également possible de sélectionner toutes les phrases dont le score est supérieur à un certain seuil.

Une autre approche, que nous avons également déjà évoquée lors de l'état de l'art, serait de convertir le problème de sélection de phrases en un problème d'optimisation dans lequel un ensemble de phrases qui maximise la quantité d'informations transmise et minimise la redondance, doit être choisi sous certaines contraintes (de taille notamment).

8.5 Quelques résultats

Nous avons effectué des tests sur quelques courts textes en utilisant plusieurs méthodes de vectorisation. Dans la table 2, on retrouve pour chaque méthode et chaque texte le top 5 des indices des phrases considérées comme les plus importantes. La première méthode du tableau, "*TF-IDF*", consiste à utiliser la méthode *TF-IDF* décrite plus haut pour vectoriser les phrases. La seconde, "Baseline", définit le vecteur d'une phrase comme la moyenne des vecteurs des mots qui la composent. La troisième méthode "T1" utilise la méthode "Tough-To-Beat", mais n'utilise pas les fréquences réelles des mots, mais plutôt une constance. Nous avons voulu tester cette alternative en pensant que les fréquences des mots pour un petit corpus (et nous avons ici fait notre test sur un corpus de 3 documents assez courts) ne seraient pas très significatives. La dernière méthode du tableau "T2", utilise la méthode "*Tough-To-Beat*", en utilisant les fréquences de mots calculées sur le corpus.

En annexe B, sont rapportés les résultats des tests du tableau mais avec les trois phrases les plus importantes surlignées dans le texte apparaissant en entier. Cela permet de juger les résultats manuellement. Nous avons choisi de surligner 3 phrases à chaque fois

Texte\Méthode	*TF-IDF*	Baseline	T1	T2
Monologue	[**10**, **6**, 2, **13**, **9**]	[**10**, 11, **6**, **13**, **9**]	[**10**, 11, **6**, **13**, **9**]	[**10**, 11, **6**, **13**, **9**]
Tract	[**6, 4, 1**, 0, 2]	[**1, 6, 4**, 3, 11]	[**6, 1, 4**, 3, 11]	[**6, 4, 1**, 8, 5]
Wiki	[1, 2, 0, 3, 10]	[12, 3, 2, 4, 13]	[12, 16, 4, 13, 2]	[3, 13, 16, 4, 12]

TABLE 2 – Top 5 des phrases selon différentes méthodes de vectorisation, et pour trois documents différents. Les phrases en commun pour tous les modèles sont indiquées en gras. L'ordre indiqué est l'ordre d'importance.

puisque cela représente environ 20% du nombre de phrases.

Une première chose intéréssante à noter, est que, pour les deux premiers textes, les 4 méthodes donnent des résultats assez similaires. En effet, pour le texte "monologue", 4 phrases sont communes à toutes les méthodes, et la 5ème est commune au trois méthodes utilisant un modèle de language. Pour le second document, le top 3, donc les phrases surlignées, c'est à dire celles qui appartiendront au résumé final, sont toujours les mêmes (même si l'ordre peut être différent). Pour le dernier texte, "Wiki", les résultats entre les méthodes se basant sur les modèles de language et *TF-IDF* sont très différents.

Même en ayant les textes et les phrases surlignées sous les yeux, il est assez difficile de dire quel modèle est le meilleur. Par exemple, pour le premier texte, la seule différence est que la phrase 2 ("La belle chose de vouloir se piquer d'un faux honneur d'être fidèle, de s'ensevelir pour toujours dans une passion, et d'être mort dès sa jeunesse à toutes les autres beautés qui nous peuvent frapper les yeux ! " a été sélectionnée dans la méthode *TF-IDF*, et la 11 ("Enfin il n'est rien de si doux que de triompher de la résistance d'une belle personne, et j'ai sur ce sujet l'ambition des conquérants, qui volent perpétuellement de victoire en victoire, et ne peuvent se résoudre à borner leurs souhaits. ") dans les autres. Mais il est difficile de dire quelle est la meilleure phrase des deux. Concernant le dernier texte, les résumés produits sont très différents. On remarque que les résumés issus de l'utilisation de modèles de language contiennent des phrases plus variées, tandis que la méthode *TF-IDF* semble privilégier beaucoup les premières phrases car elles contiennent beaucoup le mot "résumé".

Pour notre problème d'extraction de sens des tracts, nous avons choisi la méthode T2, donc en utilisant la vectorisation "*Tough-to-Beat*" et les vraies fréquence (ce que l'on peut se permettre puisqu'elles seront calculées sur un corpus d'environ 12 000 tracts),

qui a été la plus approuvée lors de tests sur différents tracts. Sur certains tracts, des phrases sont déjà mises en valeur (en gras, en rouge, ou soulignées par exemple). On peut donc considérer que ces phrases sont effectivement importantes puisqu'elles ont été considérées comme tel par l'auteur du tract, et qu'elles devraient probablement apparaître dans le résumé. Comme nous ne disposions pas de jeu de données avec des résumés de référence et qu'il était trop compliqué d'en construire un pour évaluer nos modèles sur une grande quantité de documents avec une métrique numérique, la mise en valeur de phrases nous a beaucoup aidé à sélectionner notre modèle, en plus des retours du Bureau du Dialogue Social, le métier intéréssé par ce problème. Un exemple d'un tract pour lequel les phrases mises en valeur (écrites en rouge) ont bien été surlignées par l'algorithme est donné en annexe C.

L'évaluation de la synthèse automatique est assez délicate, d'autant plus lorsque nous ne disposons pas de résumés référence pour nos données, et que nos données sont trop différentes des jeux de données labellisés qui sont souvent utilisés pour l'évaluation des modèles. Par ailleurs, même si nous disposions d'un tel jeu de données, les méthodes d'évaluation ne sont pas forcément très efficaces. La section 11 Évaluation est dédiée à une discussion sur les méthodes d'évaluation.

9 Extraction de mots clés

L'extraction de mots clés est une tâche assez similaire à celle de la synthèse automatique. En effet, elle peut être vue de la même manière que la synthèse (ou extraction de phrases clés) mais à l'échelle des mots. La tâche consiste plus précisément à identifier automatiquement dans un texte un ensemble de mots qui décriront le mieux le document.

Les mots clés peuvent servir tout simplement à cerner un texte et ses sujets rapidement, à construire un index, ou à classifier des textes par exemple.

Pour l'aide à la transmission de sens, les mots clés peuvent se réveler cruciaux, et peuvent permettre en un coup d'oeil de connaître les thèmes principaux d'un document.

Nous étudierons ici uniquement des méthodes non supervisées. Il existe en effet des méthodes supervisées, mais en pratique il est rare d'avoir accès à suffisamment de données labellisées, d'autant que ces méthodes performent souvent assez mal dans d'autres domaines que celui d'entraînement, en particulier pour des domaines ayant un jargon assez spécifique.

9.1 Algorithmes

9.1.1 Méthodes statistiques

Une première méthode peut être d'utiliser à nouveau le score *TF-IDF*. Cette fois, plutôt que de calculer le score *TF-IDF* d'un mot par rapport à une phrase, dans le contexte d'un document, il faudra calculer le score d'un mot par rapport à un document dans le contexte d'un corpus de textes. Ainsi, le score *TF-IDF* mettra en valeur pour un texte des mots qui apparaîssent souvent dans ce texte mais relativement peu dans les autres textes. Cette méthode est très simple et rapide mais nécessite cependant un corpus de taille conséquente et avec des documents concernant des thèmes variés pour atteindre son potentiel maximum.

Une seconde méthode statistique, beaucoup plus récente est *YAKE* (" *Yet Another Keyword Extractor*"). Cette méthode consiste en cinq étapes :

— 1- Le *preprocessing*, qui correspond globalement aux étapes que nous avons décrites dans la partie sur le préparation des données ;
— 2- L'extraction des caractéristiques : On calcule pour chaque terme 5 caractéristiques statistiques, notamment des caractéristiques qui prennent en compte si le mot est écrit en majuscules, la position du mot dans le texte et la fréquence ;
— 3- Le calcul du score agrégé par terme ;
— 4- La génération des *n-grams* et le calcul de leurs scores aggrégés : On peut associer certains mots en *n-grams*, et former ainsi des "mots clés" qui comprennent plusieurs mots ;
— 5- Déduplication des mots clés (pour éviter la redondance) et sélection des meilleurs mots.

L'avantage de cette méthode par rapport à *TF-IDF* est qu'elle ne nécéssite rien d'autre que le texte en question, pas de corpus. Cependant il s'agit d'une méthode beaucoup plus complexe à implémenter, et plus longue à calculer.

9.1.2 *TextRank*

Il est également possible d'utiliser *TextRank* pour trouver les mots clés. Il faudra simplement changer la manière de calculer les poids des arêtes, puisque les liens entre les mots ne sont pas les mêmes que les liens entre les phrases.

Une autre différence avec *TextRank* pour les phrases, est qu'on pourra sélectionner un préfiltre pour réduire le nombre de mots candidats et donc de sommets. Il est possible de ne sélectionner par exemple que les noms, que les verbes, que les adjectifs, ou une combinaison.

Une fois les mots candidats choisis, on construit un sommet par mot. Il reste donc à choisir quelles arêtes et avec quel poids ajouter.

Construction du graphe dans le papier Sous l'hypothèse que des mots apparaissant souvent ensemble auront des sens proches, la relation de co-occurence peut bien représenter les liens entre les mots. Deux sommets (et donc deux mots) seront connectés s'ils apparaissent ensemble dans une fenêtre de taille n (n étant compris entre 2 et 10). Dans ce cas, les arêtes n'ont pas de poids.

Autre proposition à partir de l'idée du papier Il nous paraissait assez limité de ne considérer que la co-occurence de manière binaire. En effet, cela ne prend pas en compte le fait que certaines paires de mots apparaissent plus ou moins souvent ensemble. Nous avons donc pensé que le *PMI* ("*Pointwise Mutual Information*") serait une bonne mesure des liens entre deux mots. Le *PMI* essaie de mesurer à quel point deux mots sont susceptibles d'apparaître ensemble relativement à leur chance d'apparaître ensemble s'ils étaient indépendants. Encore en 2021, le *PMI* est décrit comme l'un des concepts les plus importants du *NLP* [28]. Fano, en 1961[51], a définit

l'information mutuelle de deux mots x et y comme suit :

$$PMI(x, y) = log_2 \frac{P(x, y)}{P(x)P(y)} \tag{5}$$

Avec :

$$p(x, y) = \frac{\#W(x, y)}{\#W} \tag{6}$$

$$p(x) = \frac{\#W(x)}{\#W} \tag{7}$$

$\#W(i)$ étant le nombre de fenêtres (dans tout le corpus) contenant le mot i, $\#W(x, y)$ étant le nombre de fenêtres (toujours dans tout le corpus) contenant les mots i et j, et $\#W$ étant le nombre total de fenêtres dans le corpus.

Le numérateur de (5) indique à quelle fréquence les deux mots apparaissent ensemble, et le dénominateur indique à quel fréquence on pourrait s'attendre à ce qu'ils apparaissent ensemble s'ils étaient totalement indépendants. Très souvent, on utilise le *PPMI* (*Positive PMI*) plutôt que le *PMI*, qui se calcule en clippant les valeurs négatives du PMI à 0, car, à moins d'un immense corpus, les valeurs négatives ne sont pas très significatives (cela signifierait que deux mots apparaissent significativement moins ensemble dans un corpus que le hasard). On peut aussi ramener à 0 toutes les valeurs inférieures à 0.15 par exemple.

Pour utiliser cette méthode, il faut calculer le *PMI* de toutes les paires de mots candidats. On place ensuite une arête dont le poids est le *PMI* pour chaque paire dont le *PMI* est positif (ou supérieur à un seuil éventuellement).

Construction du graphe avec *word embeddings* La dernière méthode de construction de graphe que nous proposerons sera, bien sûr, de tirer à nouveau parti des modèles de langages. Il suffit alors de vectoriser tous les mots avec le modèle de language de notre choix, puis de relier par une arête tous les mots (ou plutôt les sommets qui leur correspondent) dont la similarité cosinus est supérieure à 0 (ou a un certain seuil) et

d'attribuer à cette arête le poids de leur similarité.

9.2 Sélection des mots clés

Tout comme pour la synthèse automatique, la méthode la plus naturelle est de simplement sélectionner les mots avec le meilleur score. Mais il est également possible de sélectionner les mots clés pour maximiser la diversité. Après avoir établi une liste de candidats pour les mots clés, on va vouloir sélectionner les mots les plus différents les uns des autres à l'intérieur de cet ensemble. Une solution pour cela serait de calculer les distances deux à deux des mots candidats, et de sélectionner les n mots qui minimisent la somme de leurs similarités.

Lors de l'utilisation d'algorithmes de détection de mots clés différents, on observe bien sûr des mots en commun, mais également des mots différents, sans pour autant qu'un des résultats soit clairement meilleur que l'autre. En particulier, si l'on compare des algorithmes qui utilisent des stratégies tout à fait différentes (méthodes statistique et méthode basée sur la vectorisation par exemple), les résultats peuvent être plutôt différents, et l'on voudrait parfois garder certains mots clés de chaque méthode. Une possibilité est donc simplement d'utiliser plusieurs méthodes, et d'agréger leurs résultats, en faisant l'union des deux, et éventuellement en utilisant un algorithme de diversification sur ces résultats ensuite.

9.3 Quelques résultats

Nous avons calculé les mots clés sur les trois mêmes textes que pour nos tests sur la synthèse et selon quatre méthodes :

— 1 : La méthode TFIDF décrite plus haut ;
— 2 : TextRank avec *PMI*, comme décrit plus haut, le *PMI* ayant été calculé sur un corpus de quelques centaines de documents, pas seulement les trois textes ;
— 3 : Une méthode "*baseline*" basée sur Bert et la vectorisation, qui sélectionne les mots dont le vecteur est le plus proche du vecteur du document ;

— 4 : TextRank avec la vectorisation , comme décrit plus haut.

