MINISTÈRE DE L'ENSEIGNEMENT SUPÉRIEUR ET DE LA RECHERCHE SCIENTIFIQUE

UNIVERSITÉ ABDELHAMID IBN BADIS - MOSTAGANEM

**Faculté des Sciences Exactes et d'Informatique**

**Département de Mathématiques et informatique**

**Filière : Informatique**

MÉMOIRE DE FIN D'ÉTUDES

Pour l'Obtention du Diplôme de Master en Informatique

Option : **Ingénierie des Systèmes d'Information**

Présenté par :

**« BENMAHAMMED Ouiame Sarah »**

**«BAROUDA Chemsseddine»**

THÈME :

# Un résumé automatique du texte basé Sur une méthode abstractive

Soutenu le :

Devant le jury composé de :

| | | |
|---|---|---|
| BENSALLOUA ABDALLAH C. | MCA Université de Mostaganem | Président |
| MAGHNI SANDID Z. | MAA Université de Mostaganem | Examinateur |
| MECHAOUI Moulay Driss | MCA Université de Mostaganem | Encadreur |

Année Universitaire 2021-2022

# Résumé

Comme les informations sont disponibles en abondance pour chaque sujet sur Internet, la condensation des informations importantes sous forme de résumé profiterait à un certain nombre d'utilisateurs. Par conséquent, la communauté des chercheurs s'intéresse de plus en plus au développement de nouvelles approches pour résumer automatiquement le texte. L'une des applications importantes du traitement du langage naturel est le résumé de texte, qui aide les utilisateurs à gérer la grande quantité d'informations disponibles, en condensant le contenu des documents et en extrayant les faits ou les sujets les plus pertinents. Le résumé de texte peut être classé selon deux types : extractif et abstractif. Dans ce mémoire, une étude exhaustive sur les méthodes abstraites de résumé de texte a été présentée. Les deux grandes méthodes de résumé abstractif sont l'approche basée sur la structure et l'approche basée sur la sémantique. Ces méthodes de résumé abstractif produisent un résumé très cohésif, cohérent, moins redondant et riche en informations, pour cela nous avons décidé de créer des outils de résumé abstractif du texte suivant un plan pour la réalisation d’un nouveau modèle NLP conçus par cette tâche et l’utilise dans une application de résumé automatique du texte.

**Mots-clés :**

Résumé automatique, Transformateur, Traitement du langage naturel, Hugging Face, Réglage fin.

# Abstract

As the information is abundantly available for each topic on the Internet, the condensation of important information in the form of a summary would benefit a number of users. Therefore, the community of researchers are increasingly interested in developing new approaches to automatically summarize the text. One of the important applications of natural language processing is the summarization of text, which helps users manage the vast amount of information available, by condensing the content of documents and extracting the most important facts or topics relevant. The text summary can be classified according two types: extractive and abstractive. In this project, an exhaustive study on abstract text summarization methods was presented. The two major methods of abstractive summarization are the approach based on structure and semantics-based approach. These abstractive summary methods produce a very cohesive, coherent summary that is less redundant and full of information, for this we decided to create abstractive text summarization tools following a plan for the realization of a new NLP model designed by this task and use it in an automatic text summarization application.



**Keywords:**

Text summarization, Transformers, NLP, Hugging Face, Fine-tuning.

# *Dédicaces*

*Ce projet fin d'étude est dédié à mes chers parents, qui m'ont toujours poussé et motivé dans mes études. Sans eux, je n'aurais certainement pas fait d'études longues. Ce projet fin d'étude représente donc l'aboutissement du soutien et des encouragements qu'ils m'ont prodigués tout au long de ma scolarité. Qu'ils en soient remerciés par cette trop modeste dédicace.*

*C'est un moment de plaisir de dédier cet œuvre à mes sœurs : Nour Elhouda, Maroua et ibtissem, en signe d'amour de reconnaissance et de gratitude pour dévouement et les sacrifices dont vous avez fait toujours à mon égard.*

*Et finalement à mes amies Sarah, Meriem, Maroua et mon binôme Chemesseddine qui n'ont jamais cessés de me soutenir.*

*Benmahammed Sarah*

*Je dédie cet ouvrage*

*A ma maman qui m'a soutenu et encouragé durant ces années d'etudes qu'elle trouve ici le témoignage de ma profonde reconnaissance. Ton affection me couvre ta bienveillance me guide et la présence à mes cotés a toujours été ma source de force pour affronter les différents obstacles.*

*A celui qui m'a fait de moi un homme, mon père.*

*A mes très chers frères Yacine, Othmen et Hichem et ma sœur Khadidja, pour leur appui et leur soutien moral.*

*A mon binôme Benmahammed Ouiame et tous mes amis qui m'ont toujours encouragé, et à qui je souhaite plus de succès.*

*A tous ceux qui j'aime.*

*Barouda Chemesseddine*

# *Remerciements*

*On remercie dieu le tout puissant de nous avoir donné la santé et la volonté d'entamer et de terminer ce mémoire.*

*Tout d'abord ce travail ne serait pas aussi riche et n'aurait pas pu avoir le jour sans l'aide et l'encadrement de "Mr. Mechaoui Moulay driss" on le remercie pour la qualité de son encadrement exceptionnel pour sa patience sa rigueur et sa disponibilité durant notre préparation de ce mémoire.*

*Notre remerciement s'adresse à nos amis pour leur aide pratique et soutien moral et leurs encouragements.*

*Notre remerciement s'adresse également à tous nos professeurs pour leurs générosités et la grande patience dont ils ont su faire preuve malgré leurs charges académiques et professionnelles.*