Les résultats de cette expérience sont rapportés en annexe D. De manière similaire à la synthèse, il est compliqué de déterminer quelle est la meilleure méthode sur quelques exemples. Une première chose à noter, est que les méthodes qui n'utilisent pas les *embeddings* ressortent parfois des mots incorrects. C'est le cas par exemple du mot "intent" pour le document "tract", qui est issu d'une mauvaise lemmatisation du verbe "intenter". Comme supposé précédemment, les noms et adjectifs semblent être de meilleurs mots clés que les verbes qui, hors contexte, perdent souvent une part conséquente de leur sens. Une amélioration serait donc de ne sélectionner que les noms et adjectifs, ou bien d'étoffer la liste de *stopwords*, notamment avec de nombreux verbes. Par ailleurs, sur ces exemples, bien que la méthode TFIDF soit très simple et basique, elle nous semble produire des mots clés de très bonne qualité, ce qui soutient l'idée que la complexité ne fait pas la qualité. Finalement, les quatre méthodes produisent des mots clés pertinents, et sur ces quatres exemples, il est difficile de choisir une seule méthode meilleure sur tous les documents. Une solution serait tout simplement d'agréger les mots clés issus des différentes méthodes. En faisant cela, il faudra veiller à garder l'ordre d'importance, puisque les mots clés sont donnés par chaque algorithme du plus important au moins important. Cette agrégation permet d'obtenir des mots clés variés, et de très bonne qualité. Pour le monologue de l'inconstance, cette agrégation résulte en le nuage de mots de la figure 12.

Cette solution peut-être satisfaisante pour trouver les mots clés de quelques documents, mais elle pourrait sembler un peu trop complexe pour pouvoir être réellement utilisée, en particulier lorsqu'on dispose d'une dizaine de milliers de documents, comme c'est notre cas, voire beaucoup plus (bien que l'on puisse également vouloir extraire les mots clés à la demande, et dans ce cas, le temps d'éxécution, environ une seconde, est largement raisonnable). Une meilleure solution, dans l'absence de labels, serait d'effectuer des tests sur les tracts à plus grande échelle et de demander à des experts métiers d'évaluer les différentes méthodes selon ce qu'ils en attendent pour leur utilisation. Cela permettrait de ne garder qu'une ou deux méthodes, et de simplifier la tâche.

FIGURE 12 – Nuage de mots clés pour le texte "monologue", issu de Dom Juan. Les mots clés sont issus de l'agrégation de quatre méthodes différentes. La taille des mots indique leur importance.

10 Attribution de thème

Un autre élément qui serait très utile pour la transmission d'informations est le thème. En effet, en particulier lorsque la quantité de documents est élevée, ce qui est le cas dans notre cas pratique, connaître les thèmes des documents peut être crucial. Cela peut notamment permettre d'effectuer des analyses sur les thèmes les plus récurrents dans le corpus par exemple, ou sur un certain laps de temps. Cela est particulièrement pertinent dans le cadre des tracts syndicaux. Il est possible d'attribuer à chaque document un ou plusieurs thèmes. Cependant il s'agit bien de thèmes et non pas de mots clés, et leur nombre doit rester très faible.

Ce problème est souvent pris sous l'aspect de la classification supervisée. Encore une fois, nous voudrons nous passer de supervision, et nous proposerons deux algorithmes qui ne sont pas des algorithmes supervisés. Cependant, nous disposons tout de même d'un grand nombre de données déjà labellisées pour les thèmes dont nous pourrons éventuellement nous servir.

$$A_{ij} = \begin{cases} \text{PMI}(i,j) & i,j \text{ are words, PMI}(i,j) > 0 \\ \text{TF-IDF}_{ij} & i \text{ is document, } j \text{ is word} \\ 1 & i = j \\ 0 & \text{otherwise} \end{cases}$$

FIGURE 13 – Définition des poids des arêtes dans le graphe du corpus dans [61].

10.1 Méthode non-supervisée

Dans notre base de données, nous disposons de quelques milliers de tracts, mais qui peuvent être regroupés autour de quelques thèmes.

Nous allons poursuivre sur l'utilisation de graphes. L'idée ici sera de construire un graphe, dans lequel chaque document sera représenté par un sommet. Nous appliquerons ensuite sur ce graphe un algorithme de détection de communautés. Chaque communauté sera alors composée de documents sur un même thème.

Plus précisément, la construction du graphe suit la méthode proposée dans un papier sur la classification automatique [61]. Le corpus sera représenté sous forme de graphe hétérogène dont les sommets seront des documents et des mots. Comme dans les méthodes décrites précédemment, une arête représentera un lien entre deux éléments. Il pourrait y avoir plusieurs manières de définir ces liens mais le papier les défini par la relation de la figure 13, c'est à dire le *PPMI* entre les paires de mots, et le *TF-IDF* entre les paires mot-document, que nous avons vus précédemment.

Sur ce graphe nous pouvons appliquer l'algorithme de Louvain [7], une méthode pour détecter les communautés de manière hiérarchique. Cet algorithme possède de nombreux avantages, notamment son efficacité et sa rapidité d'éxécution. Le principe de l'algorithme est d'essayer de maximiser la modularité [17] à chaque étape, en agrégeant des noeuds à chaque itération (d'où la construction hiérarchique).

L'algorithme nous renverra donc une partition de l'ensemble des noeuds. Dans chaque partition, il peut y avoir tout type de noeuds, que ce soient des documents ou des mots. Chaque partition représente alors un sujet : Les documents appartiennent au même thème et les mots illustrent ce thème. Nous avons donc bien réussi à attribuer

17. mesure qui indique la qualité d'une partition

un thème à chaque document. Néanmoins, nous n'avons pas nommé les thèmes, nous savons seulement quel document appartient à chaque thème et quels mots le décrivent.

Nous pouvons imaginer deux solutions à cela :

— L'algorithme ne produira qu'un nombre raisonnable de communautés. Il est donc envisageable d'attribuer un nom au thème de chaque communauté manuellement, en regardant notamment les mots de la communauté, et éventuellement en consultant certains documents.
— Dans le cas où nous possèderions certaines données déjà labellisées, disons environ 50% du *dataset* (ce qui est notre cas), une solution serait d'attribuer à la communauté le thème majoritaire parmi les documents déjà labellisés dans la communauté.

10.2 Méthode semi-supervisée

Puisque nous possédons quelques données labelisées, il est possible d'utiliser directement la méthode de classification du papier dont nous venons de parler [61]. Nous gardons exactement la même construction du graphe, mais plutôt que d'utiliser un algorithme de détéction de communautés, nous allons utiliser un réseau convolutionnel de graphe pour classer les documents. L'idée est d'entraîner le *GCN* sur un graphe partiellement labellisé. Lors de l'entrainement, les caractéristiques des sommets seront propagées à travers le graphe, ce qui permettra ensuite de classifier les documents non labellisés. Une schématisation de l'algorithme issue du papier original est donnée en figure 14.

L'inconvénient de cette méthode par rapport à la précédente (surtout Louvain dans le cas où les thèmes sont labellisés manuellement pour chaque communauté) est qu'il est impossible de prendre en compte de nouveaux thèmes.

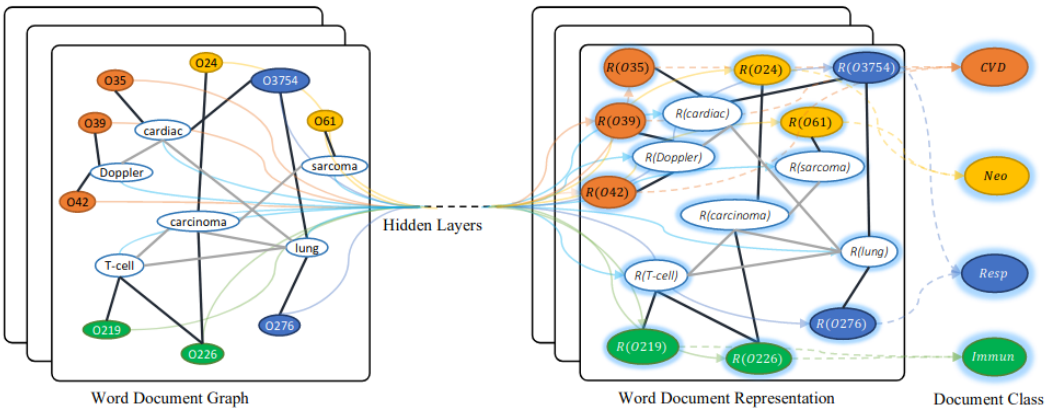


FIGURE 14 – Schématisation de l'algorithme *Text* GCN issue de [61]. Les ellipses blanches représentent les mots et les autres les documents. Un graphe est modélisé (à gauche) puis passé dans un réseau convolutionnel pour propager les caractéristiques et apprendre une représentation qui permettra d'attribuer aux documents non labellisés une classe (à droite).

11 Évaluation

Pour chacune des tâches que nous avons proposées, il existe de nombreuses méthodes différentes, sur lesquelles il est possible d'appliquer de nombreuses variantes. Il devient donc nécessaire d'avoir des méthodes d'évaluation et de comparaison pour mesurer les performances des algorithmes utilisés.

11.1 Les thèmes

Concernant l'attribution de thème, il est possible de trouver assez facilement des jeux de données labellisées (des documents avec leur classe, souvent des articles), et d'utiliser une des mesures classiques de performance. En effet, il est légitime considérer que sur un grand *dataset*, pour l'attribution d'un thème, la subjectivité est assez minime et il est possible de facilement considérer que si l'algorithme trouve le bon thème, il a juste, sinon, il a faux. Dans le cadre d'un algorithme de classification binaire, il est souvent utile de calculer la matrice de confusion qu'on retrouve en table 3.

À partir de ces valeurs, nous pourrons calculer plusieurs métriques d'évaluation. Notons que pour évaluer une tâche de classification, il est important de s'assurer que les classes

	Positif	Négatif
Positif	Vrais positifs (TP)	Faux positifs (FP)
Négatifs	Faux négatifs (FN)	Vrais négatifs (TN)

TABLE 3 – Matrice de Confusion. Les lignes représentent les valeurs prédites et les colonnes représentent les vraies valeurs.

soient équilibrées. En effet, s'il existe une classe fortement majoritaire (un thème fortement dominant) et que le classifieur prédit toujours cette classe, l'évaluation pourrait être bonne, alors que le classifieur ne l'est en réalité pas.

Lorsque l'on effectue de la classification binaire, les métriques les plus courantes sont les suivantes :

— *Accuracy* : (TP+TN)/(TP+TN+FP+FN) – Quelle proportion des échantillons (positifs et negatifs) sont correctement classifiés ?
— *Precision* : TP / (TP + FP) – Quelle proportion des prédictions positives sont réellement positives ?
— *Recall* : TP / (TP + FN) – Quelle proportion des positives sont correctement classifiées ?
— *F1 Score* : 2*(Recall * Precision) / (Recall + Precision) – Permet de prendre en compte à la fois le *recall* et la précision (il est souvent compliqué d'optimiser les deux en même temps).

La métrique choisie dépend surtout de l'utilisation du modèle évalue (si le contexte est celui de la sécurité par exemple, on ne souhaitera pas de faux positifs). Rappelons que les métriques que nous venons de décrire sont utilisées dans le cadre de classification binaire. Dans un problème multi-classe comme le notre (autant de classes que de thèmes), pour N classes, on calcule la précision et le *recall* pour chaque classe (en considérant que la classe positive est la classe en question, et la classe négative tout le reste).Nous disposons alors de N précisions et N *recall*. Il existe encore plusieurs manières d'aggréger ces scores. En effet, encore une fois, une classe majoritaire peut complètement déséquilibrer la métrique. Une possibilité peut donc être de pondérer les résultats par le nombre d'exemples de la classe.

11.2 Les mots clés

Concernant la détection de mots clés, la méthode d'évaluation sera forcément un peu plus subjective. Très souvent les méthodes sont évaluées en utilisant des *datasets* avec mots clés attribués manuellement. C'est ce qui a été fait notamment pour l'évaluation de l'extraction de mots clés du papier *TextRank* [40]. Pour un texte donné, la proportion de mots clés assignés manuellement qui ont été trouvés par l'algorithme est le rappel (*recall*), et la proportion de mots clés proposés qui sont corrects est la précision. Puisque la précision et le rappel sont considérés comme aussi importants l'un que l'autre, le score *F1* est ensuite utilisé.

Nous pouvons déjà noter certains problèmes concernant cette évaluation. Tout d'abord, l'attribution des mots clés est forcément très subjective (encore plus qu'un thème par exemple). D'autant que pour la plupart des jeux de données, les personnes les labellisant n'ont pas de nombre de mots clés précis à trouver, alors que dans la plupart des cas, l'algorithme en a un (il est donc impossible d'avoir un rappel à 100% en général). En outre, il est évident que la personne qui va attribuer des mots clés manuellement à un texte joue beaucoup dans la liste de mots clés choisis. Il semble alors tout à fait légitime de se poser la question suivante : Pour un même jeu de données sur lequel on va tester n modèles différents, si par exemple 4 équipes différentes labellisent chacune toutes les données (en donnant les mots clés qu'elles veulent, et le nombre de mots clés qu'elles veulent), créant ansi un *test set* chacune, et que les n modèles sont ensuite évalués sur les 4 *test sets*, aurons nous toujours le même classement ? Le même "meilleur" modèle ?

Nous avons demandé à plusieurs personnes de proposer une liste de mots clés (*uni-grams*) pour 3 textes différent. Nous avons testé ensuite plusieurs algorithmes sur ces mêmes textes, et avons évalué leurs résultats par rapport aux différentes références dont nous disposons. Les évaluateurs n'ont eu que la vague consigne de trouver les mots clés (*uni-grams*), sans précision sur le nombre ou sur le type. Mais le fait qu'il n'y ait pas de convention est assez représentatif de la manière dont sont labellisés les jeux de données en général.

Les résultats de cette expérience pour le monologue de Dom Juan sont donnés dans le tableau en figure 15. Le détail des mots clés donnés par nos évaluateurs est rapporté

en annexe F, ainsi que les résulats pour les deux autres textes. Les algorithmes évalués sont les mêmes que dans la section 9.3 et dont les résultats sont en annexe D.

	eval_A	eval_C	eval_E	eval_L	eval_M1	eval_M2	eval_Z
TFIDF	0.2	0.266667	0.6	0.375	0.25	0.5	0.266667
Comp. Vecteurs	0.1	0.133333	0.3	0.125	0	0.3	0.133333
TextRank PMI	0.2	0.4	0.3	0.25	0.375	0.2	0.133333
TextRank Embed.	0.1	0.133333	0.1	0.125	0.125	0.1	0

FIGURE 15 – Résultats des évaluations de quatres méthodes d'extraction de mots clés différentes, avec le score *F1*, en comparant à différents ensemble de mots clés référence.