# Liste des figures

# *Liste des tableaux*

# *Liste des abréviations*

***AI :*** *L'Intelligence Artificielle (Artificial Intelligence).*

***BART*** *: Bidirectional and Auto Regressive Transformers.*

***BERT*** *: Bidirectional Encoder Representations from Transformers.*

***BRNN*** *: Réseaux de neurones récurrents bidirectionnels (Bidirectional Recurrent Neural Network).*

***GPT*** *: Generative Pre-training Transformer.*

***GRU*** *: Unités récurrentes fermées (Gated Recurrent Unit ).*

***GSG:*** *Génération de phrases vides (Gap Sentence Generation).*

***LSA :*** *Analyse sémantique latente (Latent Semantic Analysis).*

***LSTM*** *: Mémoire à long court terme (Lange Short Time Memory).*

***MLM:*** *Modélisation du langage masqué (Masked Language Modeling).*

***MLP:*** *Perceptron multicouche (Multilayer perceptron).*

***NLG*** *: génération de langage naturel (Natural Language Generation ).*

***NLP*** *: le traitement du langage naturel (Natural Language Processing).*

***PEGASUS*** *: Pre-training with Extracted Gap-sentencesfor Abstractive Summarization Sequence-to-sequence model.*

***RNN*** *: Un réseau neuronal récurrent (Recurrent Neural Network ).*

***ANN*** *: réseau de neurones artificiels (Artificial Neural Networks).*

***RST*** *: la théorie de la structure rhétorique (Rhetorical Structure Theory) .*

***RSG*** : graphe sémantique riche ( Rich Semantic Graph ).

***Seq2Seq*** *: séquence à séquence (Sequence-to-Sequence).*

***T5*** *: Text-To-Text Transfer Transformer.*

***ROUGE :*** *Recall-Oriented Understudy for Gisting Evaluation.*

# *Table des matières*

# Introduction général

## Contexte

Avec l’augmentation rapide de la quantité des informations que nous avons aujourd'hui, spécialement les documents médicaux, nous n’arrivons pas à filtrer toutes les informations importantes. Cela prouve la nécessité de gérer cette masse pour retrouver rapidement les informations qu’ils contiennent, et récupérer l’idée pertinente parmi toutes les informations contenues dans les documents stockés. Le résumé automatique du texte est l'une des applications qui récupèrent ces informations. Il s'agit d'une méthode de compression du texte saisi en une version plus courte, en préservant son contenu d'information et sa signification globale.

## Problématique

Les documents de taille longue sont l’un des vrais problèmes face aux lecteurs. Commencer à lire un si grand nombre de documents pour rechercher des informations pertinentes est une tâche très difficile, pour la facilité on doit faire une synthèse automatique.

Donc nous voulons aider le lecteur à lire beaucoup des documents dans un court temps par la synthèse de texte, c’est un domaine d'apprentissage machine intéressant qui gagne de plus en plus de terrain parce qu’il est la meilleur solution pour qu’on traite de très long document en moindre de temps et effort possible.

# Objectifs

L’urgence de développer des outils efficaces pour le texte est devenue une nécessité essentielle, les traitements manuels de ces données s’avèrent très coûteux en temps et en personnel, ils sont peu flexibles et le traitement d’une très grande quantité dans ce cas est presque impossible. C’est pour cela que l’on cherche au point de mettre des méthodes automatiques. Les outils de résumé automatique disponibles sur le web ne donnent pas des bons résultats.

Nous travaillerons pour atteint les objectifs suivant :

- Citer les différentes techniques de résumé automatique.
- Faire comparer les techniques disponibles.
- Trouver la bonne méthode pour avoir un texte lisible et facile à comprendre sans changer de contenu et garder les informations essentielles.
- Aider le lecteur à repérer les informations qui peuvent l’intéresser sans pour autant devoir lire le document en entier.

# Chapitre1

# Intelligence artificielle pour la modélisation de traitement du langage naturel

## 1.1. Introduction

Le but de l'intelligence artificielle est de permettre aux ordinateurs de penser et d'agir comme des humains, il nécessite beaucoup de données et une puissance de traitement élevée. La génération de connaissances est rendue possible par le « machine learning » ou l'apprentissage automatique, les connaissances sont modélisées pour aider à la prise de décision et à l'analyse de texte. La recherche sur l'IA a montré un grand intérêt pour la génération automatique de résumés succincts qui préservent les informations importantes dans le texte.

## 1.2. Apprentissage automatique

L'apprentissage automatique «machine learning» est une technique de programmation informatique qui utilise des probabilités statistiques pour permettre aux ordinateurs d'apprendre par eux-mêmes sans programmation explicite. En ce qui concerne son objectif fondamental, l'apprentissage automatique apprend aux machines à apprendre, pour agissant et réagissant comme des humains, et améliorant de manière autonome leur style d'apprentissage et leurs connaissances au fil du temps. [1]

L'apprentissage automatique est utilisé lorsque les données sont abondantes (relativement) mais que les connaissances ne sont pas facilement disponibles ou sous-développées. Ainsi, l'apprentissage automatique peut également aider les humains à apprendre : les algorithmes d'apprentissage créent des modèles qui révèlent l'importance relative de certaines informations ou la manière dont elles interagissent pour résoudre un problème particulier. [4]

## 1.3. Apprentissage profond

L’apprentissage profond (Ces techniques sont utilisées dans les domaines informatiques des NTIC (reconnaissance visuelle - comme les panneaux routiers pour robots

ou véhicules autonomes - notamment reconnaissance de la parole), robotique, bio-informatique, reconnaissance ou comparaison de formes, sécurité, santé, etc. pédagogie assistée par ordinateur, et plus encore. Le plus répandu est l'intelligence artificielle «l'apprentissage en profondeur». deep learning) est l'une des principales technologies du Machine learning. Avec le Deep Learning, nous parlons d'algorithmes capables de mimer les actions du cerveau humain grâce à des réseaux de neurones, Le deep learning, quant à lui, fonctionne à partir de données non structurées. En effet, ces algorithmes s'appuient sur un réseau neuronal de plusieurs couches, similaire au cerveau humain. Un système avec un apprentissage profond est capable d'effectuer des tâches complexes alors qu'avec le machine learning il est plutôt question d'actions de routine.

Un IA à apprentissage profond possède également la capacité à apprendre en toute autonomie, contrairement à un IA à apprentissage automatique, qui nécessite l'intervention humaine [24].

Par exemple, des ordinateurs peuvent être faits pour mieux identifier et analyser des objets hautement déformables, des émotions révélées par un visage photographié ou photographié, ou pour analyser le mouvement et la position des doigts d'une main, ce qui est utile pour traduire des symboles dans le langage, améliorer le positionnement automatique des caméras, etc. Ils sont utilisés dans certaines formes d'aides au diagnostic médical (ex : reconnaissance automatique d'un cancer en imagerie médicale), ou de prospective ou de prédiction (ex : Prédire les propriétés du sol capturées par un robot).

## 1.4. Les réseaux de neurones artificiels

ANN est inspiré des neurones biologiques ( *Figure 1)*. Dans le cerveau humain, les réseaux de neurones biologiques travaillent ensemble pour recevoir des signaux d'entrée, traiter des informations et déclencher des signaux de sortie [2],

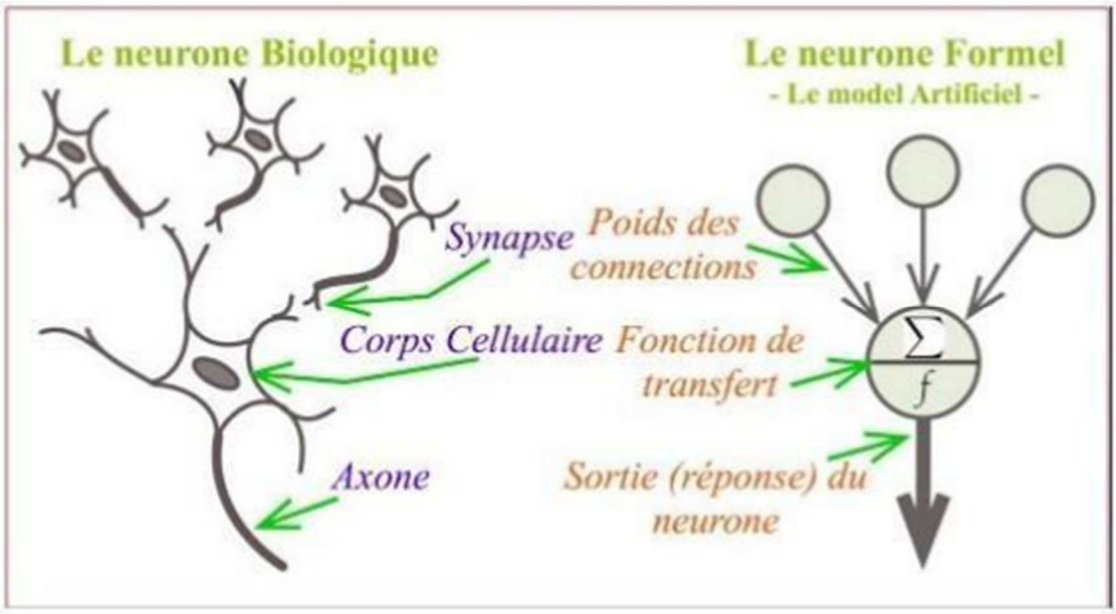



**Figure 1: Schéma des neurones biologiques et des neurones artificiel[5]**

Le réseau de neurones artificiels est un ensemble de neurones généralement virtuel disposé sur un réseau virtuel, chaque neurone est un point de réseau qui reçoit des informations entrantes et imite l'information sortante. Les signaux qui envoient les neurones sont des intensités de signaux. Un neurone peut rester silencieux tant qu'il reçoit le nombre 0, et s'activer tant qu'il reçoit un nombre supérieur à 0.

Un réseau de neurone est une fonction (machine à calculer) qui prend en entrée un vecteur d'une grande dimension et qui ressort un autre vecteur de grande dimension.

Un réseau est organisé en couches: il peut être constitué d'une unique couche, et est alors appelé perceptron. Chaque neurone de la couche reçoit toutes les informations entrées et renvoie son résultat vers un unique neurone de sortie. Ce qu'il renvoie est la sortie finale du réseau ( *Figure 2)*.