Comme nous l'avions supposé, le meilleur modèle varie beaucoup selon les évaluateurs, et ces derniers ont choisi une quantité de mots clés assez variable (entre 3 et 14 selon les textes). Si peu de tests sont bien sûr insuffisants pour déterminer le meilleur modèle, et puisqu'ils n'ont été effectués que sur trois textes, pour sept évaluateurs, il faut prendre les résultats avec précaution. Pour le monologue de Dom Juan cependant, le meilleur modèle est parfois celui basé sur *TFIDF*, parfois celui basé sur *TextRank* et le *PMI*. Mais ce dernier modèle a également donné de très mauvais résultats sur certains autres textes. Bien que les personnes ayant produit les mots clés de référence ne soient pas des experts, ce test montre, sur notre corpus, la grande variabilité des résultats selon la référence choisie.

De cette expérience, nous pouvons tirer deux conclusions : tout d'abord, il semble nécéssaire de trouver une méthode d'évaluation qui atténuerait le biais induit par la subjectivité des évaluateurs. Une solution assez simple à cela, serait d'évaluer les algorithmes sur plusieurs ensembles référence. Pour mettre en place cette solution, il serait possible ou bien de calculer un score pour chaque ensemble référence, puis d'en faire la moyenne, ou bien de construire un ensemble référence agrégé, par exemple en ne prenant que les mots qui apparaissent dans au moins la moitié des ensembles référence. Enfin, cette expérience souligne également l'intérêt de fusionner plusieurs méthodes, puisque leurs performances varient selon les textes, et les évaluateurs, et que nous aimerions tirer parti des avantages de chacun des modèles. De plus, certains modèles d'extraction de mots clés ont des scores identiques, mais pourtant des listes de mots

différentes.

11.3 La synthèse

L'évaluation de la synthèse automatique est encore plus complexe : Faire le résumé d'un document semble encore plus subjectif que trouver des mots clés. Un bon résumé doit être concis, cohérent, pas redondant, et surtout contenir les informations les plus importantes du texte. La manière la plus simple d'évaluer un résumé, est certainement l'évaluation humaine (des juges évaluent un résumé selon les critères cités précédemment par exemple), mais il est évident que pour comparer des modèles à grande échelle, cette solution serait beaucoup trop chère, et on préfèrerait un modèle d'évaluation facilement reproduisible.

Il n'existe pas de résumé parfait ni de meilleur résumé (particulièrement s'il est question de résumés abstractifs). Il peut donc être compliqué de trouver une métrique pour quantifier numériquement la pertinence d'un résumé généré automatiquement. Les métriques les plus utilisées possèdent de nombreux défauts, et ce n'est que depuis récemment que l'utilisation de ces dernières est remise en question dans la littérature scientifique.

11.3.1 Méthodes d'évaluation classiques

La plupart des méthodes existantes impliquent un "résumé référence", souvent appelé *golden summary*, qui serait un résumé écrit par un humain. Les méthodes classiques consistent alors simplement à comparer le résumé généré automatiquement, et le résumé humain. Cette méthode est certainement la plus naturelle, mais tout comme pour les mots clés, il n'est pas évident qu'un jeu test constitué de *golden summaries* écrit par des personnes différentes donnerait les mêmes résultats et les même performances des modèles. D'autant qu'avec les méthodes les plus couramment utilisées, un simple mot à la place d'un autre peut changer le score, et les résultats sont souvent très sérrés, comme on peut le constater dans la figure 16, dans laquelle on retrouve les résultats de l'algorithme *TextRank*. Pour l'évaluation de *TextRank*, les résultats sont notamment

System	ROUGE score – Ngram(1,1)		
	basic (a)	stemmed (b)	stemmed no-stopwords (c)
S27	0.4814	0.5011	0.4405
S31	0.4715	0.4914	0.4160
TextRank	**0.4708**	**0.4904**	**0.4229**
S28	0.4703	0.4890	0.4346
S21	0.4683	0.4869	0.4222
Baseline	*0.4599*	*0.4779*	*0.4162*
S29	0.4502	0.4681	0.4019

FIGURE 16 – Résultats de l'algorithme *TextRank* évalué sur un jeu de 500 documents avec la méthode *ROUGE* sur des résumés manuels. Les résultats sont mis en comparaison avec les 5 meilleurs systèmes ayant auparavant été évalués sur le même jeu de données, aisni qu'une *baseline*.

comparés à une *baseline*, celle de prendre les 100 premiers mots d'un texte comme résumé (il est courant de considérer que beaucoup d'informations importantes d'un texte se trouvent au début). Les résultats sont aussi très serrés avec la *baseline*, qui est même meilleure que certains modèles.

Précisons que les modèles de synthèse extractive sont bien évalués sur des résumés humains abstraits.

Les performances des modèles dépendent également beaucoup du jeu de données sur lequel ils sont évalués. Certaines méthodes sont meilleures seulement sur les textes longs, sur des textes courts, sur des données très similaires à leurs données d'entrainement, ou sur des textes concernant des sujets génériques (sans jargon précis).

La méthode *ROUGE* [33], publiée dans un papier de 2004, est la méthode la plus utilisée pour l'évaluation de la synthèse automatique, et la méthode qui est encore utilisée dans les derniers papiers qui paraissent (c'est notamment la méthode d'évaluation principale de *Pegasus*, dont nous avons parlé précédemment). Cette méthode est inspirée du rappel (*recall*).

La méthode *ROUGE* se décline en plusieurs variantes :

— *ROUGE-N* : *ROUGE-1* / *ROUGE-2*. La méthode *ROUGE-N* compare les *N*-

grams[18] entre le texte de référence et le texte à évaluer. Cette métrique consiste à mesurer le nombre de *N-grams* communs entre les deux textes par rapport au nombre de *N-grams* dans le texte de référence. Le plus souvent, on utilise *ROUGE-1*, qui se focalise donc sur les *uni-grams*, et *ROUGE-2*, qui se focalise sur les *bi-grams*. Plus formellement, *ROUGE-N* se calcule comme suit :

$$ROUGE - N = \frac{\sum_{S \in \{ReferenceSummaries\}} \sum_{gram_n \in S} Count_{match}(gram_n)}{\sum_{S \in \{ReferenceSummaries\}} \sum_{gram_n \in S} Count(gram_n)} \quad (8)$$

— *ROUGE-L* : Dans cette méthode, plutôt que de compter le nombre de *N-grams* en commun, on va calculer la taille de la plus grande sous-séquence commune.

La méthode *ROUGE-1* est la plus couramment utilisée. Il existe d'autres méthodes parfois utilisées, mais tout de même bien moins que *ROUGE* qui reste la méthode de base. Nous ne développerons donc pas outre-mesure ces méthodes mais pour en présenter brièvement :

— La méthode *BLEU* (*Bilingual Evaluation Understudy*) [47] a d'abord été pensée pour la traduction automatique mais est également utilisée pour la synthèse automatique. Cette méthode compare deux textes et renvoie une valeur entre 0 et 1, 0 indiquant aucune correspondance, et 1 une parfaite correspondance. Ce score évalue plutôt la précision du modèle, puisqu'il est calculé en comparant le nombre de mots du résumé automatique qui font également partie du résumé de référence, par rapport au nombre total de mots du résumé automatique, en ajoutant quelques ajustements pour prendre en compte certains cas extrêmes (notamment des cas où des mots sont répétés plusieurs fois).

— La méthode *METEOR* (*Metric for Evaluation of Translation with Explicit ORdering*) [3] est également une méthode conçue par évaluer la traduction automatique, et est supposée être une amélioration de *BLEU*. Elle fait la somme pondérée entre la precision et le rappel des *unigrams*, tout en prenant en compte les synonymes ou le *stemming* notamment, ainsi qu'une mesure d'à quel point les mots correspondants entre les deux textes sont dans le bon ordre.

Ci-après, nous discutons des défauts de la méthode *ROUGE*, dans le but de questionner

18. Un *N-gram(n,m)* est une suite d'entre n et m mots consécutifs dans un texte.

Référence A	"Une femme accouche d'un bébé de 6 kilos."
Résumé A1	"Un bébé accouche d'une femme de 6 kilos."
Résumé A2	"Une femme accouche de 6 kilos d'un bébé."
Résumé A3	"De kilos d'un femme bébé une accouche 6."
Référence B	"Le vieillard répond au policier."
Résumé B1	"Le policier répond au vieillard."

TABLE 4 – Propositions de résumés comportant tous les mêmes mots que le résumé de référence.

sa légitimité en tant que méthode privilégiée pour évaluer les synthèses automatiques depuis presque 20 ans. Cette place est, dernièrement, de plus en plus questionnée dans la littérature scientifique. C'est le cas notamment dans un papier paru en 2021, "*A Comparison of Methods for the Evaluation of Text Summarization Techniques*" [4], dans lequel les auteurs, après plusieurs expériences et tests statistiques, concluent que *ROUGE* ne serait pas une bonne méthode pour évaluer des algorithmes de synthèse automatique, et même que un bon score *ROUGE* n'est pas synonyme de bon résumé.

Dépendance au résumé référence En 2004 déjà, certains papiers soulignaient les effets des variations humaines dans l'évaluation de la synthèse automatique [25]. Comme évoqué plus haut, le score *ROUGE* dépend fortement du résumé de référence et donc de l'humain qui l'a écrit. Il n'existe, en général, aucun accord entre annotateurs : Différents juges produiront, à partir d'une même source, différents résumés, probablement de tailles différentes, mais dont la qualité n'est pas forcément très différente. La métrique *ROUGE* aurait alors déjà plus de sens, et ce problème serait amoindri, si le résumé automatique était comparé à plusieurs résumés de référence, et en agrégeant les résultats. Mais la plupart des jeux de données pour la synthèse automatique ne contiennent qu'une seule référence par texte. C'est le cas notamment de deux jeux très utilisés : *CNN Daily Mail* et *Gigaword*. Ceci dit, même en collectant un grand nombre de résumés de référence, il sera toujours possible d'en écrire/générer un qui soit tout aussi informatif mais encore différent. Il est donc impossible de trouver un ensemble de résumés de référence qui soit complet. La grande variabilité de ce mode d'évaluation rend d'autant plus important de mesurer les informations contenues plutôt que les mots en eux-mêmes.

Informations contenues et compression Il serait pertinent de mesurer le degré d'informations contenues dans le résumé, par rapport à celui contenu dans le texte d'origine, ou éventuellement par rapport au résumé référence. Il existe de très nombreuses manières de dire la même chose, et *ROUGE* ne le prend pas du tout en compte puisqu'elle ne compte que les mots en commun. Cela pénalise de manière évidente les méthodes abstractives qui vont souvent utiliser la reformulation, mais également les méthodes extractives : il est possible de trouver dans un texte deux phrases qui signifient la même chose, et dont le choix de l'une ou l'autre est équivalent concernant la qualité du résumé. Pour savoir si un résumé est souvent informatif, on peut se poser la question suivante : pour chaque information du texte d'entrée, cette information est elle retransmise dans le résumé ? Dans le cas extractif, en comptant les mots communs entre le résumé automatique, et le résumé référence, la méthode *ROUGE* prend à un degré dérivé les informations contenues, mais cela ne semble pas suffisant. Il serait donc pertinent de pouvoir évaluer cet aspect plus précisément. L'information contenue est à mettre en balance avec la compression : plus un texte est compressé, moins il peut contenir d'informations. Pour évaluer pleinement un modèle, il serait également pertinent de l'évaluer avec plusieurs taux de compression. Il est possible de mesurer *ROUGE* à différent niveaux de compression, mais *ROUGE* ne prend pas ce paramètre en compte naturellement. Entre deux résumés tout aussi informatifs, il serait pertinent de valoriser le plus court.

Cohérence et grammaire Un bon résumé doit être cohérent et lisible. Comme nous avons pu l'évoquer précédemment, un défaut des résumés extractifs est que certaines phrases peuvent manquer de contexte, notamment lors de l'utilisation de mots qui font référence à une autre phrase, ce qui peut rendre le texte du résumé incohérent. La syntaxe et la grammaire ne sont pas non plus évaluées. Cet aspect concerne principalement la synthèse abstractive, puisque la synthèse extractive reprend des phrases a priori déjà correctes. Cependant, la juxtaposition de phrases peut être, dans son ensemble, incohérente. Un modèle abstractif qui contient les bonnes informations mais qui est grammaticalement et syntaxiquement tout à fait incorrect est il meilleur qu'un modèle parfaitement correct mais moins informatif ? Ces aspects ne sont évalués d'aucune manière avec la méthode *ROUGE*. Dans la table 4, le résumé A2 contient la même

information que la référence, mais la phrase est syntaxiquement incorrecte. Pourtant, comme les deux phrases contiennent strictement les mêmes mots, le score de A2 sera très bon. Le résumé A3 n'a tout simplement aucun sens. Cependant, il contient également les mêmes mots et aura également le même score.

Consistance Dans un résumé, il est primordial que les informations soient les mêmes que dans le texte original. Encore une fois, cet aspect n'est pas pris en compte par *ROUGE*. Pour la référence B de la table 4, le résumé B1 contient les mêmes mots mais à un sens opposé. Cette information serait trompeuse dans un résumé. Ce phénomène est celui que nous avons appelé "l'effet *fake news*" dans une section précédente. L'aspect de la consistance est particulièrement important pour la synthèse abstractive mais il l'est aussi pour la synthèse extractive. En effet, une phrase sortie de son contexte peut avoir un sens très différent que celui qu'elle est censée avoir dans le texte initial.

En prenant tous ces points en compte, il semble compliqué de suivre précisément l'évolution du domaine de la synthèse automatique (par manque de bons outils de comparaison), et également compliqué de juger qu'une nouvelle méthode sera *state-of-the-art* en utilisant ces métriques. La plupart des nouveaux papiers de recherche qui sortent tentent d'évaluer leur modèle manuellement, mais il reste compliqué de comparer ces expérimentations à travers différents papiers.

Aujourd'hui, il semble sans doute plus important de concentrer la recherche sur le développement d'une méthode d'évaluation peu coûteuse, efficace (dans le sens où elle mesure les vraies qualités d'un résumé), et répétable. Le fait qu'il n'existe pas de consensus sur un bon protocole d'évaluation empêche la comparaison et l'évaluation précise et efficace des modèles et est probablement un frein au progrès dans le domaine de la synthèse automatique.

11.4 Pistes pour une méthode d'évaluation

Puisque la méthode *ROUGE* est insuffisante pour évaluer correctement un résumé généré automatiquement, nous tâcherons de donner quelques pistes pour proposer une

méthode plus précise.

11.4.1 Quels critères prendre en compte ?

Il convient d'abord de définir précisément ce que nous souhaitons évaluer. Evidemment, il faudra notamment prendre en compte les quelques critères que nous venons d'énumérer dans la section précédente. Dans *SummEval* [20], un papier qui propose une étude sur les méthodes d'évaluation actuelles, sont définis quatre critères pour que des humains évaluent manuellement des résumés générés automatiquement :

— La **pertinence** mesure à quel point le résumé comprend les points clés de l'article. Il s'agit de considérer si tous les aspects importants (et seulement ceux-ci) sont comtenus dans le résumé. [19]

— La **consistance** mesure si les faits rapportés dans le résumés sont consistants avec les faits de l'article original. Il s'agit de considérer si le résumé reproduit tous les faits précisément, et ne compose pas de fausses informations. [20]

— La **fluidité** mesure la qualité des phrases individuelles, si elles sont bien écrites et grammaticalement correctes. Il s'agit de considérer la qualité des phrases individuellement. [21]

— La **cohérence** mesure la qualité de toutes les phrases collectivement, si elles vont bien ensemble et semblent naturelles. Il s'agit de considérer la qualité du résumé dans son ensemble. [22]

Par ailleurs, la synthèse étant une épreuve dans de nombreux concours, il existe également depuis longtemps des critères précis déterminant ce qui fait un bon résumé. Certains de ces critères sont reproduits en annexe F. La plupart de ces critères rejoignent ceux énoncés ci-dessus, si ce n'est que la non-reformulation du texte est pénalisée (mais ça n'est pas intéressant d'utiliser ce critère dans notre cadre) et que l'enchainement des

19. "The rating measures how well the summary captures the key points of the article"
20. "The rating measures whether the facts in the summary are consistent with the facts in the original article. Consider whether the summary does reproduce all facts accurately and does not make up untrue infomartion."
21. "This rating measures the quality of individual sentences, are they well-written and grammatically correct. Consider the quality of individual sentences."
22. "The rating measures the quality of all sentences collectively, to the fit together and sound naturally. Consider the quality of the summary as a whole."

idées doit être ordonné et pertinent. En effet, il est demandé de garder la même structure logique dans le résumé que dans le texte. Ce critère pourrait être intéréssant à ajouter à l'évaluation de la synthèse automatique, mais ce critère semble plutôt secondaire (par rapport notamment à la pertinence), et pourrait plutôt servir à départager deux très bons résumés.

Deux points nous paraissent particulièrement importants à prendre en compte pour l'élaboration d'une méthode d'évaluation. Le premier étant qu'il faut se détacher de l'utilisation d'un *golden summary*. Lorsqu'on entraîne un modèle, il est habituel de lui montrer "la bonne solution" sur de nombreux exemples, il était donc naturel d'utiliser cette méthode pour l'évaluation de la synthèse automatique. Cependant, puisqu'il n'existe pas d'unique bonne solution ici, il est désormais évident qu'il faudra passer par d'autres moyens. Un deuxième point qui semble important est que, si une référence humaine est nécéssaire, il en faudrait plusieurs par texte, afin d'atténuer le biais.

11.4.2 Evaluer la pertinence - Comparaison à un ensemble d'idées

Afin d'évaluer la pertinence d'un résumé, une première idée serait de comparer le résumé à un ensemble d'idées, plutôt qu'à un résumé référence. Dans ce cas, plutôt que de savoir si le résumé ressemble au jeu d'idées, il faudrait plutôt chercher à savoir si les idées sont, ou non, incluses dans le résumé.

Cependant, on peut voir dans l'élaboration de cet ensemble d'idées, les mêmes problèmes de sensibilité à l'humain qu'auparavant. Une solution à cela pourrait être de demander à plusieurs évaluateurs de construire un ensemble d'idées, puis de prendre soit l'intersection, soit toutes les idées présentes dans plus de la moitié (proportion à affiner) des jeux d'idées. Pour faire cette agrégation, il faudra être capable de regrouper différentes formulations d'une même idée. Un algorithme de *clustering* utilisant la vectorisation de ces idées pourrait permettre de faire cela.

En supposant qu'il soit possible d'avoir accès à un ensemble d'idées fidèle au texte et fiable, il faudra ensuite savoir si ces idées sont contenues dans le résumé. Pour cela, on peut déjà se servir du fait que chaque idée ait été formulée plusieurs fois, de différentes

manières. Une possibilité serait alors de soit chercher l'idée dans le texte sous différentes formes, soit agréger les différentes formes d'une même idée en une seule. Dans le cadre d'un algorithme de *clustering*, il serait par exemple possible de prendre le vecteur du centroïde pour représenter le *cluster* d'une idée. Quelle que soit la méthode choisie, pour tester si une idée est présente dans le résumé, il serait ensuite envisageable de comparer l'idée à la vectorisation du document, ou a chacune des phrases du document. Au dessus d'un certain seuil de similarité, on considèrerait que l'idée est comprise dans le résumé. Une autre solution serait d'utiliser des algorithmes de *Question-Answering*, et de questionner directement le texte sur ces idées. Ensuite le résumé est évalué sur chaque idée par un label comme présent/incorrect/manquant.

Une autre possibilité serait de demander directement aux superviseurs d'établir un ensemble de questions (donnant chacune une idée) pour tester ensuite le *Question Answering*. Demander à des superviseurs de construire un ensemble d'idées/de questions ne semble pas beaucoup plus coûteux que de demander d'écrire un résumé abstractif, mais il faudrait évaluer ce point, notamment en prenant en compte le temps requis pour cette tâche. Par ailleurs, une autre possibilité serait d'extraire le jeu d'idées directement depuis les résumés déjà existants.

11.4.3 Évaluer la consistance - Poser des questions

Pour évaluer la consistance, l'idée est de vérifier si les informations sont bien les mêmes dans le texte inital et le résumé. Il ne suffit pas de savoir si ses sujets sont présents, mais plutôt, si les informations présentent dans les deux textes vont bien dans le même sens. Concrètement, si on pose une question, les deux textes devraient nous renvoyer la même réponse. C'est l'intuition développée dans [15], qui propose un protocole basé encore une fois sur le *Question-Answering* pour vérifier des informations dans un texte et son résumé. À partir du résumé, plusieurs questions sont générées. Les questions sont ensuite posées aux deux textes, et un score de qualité est attribué au résumé. Cette méthode d'évaluation semble très prometteuse. Ce score de consistance serait donc très pertinent à agréger dans un score de qualité global. Lors d'un résumé extractif, cet aspect est assez secondaire cependant, puisque les phrases sont, par définition, directement extraites du document initial et donc a priori consistantes, sauf dans de

très rares cas où le manque de contexte donnerait un sens différent.

11.4.4 Un classement de phrases pour le résumé extractif

Lorsque de la construction d'un résumé extractif, très souvent on classe les phrases, puis on choisit les n premières. Une méthode d'évaluation mais qui ne fonctionne qu'avec la synthèse extractive, serait d'évaluer le classement. Pour cela, les superviseurs donneraient chacun leur classement de phrases. Ce classement est, certes, subjectif, mais on peut supposer que pour les phrases les plus importantes, les superviseurs trouveraient un accord. Une fois que chacun a donné son classement, il faudrait agréger les résultats pour obtenir un classement unique. Cette agrégation pourrait être faite selon un vote de Condorcet.

Cette méthode n'est applicable qu'aux résumés extractifs, ce qui est un gros inconvénient si on veut comparer entre elles des méthodes abstractives et extractives. Cependant il est également pertinent de vouloir seulement comparer des méthodes extractives.

12 Proposition d'une interface pour l'aide à la transmission de sens

Nous avons étudié des méthodes pour calculer des résumés, des mots clés, et des thèmes, pour aider à une tâche de transmission de sens. Afin qu'un humain puisse tirer parti au mieux de ces informations, il faut réléchir également à une manière de les présenter.

12.1 Représenter les phrases en contexte

Comme nous avons pu l'évoquer précédemment, le manque de contexte est l'un des défauts principaux de la synthèse extractive, notamment par rapport à la synthèse abstractive. Pour palier ce problème, nous avons pensé qu'il serait pertinent de représenter les phrases sélectionnées dans leur texte. Nous avons donc décidé de surligner les phrases directement dans le tract. Cela a un double avantage : du point de vue de la synthèse

automatique, si une phrase manque de contexte, il est très simple de consulter la phrase précédente. De plus, avoir les phrases importantes qui ressortent permet au lecteur de "lire" le texte en le survolant, et en ne s'attardant plus seulement sur les phrases surlignées. Le second avantage est qu'ainsi, le lecteur ne perd pas toutes les informations données par la mise en page du tract, qui est très riche.

Afin de surligner les phrases, il a fallu, après avoir produit le résumé extractif, retrouver ces phrases directement dans le *PDF*. Cette étape est simple, mais il faut cependant prendre en compte un détail : Parfois, certaines phrases ne sont pas retrouvées dans le document *PDF*. Cela peut être du soit au fait que la phrase était coupée entre deux pages (et notre algorithme lit le document page par page) soit au fait que la phrase contenait une ligature[23], et qu'une lettre manquait donc dans la phrase extraite par le modèle. Pour résoudre ce problème, nous avons appliqué une recherche supplémentaire sur les phrases du *PDF* en comparant les phrases de ce dernier avec la phrase du résumé à retrouver avec la distance de Levenshtein.

12.2 L'application

Nous avons développé une interface, que l'on peut voir en figure 17, regroupant ces informations calculées, pour aider la tâche de transmission de sens.

Si aucun filtre n'est sélectionné, un nuage de mots avec des mots clés concernant l'ensemble des documents ainsi qu'un graphique montrant le nombre de tracts publiés par mois et par thème s'affichent.

Grâce à l'application, il est possible de filtrer la base de données selon différents critères. En annexe G, se trouve une capture d'écran de l'application avec les résultats selon un filtre particulier. Lorsqu'un filtre est sélectionné, l'application affiche le nuage de mots et le graphique mis à jour selon le corpus qui répond au filtre. Juste en dessous, on retrouve d'un côté les résumés extractifs hors contexte pour chacun des tracts du filtre.

23. Une ligature dans un texte est la fusion de deux ou trois lettres. Il existe des ligatures linguistiques, c'est le cas dans "sœur" ou "œuf", mais aussi des ligatures esthétiques, qui lient par exemple le f et le i de "finance". Les liagtures sont parfois faites automatiquement dans certains éditeurs de texte selon la police choisie, et sont mal lues lors de l'importation du texte depuis un *PDF*, ce qui résulte en une lettre manquante.

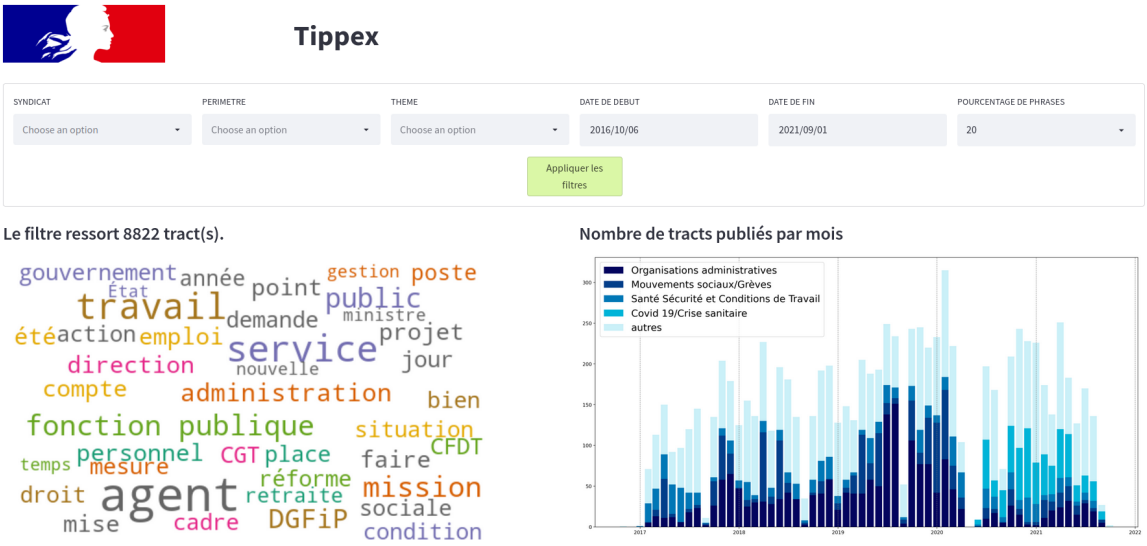


FIGURE 17 – Interface de l'application développée pour l'aide à la transmission de sens.

Cela permet une vision rapide des phrases sélectionnées pour chaque tract. De l'autre côté, sont présents des boutons cliquables pour télécharger une archive contenant tous les tracts surlignés d'un coup, ou bien télécharger les tracts un à un.

Quatrième partie

Conclusion

Apports Nous avons proposé trois méthodes complémentaires pour extraire du sens d'un corpus de documents, en tâchant de prendre au mieux en compte les réalités d'une utilisation pratique, dans un cadre avec peu de ressources. L'utilisation jointe de la synthèse, des mots clés, et du thème, permet d'avoir une vision globale et rapide d'un document, afin d'en comprendre rapidement les enjeux. Par ailleurs, posséder ces informations pour chaque document au sein d'un corpus pourra permettre de créer des visualisations pertinentes sur des sous ensembles (par exemple un certain nombre de documents séléctionnés selon divers filtres) de notre jeu de données et ainsi accéder à de nouvelles informations, entre autres les fréquences d'apparitions de thèmes ou mots clés. Nous avons d'ailleurs exploité certaines de ces possibilités dans l'application développée pour notre problème d'aide à la transmission de sens. En outre, de nombreuses autres utilisations et visualisations sont possibles à partir de ces ressources. Nous avons tâché de tirer parti de plusieurs méthodes d'extraction de mots clés, afin de profiter des qualités différentes de chacune. En particulier, la vectorisation de mots de jargons est assez peu efficace, c'est pourquoi il était pertinent de mixer des méthodes basées sur les *embeddings* et des méthodes basées sur la fréquence, pour prendre en compte tous les types de mots clés. Le fait de coupler l'extraction de mots et phrases clés à un algorithme pour diminuer la redondance (*MMR*) nous permet également de créer une liste de mots et un résumé plus efficaces. La méthode proposée pour l'extraction de phrase et mots clés, c'est à dire *TextRank* adapté aux *embeddings*, s'appuie sur l'état de l'art des modèles de language et pourra s'adapter dans le temps à de nouveaux modèles de language peut-être plus performants, ou éventuellement à d'autres modèles de languages dans d'autres langues, ou entrainés sur un domaine particulier. De plus, que ce soit pour la synthèse, pour les mots clés ou le thème, nous avons toujours proposé également une solution qui permette de trouver de bons résultats, indépendamment du domaine d'application, ou de la langue du texte, en utilisant par exemple la co-occurence et la fréquence des mots, notamment pour *PMI* et *TF-IDF*. Pour chacune

de nos tâches, nous avons dû trouver une manière de nous détacher de la supervision, notamment dans le cas de l'attribution de thèmes, dont les algorithmes efficaces sont souvent supervisés.