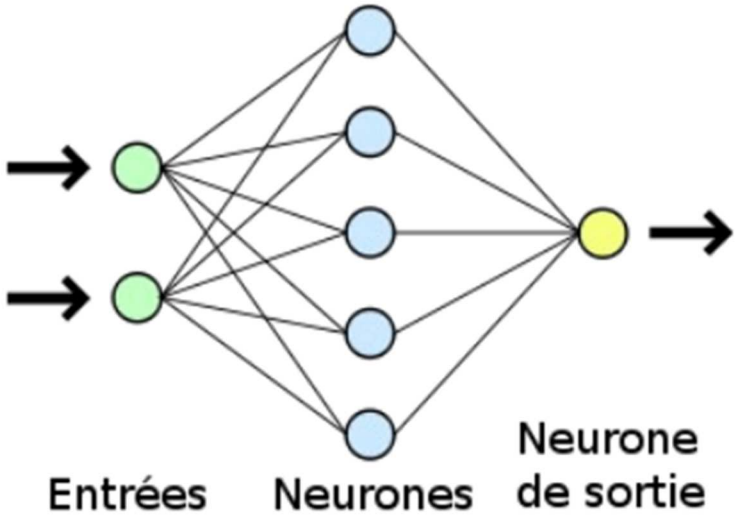



**Figure 2 : Réseau de neurones artificiels (unique couche)[6]**

Et le perceptron multi-couches (feed-forward) qui est un réseau neuronal unidirectionnel (*Figure 3)*, dans un réseau de neurones, il existe des neurones (première couche de neurones) qui sont chargés de capter des données extérieures (en machine learning c'est les bruit), comme : les paroles la musique etc.

Les neurones observables lit ces données bruit et s'active si ces données correspondent à son activation.

Les neurones observables sont actifs. Alors, envoyer des signaux selon des flèches de réseau (synapses), les neurones qui reçoivent les signaux observables sont les neurones de la 2eme couche, chaque neurones de cette couche divise les signaux en deux catégories : synapses activatrice et synapses inhibitrice.

Ce neurone collecte des contributions activatrices et inhibitrices et y a ajouté un biais, puis calcule un niveau d'activation en fonction de son résultat et d'une fonction d'activation.

La sortie s'écrira comme étant une succession de combinaisons linéaire de signaux de couche précédente auxquels sont appliquées des fonctions d'activation.

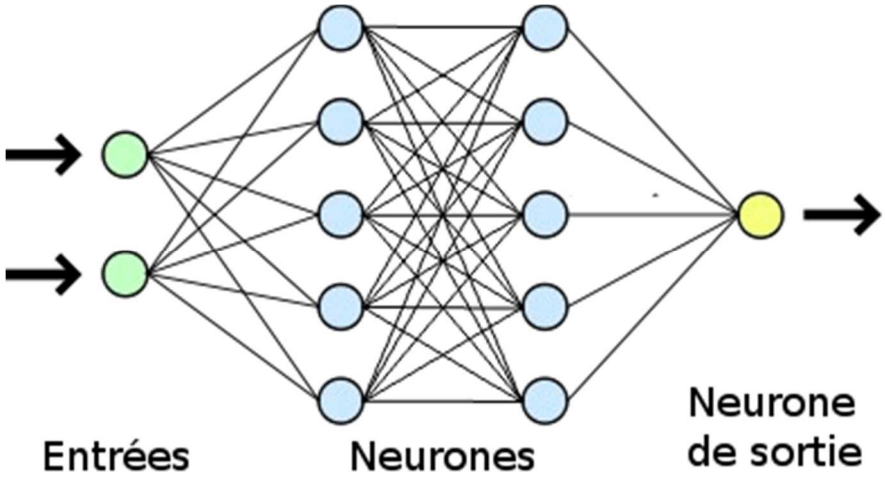



**Figure 3 : Approfondir les réseaux de neurones (multi-couches) [6]**

## 1.5. Réseau de neurones récurrent

### 1.5.1. Définition

RNN est un réseau neuronal artificiel qui utilise des données séquentielles ou chronologiques. Ces algorithmes d'apprentissage en profondeur sont couramment utilisés pour les problèmes ordinaux ou temporels, tels que la traduction de la langue, le traitement du langage naturel, la reconnaissance vocale et le sous-titrage d'images, les réseaux de neurones récurrents utilisent des données d'entraînement pour apprendre. Ils se distinguent par leur « mémoire » car ils prennent des informations d'entrées antérieures pour influencer l'entrée et la sortie actuelles (*Figure 4)*. Alors que les réseaux de neurones profonds traditionnels supposent que les entrées et les sorties sont indépendantes les unes des autres, la sortie des réseaux de neurones récurrents dépend des éléments antérieurs de la séquence (*Figure 5)*. Alors que les événements futurs seraient également utiles pour déterminer la sortie d'une séquence donnée, les réseaux de neurones récurrents unidirectionnels ne peuvent pas tenir compte de ces événements dans leurs prédictions [2].

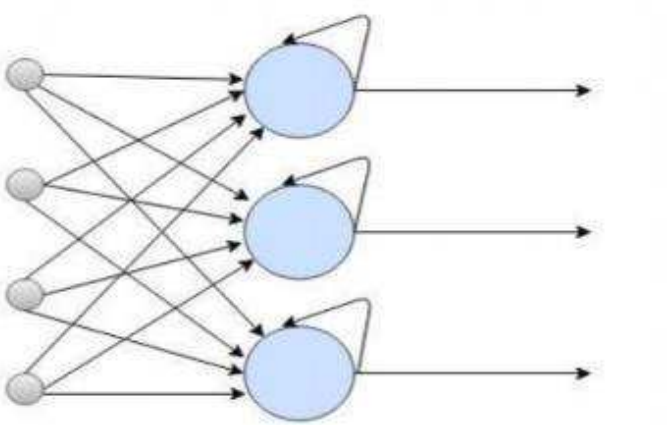


**Figure 4 : Réseau de neurones récurrent [14]**

Leurs entrées et sorties peuvent varier en longueur, et différents types de RNN sont utilisés pour différents cas d'utilisation, tels que la synthèse de texte.