Finalement, nous avons créé notre propre *pipeline*, en décomposant les étapes classiques des algorithmes d'extraction de sens, pour sélectionner à chaque étape le modèle le plus pertinent, tout en l'adaptant à notre situation, lorsque cela était possible.

Enfin, nous avons pris le temps de nous questionner à propos des méthodes d'évaluation utilisées, notamment concernant la synthèse automatique. En effet, ces méthodes ne sont que peu remises en question, mais ne nous semblent pas être les plus pertinentes pour évaluer ce type de tâche. Dans une utilisation pratique, sur un jeu de données réel, il est important de pouvoir évaluer son modèle, et les résultats des mesures d'évaluation classiques, telles que *ROUGE*, ne nous permettaient pas de bien évaluer les différents modèles que nous avons testé. Pour justifier ce propos, nous avons développé plusieurs arguments sur les défauts de *ROUGE*. Suite à cela, nous avons apporté plusieurs pistes de réflexion concernant ce qui pourrait être une bonne manière d'évaluer un résumé automatique, et qui pourrait permettre de comparer efficacement deux algorithmes de synthèse automatique.

Limites et évolutions Une évolution naturelle de notre travail serait d'utiliser un algorithme de synthèse abstractive. Le résumé abstractif pourrait être utilisé pour remplacer les résumés extractifs hors contextes, mais pourquoi pas également en complément des phrases du résumé extractif surlignées dans le document original. Cependant, il faudrait, bien sûr, prendre en compte tous les défauts que nous avons pu évoquer pour cette méthode, et qui ont en partie justifié notre choix d'utiliser la synthèse extractive. L'utilisation d'un modèle de synthèse abstractive pourrait permettre de régler le problème du manque de contexte de la synthèse extractive. Créer un modèle abstractif vraiment efficace nécéssiterait de ré-entrainer un modèle sur nos propres données, puisque celles-ci comportent beaucoup de jargon, acronymes, et termes spécifiques, ce qui aboutit à d'assez mauvaises performances des modèles pré-entrainés classiques. Par ailleurs, il existe encore peu de modèles pré-entrainés sur des données en français, et ces derniers ne sont encore pas aussi performants que sur des données en anglais. Enfin,

il serait primordial de tester la consistance des résumés abstractifs produits, puisqu'il ne serait pas acceptable de donner de fausses informations. Pour cela, il est possible de vérifier les faits en utilisant un modèle de *Question-Answering*, et en posant les mêmes questions au résumé et au texte original, comme nous avons pu l'évoquer plus haut.

Une des limites de notre *pipeline* d'extraction de sens est que nous ne prenons en compte, à aucun moment, la mise en page des textes, ou les images. Ces éléments comportent, comme le texte, beaucoup d'informations, et il serait pertinent de les utiliser. À cette fin, une évolution pourrait être d'ajouter à notre résumé, au bon endroit, une phrase indiquant la description de l'image. Cela nécéssiterait un tout autre travail sur un modèle capable de décrire une image (tâche d'*Image Captioning*).

Une seconde évolution naturelle est la synthèse multi-documents. Cet élémént avait en effet été proprosé lorsque nous avions demandé à notre entourage ce qui pourrait les aider pour une tâche de transmission de sens.

Une autre évolution, pourrait être l'étude des nouveaux mots clés et thèmes. Concernant les thèmes, il faudrait alors un modèle capable de nommer lui même des thèmes. Mais pour les mots clés, il serait possible de surveiller l'apparition de nouveaux termes et mots clés, indiquant probablement un nouvel évènement ou une nouvelle problématique dans les tracts. Par exemple, début 2020, on aurait pu observer de nouveaux mots clés tels que "covid" et "pandémie". Des études plus en détail concernant la fréquence des mots clés pourraient déjà permettre de repérer les thèmes émergents sur une période donnée. Dans le cadre de la transmission de sens, dans la même idée que la précédente, il serait intéréssant de travailler sur un modèle de détection de nouveauté (tâche de *novelty detection*), qui pourrait répondre à cette problématique.

Pour finir, nous n'avons pu que donner des pistes concernant de nouvelles méthodes d'évaluation. Une évolution évidente de notre travail serait d'implémenter et tester ces nouvelles pistes, dans l'objectif de créer une méthode d'évaluation au mieux simple, efficace et reproductible. Une meilleure évolution nous guiderait plus facilement vers le choix d'un meilleur modèle.

Pour conclure Dans ce travail, nous avons pu explorer les différentes techniques d'extraction de sens, et exploiter celles que nous avons jugées être les trois principales, pour créer un outil, une application d'aide à la transmission de sens. Lors de l'élaboration de notre *pipeline* il a été particulièrement important de prendre en compte les spécificités d'un problème réel. En effet, l'efficacité du modèle que l'on va choisir pour une tâche est toujours à mettre en balance avec sa complexité d'utilisation et les ressources nécessaires. Cela nous a mené à construire des solutions qui pouvaient sembler plus simples mais pas forcément moins performantes par rapport aux exigences que nous avions. Par ailleurs il a été important de remettre en question ce qui est désigné comme l'état de l'art. Selon les exigences du problème - par exemple, dans notre cas, il n'etait pas envisageable de se risquer de donner de fausses informations - certaines méthodes peuvent finalement sembler bien moins performantes que dans la théorie. Le domaine de la synthèse automatique est, depuis longtemps, mais en particulier dernièrement, très actif et très prolifique. Cependant, les recherches se concentrent très majoritairement sur la création de nouveaux modèles, avec un meilleur score *ROUGE* que les précédents. Nous pensons que ce score *ROUGE* n'est plus un bon indicateur de la qualité des résumés et qu'il est possible de faire bien mieux. Rien ne garantit que si on utilisait une méthode d'évaluation qui prend vraiment en compte les qualités d'un résumé, telles que la pertinence, la consistance, la fluidité et la cohérence, les modèles "*état de l'art*" seraient les mêmes qu'actuellement. La recherche concernant les questions d'extraction d'informations a encore de beaux jours devant elle, notamment concernant la recherche d'une nouvelle méthode d'évaluation standard, mais également concernant les méthodes non supervisées, ou agnostiques concernant le domaine et la langue.

Références

[1] Sanjeev Arora, Yingyu Liang, and Tengyu Ma. A simple but tough-to-beat baseline for sentence embeddings. *ICLR*, 2017.

[2] Dzmitry Bahdanau, KyungHyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. *ICLR*, 2015.

[3] Satanjeev Banerjee and Alon Lavie. Meteor : An automatic metric for mt evaluation with improved correlation with human judgments. *ACL*, 2005.

[4] Marcello Barbella, Michele Risi, and Genoveffa Tortora. A comparison of methods for the evaluation of text summarization techniques. *Proceedings of the 10th International Conference on Data Science, Technology and Applications (DATA 2021)*, 2021.

[5] Marcello Barbella, Michele Risi, and Genoveffa Tortora. A comparison of methods for the evaluation of text summarization techniques. *Proceedings of the 10th International Conference on Data Science, Technology and Applications*, 2021.

[6] Taylor Berg-Kirkpatrick, Dan Gillick, and Dan Klein. Jointly learning to extract and compress. *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics : Human Language Technologies-Volume 1. Association for Computational Linguistics*, 2011.

[7] Vincent D. Blondel, Jean-Loup Guillaume, Renaud Lambiotte, and Etienne Lefebvre. Fast unfolding of communities in large networks. *Journal of Statistical Mechanics : Theory and Experiment*, 2008.

[8] Jaime Carbonell and Jade Goldstein. The use of mmr, diversity-based reranking for reordering documents and producing summaries. *International Joint Conference on Artificial Intelligence*, 1998.

[9] K. W. Church and P. Hanks. Word association norms, mutual information, and lexicography. *ACL*, 1990.

[10] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert : Pre-training of deep bidirectional transformers for language understanding. *Association for Computational Linguistics*, 2019.

[11] Ted Dunning. Accurate methods for the statistics of surprise and coincidence. *Computational Linguistics*, 1993.

[12] Moussa Kamal Eddine, Antoine J-P Tixier, and Michalis Vazirgiannis. Barthez : a skilled pretrained french sequence-to-sequence model. *arXiv preprint arXiv :2010.12321*, 2021.

[13] Harold P Edmundson. New methods in automatic extracting. *Journal of the ACM (JACM), Austin, Texas*, 1969.

[14] effrey Pennington and Richard Socherand Christopher D. Manning. Glove : Global vectors for word representation. *EMNLP*, 2014.

[15] Alex Wang et al. Asking and answering questions to evaluate the factual consistency of summaries. *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, 2020.

[16] Cho et al. Learning phrase representations using rnn encoder-decoder for statistical machine translation. *EMNLP*, 2014.

[17] Jingqing Zhang et al. Pegasus : Pre-training with extracted gap sentences for abstractive summarization. *ICML*, 2020.

[18] Mehdi Allahyari et al. Text summarization techniques : A brief survey. *International Journal of Advanced Computer Science and Applications*, pages 397 – 405, 2017.

[19] Yue Dong et al. Multi-fact correction in abstractive text summarization. *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing*, 2020.

[20] Alexander R. Fabbri, Wojciech Kryscinski, Bryan McCann, Caiming Xiong, Richard Socher, and Dragomir Radev. Summeval : Re-evaluating summarization evaluation. *ACL*, 2021.

[21] Tobias Falke, Leonardo FR Ribeiro, Prasetya Ajie Utama, Ido Dagan, and Iryna Gurevych. Ranking generated summaries by correctness : An interesting but challenging application for natural language inference. *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, 2020.

[22] Sebastian Gehrmann, Yuntian Deng, and Alexander M. Rush. Bottom-up abstractive summarization. *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, 2018.

[23] Dan Gillick and Benoit Favre. A scalable global model for summarization. *ACL*, 2009.

[24] Yihong Gong and Xin Liu. Generic text summarization using relevance measure and latent semantic analysis. *Proceedings of the 24th annual international ACM SIGIR conference on Research and development in information retrieval. ACM*, 2001.

[25] Donna Harman and Paul Over. The effects of human variation in duc summarization evaluation. *ACL*, 2004.

[26] Dragomir R Radevand Hongyan Jing, Małgorzata Styś, and Daniel Tam. Centroid-based summarization of multiple documents. *Information processing management*, 2004.

[27] Kevin Gimpel Karen Livescu John Wieting, Mohit Bansal. From paraphrase database to compositional paraphrase model and back. *ACL*, 2015.

[28] Dan Jurafsky and James H. Martin. *Speech and Language Processing*. 2021.

[29] Thomas N. Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. *ICLR*, 2017.

[30] Oleksandra Klymenko, Daniel Braun, and Florian Matthes. Automatic text summarization : A state-of-the-art review. *22nd International Conference on Enterprise Information Systems*, pages 648 – 655, 2020.

[31] Kevin Knight and Daniel Marcu. Statistics-based summarization-step one : Sentence compression. *AAAI/IAAI.*, 2000.

[32] Chang-Shing Lee, Zhi-Wei Jian, and Lin-Kai Huang. A fuzzy ontology and its application to news summarization. *IEEE*, 2010.

[33] Chin-Yew Lin. Rouge : A package for automatic evaluation of summaries. *ACL*, 2004.

[34] Yang Liu and Mirella Lapata. Text summarization with pretrained encoders. *EMNLP*, 2019.

[35] Yinhan Liu, Myle Ott, Naman Goyal, and Jingfei Du. Roberta : A robustly optimized bert pretraining approach. 2020.

[36] H.P. Luhn. The automatic creation of literature abstracts. *IBM Journal of Research Development*, pages 159 – 165, 1958.

[37] E. Marsh, H. Hamburger, and Ralph Grishman. A production rule system for message summarization. *Proceedings of the 1984 National Conference on Artificial Intelligence, Austin, Texas*, 1984.

[38] Louis Martin, Benjamin Muller, Pedro Javier Ortiz Suárez, Yoann Dupont, Laurent Romary, Éric Villemonte de la Clergerie, Djamé Seddah, and Benoît Sagot. Camembert : a tasty french language model. 2020.

[39] Joshua Maynez, Shashi Narayan, Bernd Bohnet, and Ryan McDonald. On faithfulness and factuality in abstractive summarization. *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, 2020.

[40] Rada Mihalcea and Paul Tarau. Textrank : Bringing order into texts. *Association for Computational Linguistics*, 2004.

[41] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. 2013.

[42] Molière. *Dom Juan ou le Festin de Pierre*. 1682.

[43] Ramesh Nallapati, Feifei Zhai, and Bowen Zhou. Summarunner : A recurrent neural network based sequence model for extractive summarization of documents. *AAAI*, 2016.

[44] osef Steinberger, Massimo Poesio, Mijail A Kabadjov, and Karel Ježek. Two uses of anaphora resolution in summarization. *nformation Processing Management*, 2007.

[45] Makbule Gulcin Ozsoy, Ilyas Cicekli, and Ferda Nur Alpaslan. Text summarization of turkish texts using latent semantic analysis. *Proceedings of the 23rd international conference on computational linguistics. Association for Computational Linguistics*, 2010.

[46] Larry Page, Sergey Brin, R. Motwani, and T. Winograd. The pagerank citation ranking : Bringing order to the web. 1998.

[47] Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. Bleu : a method for automatic evaluation of machine translation. *ACL*, 2002.

[48] Armand Joulin Tomas Mikolov Piotr Bojanowski, Edouard Grave. Enriching word vectors with subword information. 2013.

[49] Campos R., V. Mangaravite, Pasquali A., Jorge A., Nunes C., and Jatowt A. Yake! collection-independent automatic keyword extractor. *Information Sciences Journal*, 2018.

[50] Campos R., V. Mangaravite, Pasquali A., Jorge A., Nunes C., and Jatowt A. Yake! keyword extraction from single documents using multiple local features. *Information Sciences Journal*, 2020.

[51] Fano R. Transmission of information : A statistical theory of communications. *MIT Press*, 1961.

[52] Horacio Saggion and Thierry Poibeau. Automatic text summarization : Past, present and future. *Theory and Applications of Natural Language Processing*, 2012.

[53] Gerard Salton and Christopher Buckley. Term-weighting approaches in automatic text retrieval. *Information processing management*, 1988.

[54] Thomas Scialom, Paul-Alexis Dray, Sylvain Lamprier, Benjamin Piwowarski, and Jacopo Staiano. Mlsum : The multilingual summarization corpus. *EMNLP*, 2010.

[55] Abigail See, Peter J. Liu, and Christopher D. Manning. . get to the point : Summarization with pointer-generator networks. *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics*, 2017.

[56] Wen tau Yih, Joshua Goodman, Lucy Vanderwende, , and Hisami Suzuki. 2. Multi-document summarization by maximizing informative content-words. *IJCAI*, 2007.

[57] Lucy Vanderwende, Hisami Suzuki, Chris Brockett, and Ani Nenkova. Beyond sumbasic : Task-focused summarization with sentence simplification and lexical expansion. *Information Processing Management*, 2005.

[58] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. 2017.

[59] John Wieting, Mohit Bansal, Kevin Gimpel, and Karen Livescu. Towards universal paraphrastic sentence embeddings. *ICLR*, 2016.

[60] Ming Zhou Xingxing Zhang, Furu Wei. Hibert : Document level pre-training of hierarchical bidirectional transformers for document summarization. *ACL*, 2019.

[61] Liang Yao, Chengsheng Mao, and Yuan Luo. Graph convolutional networks for text classification. *IAAA*, 2018.

[62] Ming Zhong, Pengfei Liu, Yiran Chen, Danqing Wang, Xipeng Qiu, and Xuanjing Huang. Extractive summarization as text matching. *ACL*, 2020.

Annexes

Table des matières

A Monologue *tokenizé* par phrases : La tirade de l'inconstance [42]

0 : Quoi ?

1 : tu veux qu'on se lie à demeurer au premier objet qui nous prend, qu'on renonce au monde pour lui, et qu'on n'ait plus d'yeux pour personne ?

2 : La belle chose de vouloir se piquer d'un faux honneur d'être fidèle, de s'ensevelir pour toujours dans une passion, et d'être mort dès sa jeunesse à toutes les autres beautés qui nous peuvent frapper les yeux !

3 : Non, non :

4 : la constance n'est bonne que pour des ridicules ; toutes les belles ont droit de nous charmer, et l'avantage d'être rencontrée la première ne doit point dérober aux autres les justes prétentions qu'elles ont toutes sur nos cœurs.

5 : Pour moi, la beauté me ravit partout où je la trouve, et je cède facilement à cette douce violence dont elle nous entraîne.

6 : J'ai beau être engagé, l'amour que j'ai pour une belle n'engage point mon âme à faire injustice aux autres ; je conserve des yeux pour voir le mérite de toutes, et rends à chacune les hommages et les tributs où la nature nous oblige.

7 : Quoi qu'il en soit, je ne puis refuser mon cœur à tout ce que je vois d'aimable ; et dès qu'un beau visage me le demande, si j'en avais dix mille, je les donnerais tous.

8 : Les inclinations naissantes, après tout, ont des charmes inexplicables, et tout le plaisir de l'amour est dans le changement.

9 : On goûte une douceur extrême à réduire, par cent hommages, le cœur d'une jeune beauté, à voir de jour en jour les petits progrès qu'on y fait, à combattre par des transports, par des larmes et des soupirs, l'innocente pudeur d'une âme qui a peine à rendre les armes, à forcer pied à pied toutes les petites résistances qu'elle nous oppose, à vaincre les scrupules dont elle se fait un honneur et la mener doucement où nous avons envie de la faire venir.

10 : Mais lorsqu'on en est maître une fois, il n'y a plus rien à dire ni rien à souhaiter ; tout le beau de la passion est fini, et nous nous endormons dans la tranquillité d'un tel amour, si quelque objet nouveau ne vient réveiller nos désirs, et présenter à notre cœur les charmes attrayants d'une conquête à faire.

11 : Enfin il n'est rien de si doux que de triompher de la résistance d'une belle personne, et j'ai sur ce sujet l'ambition des conquérants, qui volent perpétuellement de victoire en victoire, et ne peuvent se résoudre à borner leurs souhaits.
12 : Il n'est rien qui puisse arrêter l'impétuosité de mes désirs :
13 : je me sens un cœur à aimer toute la terre ; et comme Alexandre, je souhaiterais qu'il y eût d'autres mondes, pour y pouvoir étendre mes conquêtes amoureuses.

B Résultats de la synthèse automatique selon 4 méthodes et sur un corpus de 3 textes

Comparaison de 4 méthodes de vectorisation pour le texte monologue. On surligne les 3 phrases les plus importantes.

Méthode TFIDF, top 5 : [10, 6, 2, 13, 9]

Quoi ? tu veux qu'on se lie à demeurer au premier objet qui nous prend, qu'on renonce au monde pour lui, et qu'on n'ait plus d'yeux pour personne ? **La belle chose de vouloir se piquer d'un faux honneur d'être fidèle, de s'ensevelir pour toujours dans une passion, et d'être mort dès sa jeunesse à toutes les autres beautés qui nous peuvent frapper les yeux !** Non, non : la constance n'est bonne que pour des ridicules ; toutes les belles ont droit de nous charmer, et l'avantage d'être rencontrée la première ne doit point dérober aux autres les justes prétentions qu'elles ont toutes sur nos cœurs. Pour moi, la beauté me ravit partout où je la trouve, et je cède facilement à cette douce violence dont elle nous entraîne. **J'ai beau être engagé, l'amour que j'ai pour une belle n'engage point mon âme à faire injustice aux autres ; je conserve des yeux pour voir le mérite de toutes, et rends à chacune les hommages et les tributs où la nature nous oblige.** Quoi qu'il en soit, je ne puis refuser mon cœur à tout ce que je vois d'aimable ; et dès qu'un beau visage me le demande, si j'en avais dix mille, je les donnerais tous. Les inclinations naissantes, après tout, ont des charmes inexplicables, et tout le plaisir de l'amour est dans le changement. On goûte une douceur extrême à réduire, par cent hommages, le cœur d'une jeune beauté, à voir de jour en jour les petits progrès qu'on y fait, à combattre par des transports, par des larmes et des soupirs, l'innocente pudeur d'une âme qui a peine à rendre les armes, à forcer pied à pied toutes les petites résistances qu'elle nous oppose, à vaincre les scrupules dont elle se fait un honneur et la mener doucement où nous avons envie de la faire venir. **Mais lorsqu'on en est maître une fois, il n'y a plus rien à dire ni rien à souhaiter ; tout le beau de la passion est fini, et nous nous endormons dans la tranquillité d'un tel amour, si quelque objet nouveau ne vient réveiller nos désirs, et présenter à notre cœur les charmes attrayants d'une conquête à faire.** Enfin il n'est rien de si doux que de triompher de la résistance d'une belle personne, et j'ai sur ce sujet l'ambition des conquérants, qui volent perpétuellement de victoire en victoire, et ne peuvent se résoudre à borner leurs souhaits. Il n'est rien qui puisse arrêter l'impétuosité de mes désirs : je me sens un cœur à aimer toute la terre ; et comme Alexandre, je souhaiterais qu'il y eût d'autres mondes, pour y pouvoir étendre mes conquêtes amoureuses.

Méthode Baseline, top 5 : [10, 11, 6, 13, 9]

Quoi ? tu veux qu'on se lie à demeurer au premier objet qui nous prend, qu'on renonce au monde pour lui, et qu'on n'ait plus d'yeux pour personne ? La belle chose de vouloir se piquer d'un faux honneur d'être fidèle, de s'ensevelir pour toujours dans une passion, et d'être mort dès sa jeunesse à toutes les autres beautés qui nous peuvent frapper les yeux ! Non, non : la constance n'est bonne que pour des ridicules ; toutes les belles ont droit de nous charmer, et l'avantage d'être rencontrée la première ne doit point dérober aux autres les justes prétentions qu'elles ont toutes sur nos cœurs. Pour moi, la beauté me ravit partout où je la trouve, et je cède facilement à cette douce violence dont elle nous entraîne. **J'ai beau être engagé, l'amour que j'ai pour une belle n'engage point mon âme à faire injustice aux autres ; je conserve des yeux pour voir le mérite de toutes, et rends à chacune les hommages et les tributs où la nature nous oblige.** Quoi qu'il en soit, je ne puis refuser mon cœur à tout ce que je vois d'aimable ; et dès qu'un beau visage me le demande, si j'en avais dix mille, je les donnerais tous. Les inclinations naissantes, après tout, ont des charmes inexplicables, et tout le plaisir de l'amour est dans le changement. On goûte une douceur extrême à réduire, par cent hommages, le cœur d'une jeune beauté, à voir de jour en jour les petits progrès qu'on y fait, à combattre par des transports, par des larmes et des soupirs, l'innocente pudeur d'une âme qui a peine à rendre les armes, à forcer pied à pied toutes les petites résistances qu'elle nous oppose, à vaincre les scrupules dont elle se fait un honneur et la mener doucement où nous avons envie de la faire venir. **Mais lorsqu'on en est maître une fois, il n'y a plus rien à dire ni rien à souhaiter ; tout le beau de la passion est fini, et nous nous endormons dans la tranquillité d'un tel amour, si quelque objet nouveau ne vient réveiller nos désirs, et présenter à notre cœur les charmes attrayants d'une conquête à faire. Enfin il n'est rien de si doux que de triompher de la résistance d'une belle personne, et j'ai sur ce sujet l'ambition des conquérants, qui volent perpétuellement de victoire en victoire, et ne peuvent se résoudre à borner leurs souhaits.** Il n'est rien qui puisse arrêter l'impétuosité de mes désirs : je me sens un cœur à aimer toute la terre ; et comme Alexandre, je souhaiterais qu'il y eût d'autres mondes, pour y pouvoir étendre mes conquêtes amoureuses.

Méthode Tough-To-Beat freq constante, top 5 : [10, 11, 6, 13, 9]

Quoi ? tu veux qu'on se lie à demeurer au premier objet qui nous prend, qu'on renonce au monde pour lui, et qu'on n'ait plus d'yeux pour personne ? La belle chose de vouloir se piquer d'un faux honneur d'être fidèle, de s'ensevelir pour toujours dans une passion, et d'être mort dès sa jeunesse à toutes les autres beautés qui nous peuvent frapper les yeux ! Non, non : la constance n'est bonne que pour des ridicules ; toutes les belles ont droit de nous charmer, et l'avantage d'être rencontrée la première ne doit point dérober aux autres les justes prétentions qu'elles ont toutes sur nos cœurs. Pour moi, la beauté me ravit partout où je la trouve, et je cède facilement à cette douce violence dont elle nous entraîne. **J'ai beau être engagé, l'amour que j'ai pour une belle n'engage point mon âme à faire injustice aux autres ; je conserve des yeux pour voir le mérite de toutes, et rends à chacune les hommages et les tributs où la nature nous oblige.** Quoi qu'il en soit, je ne puis refuser mon cœur à tout ce que je vois d'aimable ; et dès qu'un beau visage

me le demande, si j'en avais dix mille, je les donnerais tous. Les inclinations naissantes, après tout, ont des charmes inexplicables, et tout le plaisir de l'amour est dans le changement. On goûte une douceur extrême à réduire, par cent hommages, le cœur d'une jeune beauté, à voir de jour en jour les petits progrès qu'on y fait, à combattre par des transports, par des larmes et des soupirs, l'innocente pudeur d'une âme qui a peine à rendre les armes, à forcer pied à pied toutes les petites résistances qu'elle nous oppose, à vaincre les scrupules dont elle se fait un honneur et la mener doucement où nous avons envie de la faire venir. **Mais lorsqu'on en est maître une fois, il n'y a plus rien à dire ni rien à souhaiter ; tout le beau de la passion est fini, et nous nous endormons dans la tranquillité d'un tel amour, si quelque objet nouveau ne vient réveiller nos désirs, et présenter à notre cœur les charmes attrayants d'une conquête à faire. Enfin il n'est rien de si doux que de triompher de la résistance d'une belle personne, et j'ai sur ce sujet l'ambition des conquérants, qui volent perpétuellement de victoire en victoire, et ne peuvent se résoudre à borner leurs souhaits.** Il n'est rien qui puisse arrêter l'impétuosité de mes désirs : je me sens un cœur à aimer toute la terre ; et comme Alexandre, je souhaiterais qu'il y eût d'autres mondes, pour y pouvoir étendre mes conquêtes amoureuses.

Méthode Tough-To-Beat, top 5 : [10, 11, 6, 13, 9]

Quoi ? tu veux qu'on se lie à demeurer au premier objet qui nous prend, qu'on renonce au monde pour lui, et qu'on n'ait plus d'yeux pour personne ? La belle chose de vouloir se piquer d'un faux honneur d'être fidèle, de s'ensevelir pour toujours dans une passion, et d'être mort dès sa jeunesse à toutes les autres beautés qui nous peuvent frapper les yeux ! Non, non : la constance n'est bonne que pour des ridicules ; toutes les belles ont droit de nous charmer, et l'avantage d'être rencontrée la première ne doit point dérober aux autres les justes prétentions qu'elles ont toutes sur nos cœurs. Pour moi, la beauté me ravit partout où je la trouve, et je cède facilement à cette douce violence dont elle nous entraîne. **J'ai beau être engagé, l'amour que j'ai pour une belle n'engage point mon âme à faire injustice aux autres ; je conserve des yeux pour voir le mérite de toutes, et rends à chacune les hommages et les tributs où la nature nous oblige.** Quoi qu'il en soit, je ne puis refuser mon cœur à tout ce que je vois d'aimable ; et dès qu'un beau visage me le demande, si j'en avais dix mille, je les donnerais tous. Les inclinations naissantes, après tout, ont des charmes inexplicables, et tout le plaisir de l'amour est dans le changement. On goûte une douceur extrême à réduire, par cent hommages, le cœur d'une jeune beauté, à voir de jour en jour les petits progrès qu'on y fait, à combattre par des transports, par des larmes et des soupirs, l'innocente pudeur d'une âme qui a peine à rendre les armes, à forcer pied à pied toutes les petites résistances qu'elle nous oppose, à vaincre les scrupules dont elle se fait un honneur et la mener doucement où nous avons envie de la faire venir. **Mais lorsqu'on en est maître une fois, il n'y a plus rien à dire ni rien à souhaiter ; tout le beau de la passion est fini, et nous nous endormons dans la tranquillité d'un tel amour, si quelque objet nouveau ne vient réveiller nos désirs, et présenter à notre cœur les charmes attrayants d'une conquête à faire. Enfin il n'est rien de si doux que de triompher de la résistance d'une belle personne, et j'ai sur ce sujet l'ambition des conquérants, qui volent perpétuellement de victoire en victoire, et ne peuvent se résoudre à borner leurs souhaits.** Il n'est rien qui puisse arrêter l'impétuosité de mes désirs : je me sens un cœur à aimer toute la terre ; et comme Alexandre, je souhaiterais qu'il y eût d'autres mondes, pour y pouvoir étendre mes conquêtes amoureuses.