Différents types de RNN sont : Un par un, Un-à-plusieurs, Plusieurs-à- un ,Plusieurs à plusieurs.

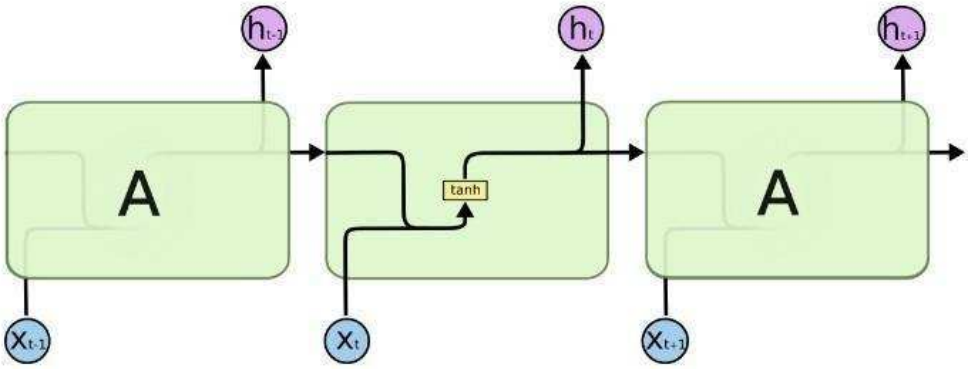



**Figure 5 : Unité de base RNN [14]**

- *$x_t$: Entrée à l'instant t*
- *$h_{t-1}$: État à l'instant t-1*
- *w : Une matrice soit séparée en deux parties:*

  *$w_{hh}$: correspondant aux connexions entre $h_{t-1}$ et $h_t$*

  *$w_{xt}$: correspondant aux connexions entre $x_t$ et $h_t$*

- *$h_t$est la sortie finale de la couche RNN, une couche RNN définit donc une relation de récurrence de la forme :*

$$h_t = f(x_t, h_{t-1}) \Rightarrow h_t = \tanh(w_{hh}\, h_{t-1} + w_{xh}\, x_t)$$

### 1.5.2. Le problème de gradient avancent (vanishing gradient)

À chaque fois que le signal passe par une boucle, il est multiplié par le poids des synapses qu'il traverse lors de l'apprentissage les gradients sont multipliés par le poids synaptique et varient de manière exponentielle. Ainsi ils ont tendance à disparaître ou d'exploser. Si le signal est multiplié par un nombre supérieur à 1 le signal explosera après quelque boucle. Si le signal est multiplié par un nombre inférieur à 1, le signal disparaît exponentiellement vite.

### 1.5.3. Architectures RNN variantes

#### 1.5.3.1. Réseaux de neurones récurrents bidirectionnels

Il s'agit d'une variante de l'architecture de réseau des RNN. Alors que les RNN unidirectionnels ne peuvent être extraits que des entrées précédentes pour faire des prédictions sur l'état actuel ( *Figure 6)* , les RNN bidirectionnels extraient les données futures pour en améliorer la précision. Si nous prenons l'exemple de "se sentir sous le temps", le modèle peut mieux prédire que le deuxième mot de cette phrase est "sous" s'il savait que le dernier mot de la séquence est "temps".[2]

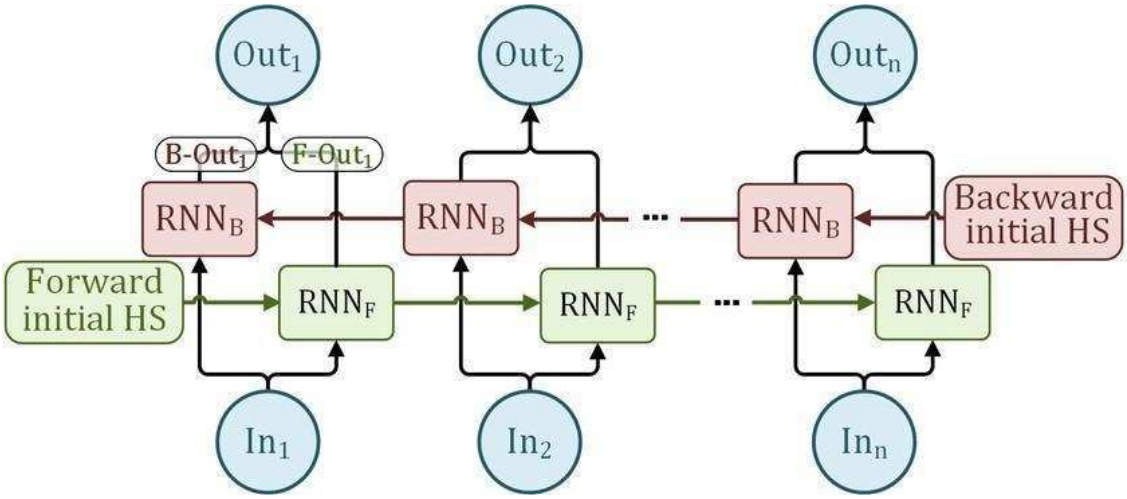



**Figure 6 : Architecture des réseaux de neurones récurrents bidirectionnels (BRNN) [14]**

### 1.5.3.2. Mémoire à long court terme

Il s'agit d'une architecture RNN populaire, qui a été introduite comme solution au problème de gradient de fuite. C'est-à-dire que si l'état précédent qui influence la prédiction actuelle n'est pas dans un passé récent (*Figure 7)*, le modèle RNN ne peut pas être en mesure de prédire avec précision l'état actuel, la figure 8 montre *le fonctionnement des LSTM.*[2]