Comparaison de 4 méthodes de vectorisation pour le texte tract. On surligne les 3 phrases les plus importantes.

Méthode TFIDF, top 5 : [6, 4, 1, 0, 2]

Plus de 160 000 salariés, actifs comme retraités, ont exprimé, avec près de 200 rassemblements, partout en France, leur mécontentement face aux choix économiques et sociaux gouvernementaux dictés par le patronat. **Dans chaque territoire et dans de très nombreuses entreprises de tous les secteurs d'activité, comme au sein des différentes administrations, ils ont porté leurs propositions et revendications pour le monde du travail.** Tout augmente, sauf les salaires et les pensions ! L'inflation repart à la hausse, les prix de l'énergie flambent, le pouvoir d'achat des ménages se réduit comme « peau de chagrin », dans le même temps, les bénéfices des grandes entreprises battent des records, avec plus de 57 milliards d'euros versés aux actionnaires ! **Les inégalités sociales n'ont jamais été aussi grandes, les choix politiques rarement aussi violents à l'encontre des services publics, de la protection sociale et des dispositifs de solidarité intergénérationnels.** Des politiques qui précarisent particulièrement les plus fragiles et la jeunesse. **En se mobilisant de manière unitaire dans de très nombreux secteurs, les salariés, les agents, les privés d'emplois et les retraités ont exprimé leurs revendications en matière de salaires, de pensions et de conditions de travail.** La CGT revendique l'augmentation automatique de tous les minima de branche et des pensions dès que le Smic augmente pour qu'aucun minima ne soit inférieur au Smic ! Elle agit aussi par la contestation de la réforme de l'assurance-chômage et, après avoir organisé de nombreuses initiatives de mobilisations contre ce projet funeste pour celles et ceux qui sont privés d'emploi, elle intente, comme l'ensemble des organisations syndicales de salariés, une action en justice devant le tribunal judiciaire. Le débat national doit se porter sur ce qui préoccupe prioritairement le monde du travail : les questions sociales ! Il faut en finir avec les thématiques nauséabondes qui irriguent les plateaux TV et nombre de médias qui ne visent qu'à détourner les débats des véritables enjeux de la période. Les mécontentements sont réels, la capacité d'y répondre tient à des choix politiques qui ne s'imposeront qu'à la force des combats qui seront menés !

Méthode Baseline, top 5 : [1, 6, 4, 3, 11]

Plus de 160 000 salariés, actifs comme retraités, ont exprimé, avec près de 200 rassemblements, partout en France, leur mécontentement face aux choix économiques et sociaux gouvernementaux dictés par le patronat. **Dans chaque territoire et dans de très nombreuses entreprises de tous les secteurs d'activité, comme au sein des différentes administrations, ils ont porté leurs propositions et revendications pour le monde du travail.** Tout augmente, sauf les salaires et les pensions ! L'inflation repart à la hausse, les prix de l'énergie flambent, le pouvoir d'achat des ménages se réduit comme « peau de chagrin », dans le même temps, les bénéfices des grandes entreprises battent des records, avec plus de 57 milliards d'euros versés aux actionnaires ! **Les inégalités sociales n'ont jamais été aussi grandes, les choix politiques rarement aussi violents à l'encontre des services publics, de la protection sociale et des dispositifs de solidarité intergénérationnels.** Des politiques qui précarisent particulièrement les plus fragiles et la jeunesse. **En se mobilisant de manière unitaire dans de très nombreux secteurs, les salariés, les agents, les privés d'emplois et les retraités ont exprimé leurs revendications en matière de salaires, de pensions et de conditions de travail.** La CGT revendique l'augmentation automatique de tous les minima de branche et des pensions dès que le Smic augmente pour qu'aucun minima ne soit inférieur au Smic ! Elle agit aussi par la contestation de la réforme de l'assurance-chômage et, après avoir organisé de nombreuses initiatives de mobilisations contre ce projet funeste pour celles et ceux qui sont privés d'emploi, elle intente, comme l'ensemble des organisations syndicales de salariés, une action en justice devant le tribunal judiciaire. Le débat national doit se porter sur ce qui préoccupe prioritairement le monde du travail : les questions sociales ! Il faut en finir avec les thématiques nauséabondes qui irriguent les plateaux TV et nombre de médias qui ne visent qu'à détourner les débats des véritables enjeux de la période. Les mécontentements sont réels, la capacité d'y répondre tient à des choix politiques qui ne s'imposeront qu'à la force des combats qui seront menés !

Méthode Tough-To-Beat freq constante, top 5 : [6, 1, 4, 3, 11]

Plus de 160 000 salariés, actifs comme retraités, ont exprimé, avec près de 200 rassemblements, partout en France, leur mécontentement face aux choix économiques et sociaux gouvernementaux dictés par le patronat. **Dans chaque territoire et dans de très nombreuses entreprises de tous les secteurs d'activité, comme au sein des différentes administrations, ils ont porté leurs propositions et revendications pour le monde du travail.** Tout augmente, sauf les salaires et les pensions ! L'inflation repart à la hausse, les prix de l'énergie flambent, le pouvoir d'achat des ménages se réduit comme « peau de chagrin », dans le même temps, les bénéfices des grandes entreprises battent des records, avec plus de 57 milliards d'euros versés aux actionnaires ! **Les inégalités sociales n'ont jamais été aussi grandes, les choix politiques rarement aussi violents à l'encontre des services publics, de la protection sociale et des dispositifs de solidarité intergénérationnels.** Des politiques qui précarisent particulièrement les plus fragiles et la jeunesse. **En se mobilisant de manière unitaire dans de très nombreux secteurs, les salariés, les agents, les privés d'emplois et les retraités ont exprimé leurs revendications en matière de salaires, de pensions et de conditions de travail.** La CGT revendique l'augmentation automatique de tous les minima de branche et des pensions dès que le Smic augmente pour qu'aucun minima ne soit inférieur au Smic !

Elle agit aussi par la contestation de la réforme de l’assurance-chômage et, après avoir organisé de nombreuses initiatives de mobilisations contre ce projet funeste pour celles et ceux qui sont privés d’emploi, elle intente, comme l’ensemble des organisations syndicales de salariés, une action en justice devant le tribunal judiciaire. Le débat national doit se porter sur ce qui préoccupe prioritairement le monde du travail : les questions sociales ! Il faut en finir avec les thématiques nauséabondes qui irriguent les plateaux TV et nombre de médias qui ne visent qu’à détourner les débats des véritables enjeux de la période. Les mécontentements sont réels, la capacité d’y répondre tient à des choix politiques qui ne s’imposeront qu’à la force des combats qui seront menés !

Méthode Tough-To-Beat, top 5 : [6, 4, 1, 8, 5]

Plus de 160 000 salariés, actifs comme retraités, ont exprimé, avec près de 200 rassemblements, partout en France, leur mécontentement face aux choix économiques et sociaux gouvernementaux dictés par le patronat. **Dans chaque territoire et dans de très nombreuses entreprises de tous les secteurs d’activité, comme au sein des différentes administrations, ils ont porté leurs propositions et revendications pour le monde du travail.** Tout augmente, sauf les salaires et les pensions ! L’inflation repart à la hausse, les prix de l’énergie flambent, le pouvoir d’achat des ménages se réduit comme « peau de chagrin », dans le même temps, les bénéfices des grandes entreprises battent des records, avec plus de 57 milliards d’euros versés aux actionnaires ! **Les inégalités sociales n’ont jamais été aussi grandes, les choix politiques rarement aussi violents à l’encontre des services publics, de la protection sociale et des dispositifs de solidarité intergénérationnels.** Des politiques qui précarisent particulièrement les plus fragiles et la jeunesse. **En se mobilisant de manière unitaire dans de très nombreux secteurs, les salariés, les agents, les privés d’emplois et les retraités ont exprimé leurs revendications en matière de salaires, de pensions et de conditions de travail.** La CGT revendique l’augmentation automatique de tous les minima de branche et des pensions dès que le Smic augmente pour qu’aucun minima ne soit inférieur au Smic ! Elle agit aussi par la contestation de la réforme de l’assurance-chômage et, après avoir organisé de nombreuses initiatives de mobilisations contre ce projet funeste pour celles et ceux qui sont privés d’emploi, elle intente, comme l’ensemble des organisations syndicales de salariés, une action en justice devant le tribunal judiciaire. Le débat national doit se porter sur ce qui préoccupe prioritairement le monde du travail : les questions sociales ! Il faut en finir avec les thématiques nauséabondes qui irriguent les plateaux TV et nombre de médias qui ne visent qu’à détourner les débats des véritables enjeux de la période. Les mécontentements sont réels, la capacité d’y répondre tient à des choix politiques qui ne s’imposeront qu’à la force des combats qui seront menés !

Comparaison de 4 méthodes de vectorisation pour le texte wiki. On surligne les 3 phrases les plus importantes.

Méthode TFIDF, top 5 : [1, 2, 0, 3, 10]

Un résumé est une forme de compression textuelle avec perte d'information. Un résumé automatique de texte est une version condensée d'un document textuel, obtenu au moyen de techniques informatiques. La forme la plus connue et la plus visible des condensés de textes est le résumé, représentation abrégée et exacte du contenu d'un document. Cependant, produire un résumé pertinent et de qualité demande au résumeur (un humain ou un système automatique) l'effort de sélectionner, d'évaluer, d'organiser et d'assembler des segments d'information selon leur pertinence. Bien comprendre et gérer les phénomènes de redondance, cohérence et cohésion est fondamental afin de produire des résumés automatiques humainement crédibles. Il y a plusieurs types de résumés selon leur but : mono-document, mi-document, guidé (personnalisé) ou non (générique) par une requête utilisateur, entre autres. Dernièrement des résumés autres que textuelles ont vu leur jour. Ainsi des résumés audio et vidéo font partie des recherches actuelles. Des résumés dans des domaines très spécialisés comme la médecine ou la chimie organique posent des vraies défis aux systèmes de traitement automatique de la langue naturelle. Un sujet connexe est l'extraction de sentiments à partir d'un texte. On part de l'hypothèse que pour un texte donné, il est non seulement possible de déterminer s'il contient une opinion (i.e. une vue subjective) mais également de déterminer si cette opinion est positive ou négative. Un exemple immédiat d'application est la recherche de critiques sur un film, où elles seraient organisées automatiquement en critiques positives et négatives. On peut également penser à un classement de produits du commerce en fonction des sentiments donnés en retour par les commentaires. Une première approche naïve fait appel aux mots clés du texte : en se basant sur un dictionnaire d'adjectifs, on atteindrait une précision de 62 % sur les sentiments exprimés dans un texte, pouvant aller jusqu'à 68 % si on prend en compte noms, verbes, etc. D'autres approches utilisent des arbres de décision pour classer le sujet (jusqu'à 73 % de précision) ou la rhétorique utilisée dans le texte.

Méthode Baseline, top 5 : [12, 3, 2, 4, 13]

Un résumé est une forme de compression textuelle avec perte d'information. Un résumé automatique de texte est une version condensée d'un document textuel, obtenu au moyen de techniques informatiques. **La forme la plus connue et la plus visible des condensés de textes est le résumé, représentation abrégée et exacte du contenu d'un document. Cependant, produire un résumé pertinent et de qualité demande au résumeur (un humain ou un système automatique) l'effort de sélectionner, d'évaluer, d'organiser et d'assembler des segments d'information selon leur pertinence.** Bien comprendre et gérer les phénomènes de redondance, cohérence et cohésion est fondamental afin de produire des résumés automatiques humainement crédibles. Il y a plusieurs types de résumés selon leur but : mono-document, mi-document, guidé (personnalisé) ou non (générique) par une requête utilisateur, entre autres. Dernièrement des résumés autres que textuelles ont vu leur jour. Ainsi des résumés audio et vidéo font partie des recherches actuelles. Des résumés dans des domaines très spécialisés comme la médecine ou la chimie organique posent des vraies défis aux systèmes de traitement automatique de la langue naturelle. Un sujet connexe est l'extraction de sentiments à partir d'un texte. On part de l'hypothèse que pour un texte donné, il est non seulement possible de déterminer s'il contient une opinion (i.e. une vue subjective) mais également de déterminer si cette opinion est positive ou négative. **Un exemple immédiat d'application est la recherche de critiques sur un film, où elles seraient organisées automatiquement en critiques positives et négatives.** On peut également penser à un classement de produits du commerce en fonction des sentiments donnés en retour par les commentaires. Une première approche naïve fait appel aux mots clés du texte : en se basant sur un dictionnaire d'adjectifs, on atteindrait une précision de 62 % sur les sentiments exprimés dans un texte, pouvant aller jusqu'à 68 % si on prend en compte noms, verbes, etc. D'autres approches utilisent des arbres de décision pour classer le sujet (jusqu'à 73 % de précision) ou la rhétorique utilisée dans le texte.

Méthode Tough-To-Beat freq constante, top 5 : [12, 16, 4, 13, 2]

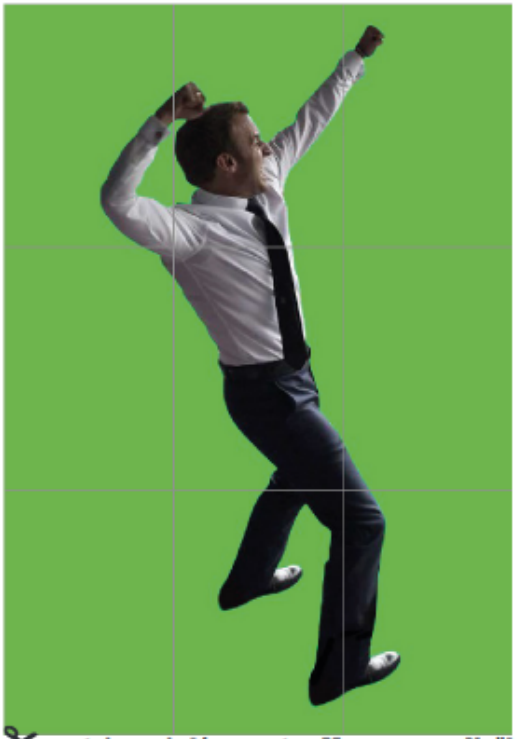
Un résumé est une forme de compression textuelle avec perte d'information. Un résumé automatique de texte est une version condensée d'un document textuel, obtenu au moyen de techniques informatiques. La forme la plus connue et la plus visible des condensés de textes est le résumé, représentation abrégée et exacte du contenu d'un document. Cependant, produire un résumé pertinent et de qualité demande au résumeur (un humain ou un système automatique) l'effort de sélectionner, d'évaluer, d'organiser et d'assembler des segments d'information selon leur pertinence. **Bien comprendre et gérer les phénomènes de redondance, cohérence et cohésion est fondamental afin de produire des résumés automatiques humainement crédibles.** Il y a plusieurs types de résumés selon leur but : mono-document, mi-document, guidé (personnalisé) ou non (générique) par une requête utilisateur, entre autres. Dernièrement des résumés autres que textuelles ont vu leur jour. Ainsi des résumés audio et vidéo font partie des recherches actuelles. Des résumés dans des domaines très spécialisés comme la médecine ou la chimie organique posent des vraies défis aux systèmes de traitement automatique de la langue naturelle. Un sujet connexe est l'extraction de sentiments à partir d'un texte. On part de l'hypothèse que pour un texte donné, il est non seulement possible de déterminer s'il contient une opinion (i.e. une vue subjective) mais également de déterminer si cette opinion est positive ou négative. **Un exemple immédiat d'application est la recherche de critiques sur un film, où elles seraient**

organisées automatiquement en critiques positives et négatives. On peut également penser à un classement de produits du commerce en fonction des sentiments donnés en retour par les commentaires. Une première approche naïve fait appel aux mots clés du texte : en se basant sur un dictionnaire d'adjectifs, on atteindrait une précision de 62 % sur les sentiments exprimés dans un texte, pouvant aller jusqu'à 68 % si on prend en compte noms, verbes, etc. **D'autres approches utilisent des arbres de décision pour classer le sujet (jusqu'à 73 % de précision) ou la rhétorique utilisée dans le texte.**

Méthode Tough-To-Beat, top 5 : [3, 13, 16, 4, 12]

Un résumé est une forme de compression textuelle avec perte d'information. Un résumé automatique de texte est une version condensée d'un document textuel, obtenu au moyen de techniques informatiques. La forme la plus connue et la plus visible des condensés de textes est le résumé, représentation abrégée et exacte du contenu d'un document. **Cependant, produire un résumé pertinent et de qualité demande au résumeur (un humain ou un système automatique) l'effort de sélectionner, d'évaluer, d'organiser et d'assembler des segments d'information selon leur pertinence.** Bien comprendre et gérer les phénomènes de redondance, cohérence et cohésion est fondamental afin de produire des résumés automatiques humainement crédibles. Il y a plusieurs types de résumés selon leur but : mono-document, mi-document, guidé (personnalisé) ou non (générique) par une requête utilisateur, entre autres. Dernièrement des résumés autres que textuelles ont vu leur jour. Ainsi des résumés audio et vidéo font partie des recherches actuelles. Des résumés dans des domaines très spécialisés comme la médecine ou la chimie organique posent des vraies défis aux systèmes de traitement automatique de la langue naturelle. Un sujet connexe est l'extraction de sentiments à partir d'un texte. On part de l'hypothèse que pour un texte donné, il est non seulement possible de déterminer s'il contient une opinion (i.e. une vue subjective) mais également de déterminer si cette opinion est positive ou négative. Un exemple immédiat d'application est la recherche de critiques sur un film, où elles seraient organisées automatiquement en critiques positives et négatives. **On peut également penser à un classement de produits du commerce en fonction des sentiments donnés en retour par les commentaires.** Une première approche naïve fait appel aux mots clés du texte : en se basant sur un dictionnaire d'adjectifs, on atteindrait une précision de 62 % sur les sentiments exprimés dans un texte, pouvant aller jusqu'à 68 % si on prend en compte noms, verbes, etc. **D'autres approches utilisent des arbres de décision pour classer le sujet (jusqu'à 73 % de précision) ou la rhétorique utilisée dans le texte.**

C Page d'un tract avec les phrases les plus importantes surlignées



... toi aussi, découpe ton Macron pour Noël

Et 0 pour les fonctionnaires ?

Totalement « oublié-es » du dispositif, ceux-ci, notamment celles et ceux dont les salaires sont égaux ou inférieurs au SMIC, n'auront droit à aucune revalorisation de leurs traitements, ni aucune prime de fin d'année (sauf pour les fonctionnaires de la police à ce jour). Ils et elles devront continuer à supporter le gel de leurs salaires et subir les conséquences des baisses de recettes publiques pour financer les annonces de Macron. A savoir de nouvelles suppressions d'effectifs, des coupes dans les moyens et des suppressions de services publics pourtant en grande partie à l'origine de la colère des gilets jaunes. C'est au patronat et aux actionnaires de payer les annonces de Macron ! **L'augmentation des salaires doit se faire pour toutes et tous !**

Les cotisations sociales ça sert à quoi ?

Les gouvernements successifs et le patronat n'ont eu de cesse de nous parler de coût du travail, de charges sociales... ils tentent de faire croire qu'on y gagnerait à ne plus payer de cotisations sociales. Pourtant à y regarder de plus près rien n'est plus faux. Les cotisations retraites, chômage, maladie... payées par l'employeur et/ou par les salarié-es (pour les cotisations retraite), c'est du salaire. Ce n'est pas le salaire qu'on touche à la fin du mois, mais c'est celui qu'on touche toute sa vie, quand on est malade, qu'on est au chômage, en retraite. **Toutes ces cotisations qu'ils suppriment sans les remplacer, c'est du fric en moins pour la sécurité sociale, c'est ce qui fait qu'on baisse les allocations chômage, le remboursement des soins et les retraites.** Mais ça n'est pas perdu pour tout le monde, le patronat en fait son affaire en réduisant considérablement la part qu'il doit verser pour chaque salarié-e.

Et les impôts ça sert à quoi ?

On nous fait un sacré tour de passe-passe, l'Etat va prendre en charge les augmentations de pouvoir d'achat que les patrons ne veulent pas payer (à peine le gouvernement a-t-il évoqué la possibilité de rogner 10 % du CICE que le Medef est immédiatement monté au créneau)... mais cette générosité a un coût parce que le gouvernement maintient sa volonté de réduire le déficit public. Alors comment dans un tel contexte développer la proximité des services publics réclamée dans cette mobilisation ? **Une autre répartition des richesses ça se construit d'abord dans le rapport entre les salarié-es et les patrons, par plus de salaire, moins de profits et de dividendes, et ça se construit ensuite au niveau de l'Etat par un système fiscal juste, par des politiques de redistributions au profit de ceux et celles qui en ont le plus besoin et un développement des services publics au bénéfice de tous et toutes.**

Et 0 pour les chômeur-euses et les minimas sociaux ?

Rien non plus, pourtant ils et elles sont touché-es aussi par l'augmentation du coût de la vie, les dépenses incompressibles, les difficultés de mobilité pour trouver du travail.
Pour Solidaires, ce sont les patrons qui licencient, ce ne sont pas les chômeur-euses qui sont responsables alors personne ne doit rester sur le carreau.
Aucune mesure de revalorisation des minima sociaux n'a été annoncée par le gouvernement, rien pour le RSA, rien pour l'allocation aux adultes handicapé-es, rien pour le minimum vieillesse, rien pour les APL...

D Résultats de quatre algorithmes pour l'extraction de mots clés sur trois textes.

	TFIDF	Comp. Vecteurs	TextRank PMI	TextRank Embed.
Monologue	beauté amour cœur honneur charme âme passion oeil victoire résistance	âme honneur souhait beauté mérite désir vouloir passion fidèle donner	prétention passion honneur charmer beauté voler constance charme ridicule dérober	passion renoncer doucement constance rendre pouvoir faire réveiller perpétuellement facilement
Tract	mécontentement pension augmente minimer choix smic intent intergénérationnel détourner nauséabonde	chômage pension mobilisation économique réforme salarié mobiliser mécontentement retraité travail	nauséabonde contestation tv intergénérationnel peau flamber détourner chagrin irriguer intent	prioritairement économique revendication contestation action organisation administration mobilisation intergénérationnel inflation
Wiki	résumé automatique texte condenser opinion textuel sentiment approche critique document	texte textuel document résumé textuelles contenu résumeur exemple informatique dictionnaire	résumeur opinion vidéo classer segment dictionnaire requête générique assembler adjectif	également compression comprendre application traitement pertinence représentation précision information contenu

Nuage de mots pour le texte "tract" :

Nuage de mots pour le texte "wiki" :

E Mots clés donnés par les évaluateurs et résultats

Pour le texte ”monologue” :

A = [”Passion”, ”fidélité”, ”ridicule”, ”cœurs”, ”changement”, ”combattre”, ”innocente”, ”pudeur”, ”désirs”, ”conquête”]

C = [”amour”, ”beauté”, ”conquêtes”, ”ridicule”, ”constance”]

E = [”passion”, ”beauté”, ”cœur”, ”amour”, ”nature”, ”charme”, ”changement”, ”désir”, ”conquête”, ”victoire”]

L = [”passion”, ”charme”, ”amour”, ”désirs”, ”conquêtes”, ”amoureuses”]

M1 = [”constance”, ”charme”, ”amour”, ”désirs”, ”conquête”, ”dérober”]

M2 = [”fidèle”, ”honneur”, ”passion”, ”beautés”, ”amour”, ”yeux”, ”coeur”, ”naissantes”, ”changement”, ”désirs”]

Z = [”amour”, ”changement”, ”conquête”, ”beauté”, ”triomphe”]

Scores F1 pour les quatres algorithmes selon les différents évaluateurs :

	eval_A	eval_C	eval_E	eval_L	eval_M1	eval_M2	eval_Z
TFIDF	0.2	0.266667	0.6	0.375	0.25	0.5	0.266667
Comp. Vecteurs	0.1	0.133333	0.3	0.125	0	0.3	0.133333
TextRank PMI	0.2	0.4	0.3	0.25	0.375	0.2	0.133333
TextRank Embed.	0.1	0.133333	0.1	0.125	0.125	0.1	0

Pour le texte "tract" :

A = ["Mécontentement", "choix", "politiques", "gouvernement", "tous", "secteurs", "revendications", "Inflation", "justice", "Questions", "Sociales"]

C = ["manifestation", "inflation", "mesures", "mécontentement", "revendiquer"]

E = ["salariés", "inflation", "énergie", "inégalités", "social", "politique", "salaire", "pension", "travail", "chômage"]

L = ["inégalités", "sociales", "inflation", "revendications", "politique"]

M1 = ["salariés", "mécontentement", "inflation", "politiques", "entreprises", "revendications", "augmentation", "conditions", "travail", "organisations", "sociales", "débat", "national", "choix"]

M2 = ["mécontentement", "choix", "sociaux", "patronat", "inégalités", "Smic", "augmentation", "action"]

Z = ["salariés", "patronat", "inégalités", "sociales", "CGT", "questions", "sociales", "chômage"]

Scores F1 pour les quatres algorithmes selon les différents évaluateurs :

	eval_A	eval_C	eval_E	eval_L	eval_M1	eval_M2	eval_Z
TFIDF	0.190476	0.133333	0.1	0	0.166667	0.333333	0
Comp. Vecteurs	0.095238	0.133333	0.4	0	0.25	0.111111	0.222222
TextRank PMI	0	0	0	0	0	0	0
TextRank Embed.	0.190476	0.133333	0.1	0.266667	0.25	0.111111	0

Pour le texte "wiki" :

A = ["Résumé", "compression", "textuel", "défi", "classement"]

C = ["résumé", "méthode", "complexité", "diversité"]

E = ["résumé", "automatique", "information", "opinion", "précision"]

L = ["résumé", "information", "automatique"]

M1 = ["résumé", "automatique", "segments","informations", "audio", "vidéo", "recherches", "mots", "clés", "texte", "arbre", "décision", "rhétorique"]

M2 = ["résumé", "automatique", "document", "langue", "texte", "comprendre", "opinion", "défi"]

Z = ["résumé", "automatique", "extraction", "sentiments", "opinion"]

Scores F1 pour les quatres algorithmes selon les différents évaluateurs :

	eval_A	eval_C	eval_E	eval_L	eval_M1	eval_M2	eval_Z
TFIDF	0.266667	0.142857	0.4	0.307692	0.26087	0.555556	0.533333
Comp. Vecteurs	0.266667	0.142857	0.133333	0.153846	0.173913	0.333333	0.133333
TextRank PMI	0	0	0.133333	0	0.173913	0.111111	0.133333
TextRank Embed.	0.133333	0	0.266667	0.153846	0.086957	0.111111	0

F Critères d'évaluation pour l'épreuve de synthèse pour les concours aux grandes écoles

Disponible sur https://lewebpedagogique.com/carolinecaffier/files/2013/07/CPGE-m%C3%A9thode-r%C3%A9sum%C3%A9-de-texte.pdf

II. CRITERES D'EVALUATION :

+ Critères positifs :

- La présence de toutes les **idées essentielles** du texte clairement **reformulées**
- **L'enchaînement ordonné et pertinent de ces idées** (conformité de la structure logique du résumé à la structure logique du texte)
- **la qualité et la finesse de l'expression** pour restituer fidèlement la pensée du texte.

+ Critères négatifs (pénalités) :

- **Le non respect du nombre de mots** :

« Il est conseillé aux candidats de ne pas croire qu'un enseignant de français est incapable de compter au moins jusqu'à cent cinquante et donc d'afficher un nombre de mots résolument fantaisiste. Les (lourdes) pénalités prévues sont appliquées en totalité lorsque la tricherie est manifeste. »

Concours commun polytechnique. Rapport 2002 série PSI

- **Fautes d'orthographe ou de syntaxe**. **Manque de lisibilité de l'écriture**.

- **Absence de reformulation du texte**

- **Juxtaposition pure de bribes de texte** sans travail sur la construction logique ou construction logique erronée ou artificielle.

G Interface de l'application pour l'aide à la transmission de sens

Résultats pour un filtre donné : Les tracts concernant les syndicats CFE-CGC sur le périmètre de la fonction publique, pour tous les thèmes, entre le 18 septembre 2017 et le 6 janvier 2021. L'algorithme surlignera dans les tracts 20% des phrases.