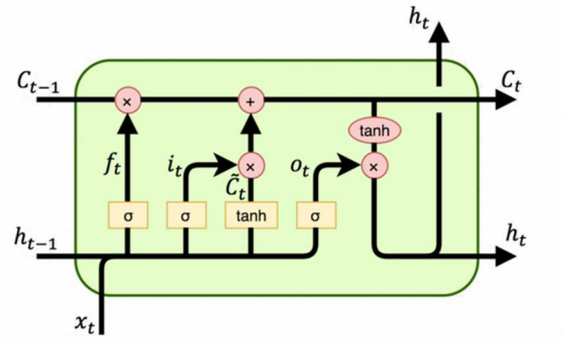



**Figure 7 : Unité de base LSTM [14]**

Exemple : nous voulions prédire les mots en italique suivants :

"Alice est allergique aux noix. Elle ne peut pas manger *de beurre de cacahuète*."

Le contexte d'une allergie aux noix peut nous aider à anticiper que les aliments qui ne peuvent pas être consommés contiennent des noix. Cependant, si ce contexte était précédé de quelques phrases, il serait alors difficile, pour le RNN de connecter les informations, pour y remédier.

Les LSTM ont des « cellules » dans les couches cachées du réseau de neurones, qui ont trois portes : une porte d'entrée, une porte de sortie et une porte d'oubli. Ces portes contrôlent le flux d'informations qui est nécessaire pour prédire la sortie dans le réseau.

Par exemple, si des pronoms de genre, tels que « elle », ont été répétés plusieurs fois dans des phrases précédentes, vous pouvez les exclure de l'état de la cellule.

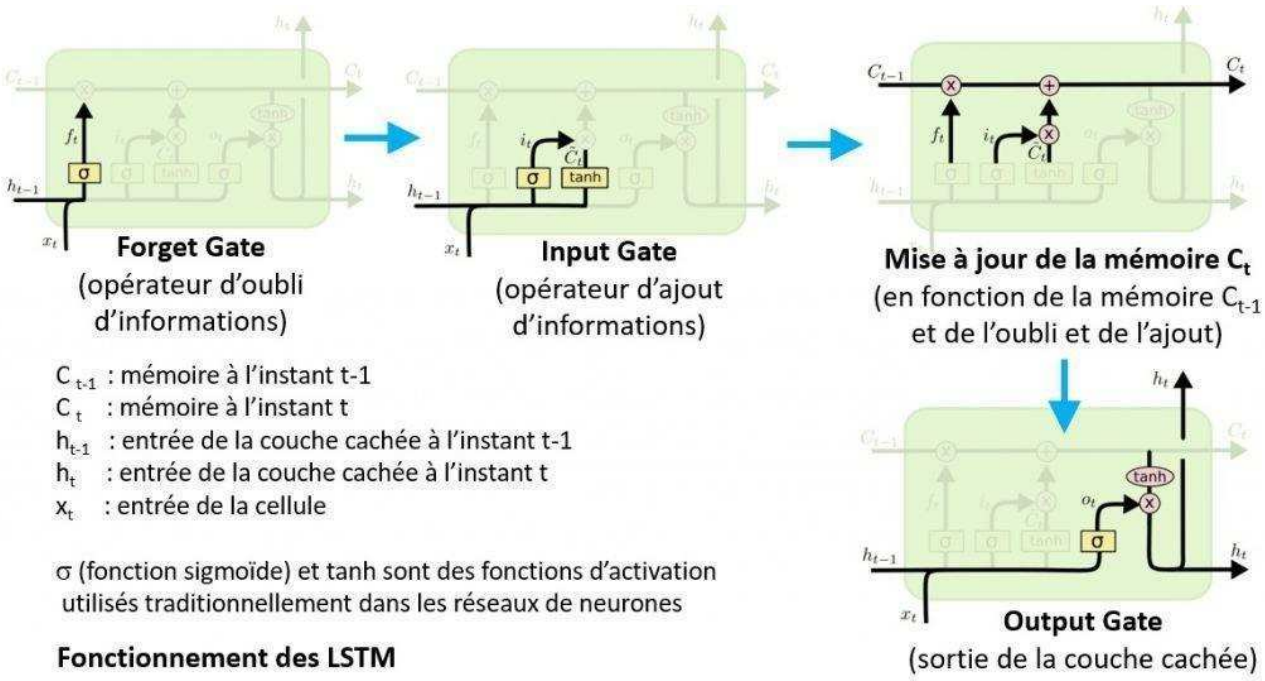
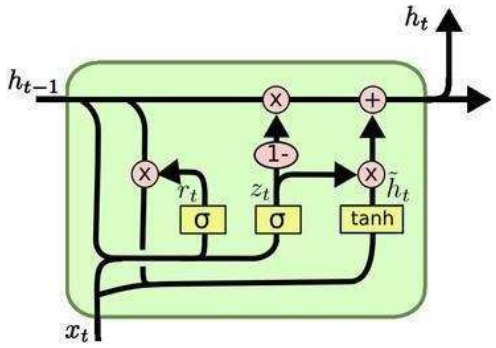



**Figure 8 : Fonctionnement des LSTM [14]**

### 1.5.3.3. Unités récurrentes fermées

Cette variante RNN est similaire aux LSTM car elle permet également de résoudre le problème de mémoire à court terme des modèles RNN. Au lieu d'utiliser un « état de cellule » pour réguler les informations, il utilise des états cachés, et au lieu de trois portes, il en a deux : une porte de réinitialisation et une porte de mise à jour ( *Figure 9)*. Semblables aux portes dans les LSTM, les portes de réinitialisation et de mise à jour contrôlent la quantité et les informations à conserver.[2]




$$z_t = \sigma\left(W_z \cdot [h_{t-1}, x_t]\right)$$
$$r_t = \sigma\left(W_r \cdot [h_{t-1}, x_t]\right)$$
$$\tilde{h}_t = \tanh\left(W \cdot [r_t * h_{t-1}, x_t]\right)$$
$$h_t = (1 - z_t) * h_{t-1} + z_t * \tilde{h}_t$$

**Figure 9 : Unité de base GRU [14]**

## 1.6. Conclusion

Dans ce chapitre, nous avons présenté les différentes domaine et les outils nécessaires utiliser pour le traitement du langage, nous avons vu que les RNNs sont les plus utilisés dans ce domaine, en déduisant que les anciens méthode de RNN ont beaucoup des inconvénients comme le LSTM qui a une mémoire courte qui n'aide pas à faire ça, et rendre le traitement de l'information plus difficile, alors que cette méthode ne donnera pas les bonnes résultat qu'on veut avoir.

# Chapitre2

# Techniques de résumé automatique

## 2.1. Introduction

Avec la pandémie de COVID-19, la communauté médicale a un besoin de plus en plus urgent de suivre la croissance accélérée de la nouvelle littérature liée au coronavirus. Cette pandémie cause une augmentation des documents, qui pose un vrai problème sur le domaine médical parce qu'il nécessite la rapidité de lire les rapports. Pour cela les chercheurs ont pris ce problème en sérieux, ils ont décidé d'évoluer des résumés pour ces documents pour gagner du temps et prendre des décisions. Le résumé de texte est une tâche essentielle dans le NLP qui vise à générer des résumés condensés retenant les informations importantes du document d'entrée.

## 2.2. Traitement du langage naturel

Le traitement du langage naturel est un domaine à la fois informatique et linguistique qui traite de la façon qu'un ordinateur traite les langues naturelles. L'objectif principal de la NLP est de permettre aux machines de comprendre le langage naturel et de le manipuler.

Quelques domaines de recherche majeurs en NLP: la traduction automatique, l'extraction d'informations, Récupération d'informations, Génération de langage naturel, Compréhension du langage naturel, Reconnaissance de la parole, Résumé, etc.[23]

## 2.3. Résumé automatique

Résumé automatique de texte est un domaine de recherche actif axé sur la condensation de gros morceaux de texte en texte plus petit en conservant les informations pertinentes. Le résumé peut être généré par des méthodes extractives ainsi que par des méthodes abstraites.

### 2.3.1. Méthode d'extraction

Le résumé extractif produit des résumés en concaténant plusieurs sens. Prises la phrase telle qu'elle est dans le document original en cours l'augmentation de résumé. Ce processus peut être divisé en deux étapes : le prétraitement et le traitement [21].

Le prétraitement convertit le texte non structuré en texte structuré du texte original de sorte que le principal avantage du prétraitement est la représentation structurée du texte original. Le prétraitement comprend généralement certains des sous-étapes suivantes :

1. Segmentation est le fait d'identifier les limites des phrases. La limite d'une phrase est identifiée par la présence d'un point à la fin de la phrase.
2. Élimination des mots vides.
3. Tokenisation identification des limites de mots). La limite du mot s'identifie à la présence d'un espace entre deux mots.
4. Racine, le but du radical est de ramener le mot à sa racine.

Traitement de la méthode d'extraction qui utilise des fonctionnalités, des règles ou une fonction heuristique pour créer un résumé à partir de texte structuré comprend les sous-étapes suivantes :

1. Les caractéristiques des phrases sont décidées et calculées à l'aide de méthodes statistiques ou linguistiques.
2. Des poids sont attribués à ces caractéristiques à l'aide d'une méthode d'apprentissage des poids.
3. Le score final de chaque phrase est déterminé à l'aide d'une équation de pondération des caractéristiques.
4. Les phrases les mieux classées sont sélectionnées pour le résumé final.

### 2.3.2. Méthode abstractive

Les méthodes de cette approche sont apparues vers la fin des années 1970. Elles s'inspirent principalement du domaine de l'IA et de la psychologie cognitive, et elles cherchent à produire des résumés avec une qualité linguistique comparable à celle des résumés produits par les êtres humains. Généralement, Les méthodes abstraites sont un processus de compréhension des principaux concepts dans un document et ensuite exprimer

ces concepts dans un langage naturel clair. Les techniques de résumé abstractive sont généralement classées en deux catégories (*Figure 10)*.[10] :

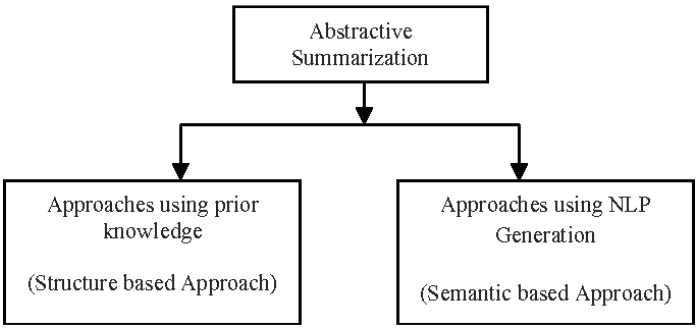



**Figure 10 : Présentation de la synthèse abstractive [13]**

1. Approche structurée

Différentes méthodes utilisant une approche structurée sont les suivantes : méthode basée sur l'arbre, méthode basée sur les modèles et méthode basée sur l'ontologie. [13]

2. Approche basée sur la sémantique

Des méthodes qui utilisent une approche basée sur la sémantique sont les suivantes : Modèle sémantique multimodal, Méthode basée sur les éléments d'information et méthode basée sur les graphes sémantiques. [13]

La représentation sémantique des documents est utilisée pour alimenter un système de NLG. Ce type dépend d'identification de syntagmes nominaux et de syntagmes verbaux en traitant des données linguistiques : chaîne lexicale, LSA, Informations de Coréférence, et RST. [11]

On distingue trois familles de méthodes pour l'approche abstractive [12]:

- des méthodes qui traduisent les informations importantes contenues dans les documents sources en des schémas cognitifs tels que les ontologies, les patrons, les graphes,
- les méthodes se basant sur la représentation sémantique des documents.
- les méthodes utilisant les réseaux de neurones dans le cadre de l'apprentissage profond (Deep Learning) à l'aide des transformers.

## 2.4. Les transformateurs «Transformers»

### 2.4.1. Définition

Transformers sont des types d'architecture de réseau neuronal (Deep Learning), développés en 2017. Les transformateurs et ce que nous appelons une architecture séquence à séquence. Seq2Seq est un réseau de neurones qui transforme une séquence donnée d'éléments, telle que la séquence de mots dans une phrase, en une autre séquence [9].

Il peut être étendu pour résoudre des tâches différentes, comme : résumé de texte, réponse aux questions, classification, résolution d'entité nommée, similitude de texte, détection de message offensant/de blasphème, compréhension des requêtes des utilisateurs etc.

### 2.4.2. Architecture du Transformer

L'innovation derrière Transformers se résume à trois concepts principaux : Encodage Positionnels, Attention, L'auto-attention.

#### 2.4.2.1. Encodages positionnels

L'idée est de prendre tous les mots de la séquence d'entrée et d'ajouter a chaque mot un numéro dans son ordre, Ainsi, vous alimentez votre réseau une séquence comme :

[("Dale", 1), ("says", 2), ("hello", 3), ("world", 4)].

Au début, avant que le Transformer n'ait été formé sur des données, il ne sait pas comment interpréter ces encodages positionnels. Mais à mesure que le modèle voit de plus en plus d'exemples de phrases et de leurs encodages, il apprend à les utiliser efficacement [15].

#### 2.4.2.2. Attention

L'attention est une structure de réseau de neurones. Elle a été introduite dans le cadre de la traduction, en 2015.

L'attention est un mécanisme qui permet à un modèle de texte de « regarder » chaque mot de la phrase d'origine lorsqu'il prend une décision sur la façon de traduire les mots dans la phrase de sortie. Voici la Figure 11 d'exemple de mécanisme d'attention :

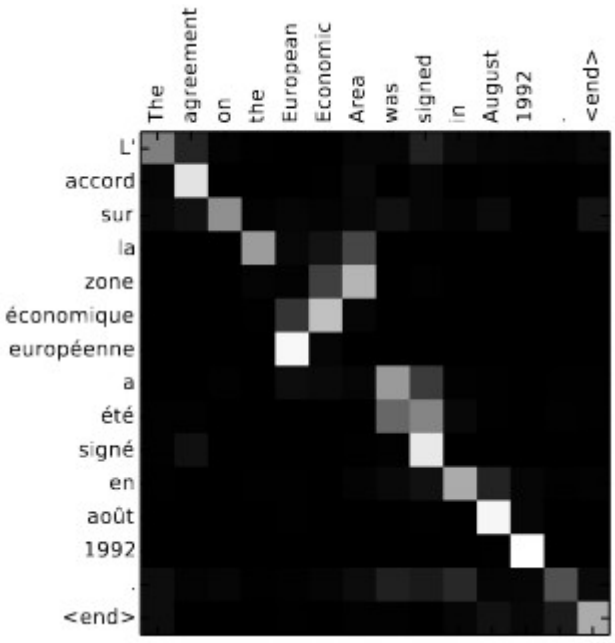



**Figure 11 : Exemple de mécanisme d'attention [20]**

Le mécanisme d'attention est un outil extrêmement utile pour le traitement du langage naturel depuis sa découverte, mais dans sa forme originale, il a été utilisé avec les réseaux de neurones récurrents. Ainsi, l'innovation du papier Transformers 2017 consistait, en partie, à abandonner complètement les RNN [15].

#### 2.4.2.3. L'auto-attention

L'élément final du Transformateur est une distorsion de l'attention, appelée « auto-attention ». Type d'attention, mais plutôt pour créer un modèle qui comprend les schémas sous-jacents de sens et de langage, une tâche langagière qui peut être utilisée pour réaliser un certain nombre de tâches linguistiques.

Ce qui rend les réseaux de neurones puissants et passionnants, c'est qu'ils créent souvent automatiquement des représentations internes significatives des données sur lesquelles ils sont formés. Par exemple, lorsque vous examinez les couches d'un réseau de neurones visuels, vous trouverez un ensemble de neurones qui "reconnaissent" les contours, les formes et même les structures de niveau supérieur comme les yeux et la bouche. Les modèles entraînés sur des données textuelles peuvent automatiquement apprendre des parties du discours, des règles de grammaire et savoir si les mots sont des synonymes.

Plus la représentation interne du langage qu'un réseau de neurones apprend est bonne, mieux il sera dans n'importe quelle tâche linguistique. Et il s'avère que l'attention peut être un moyen très efficace de le faire, si elle est activée sur le texte d'entrée lui- même.

L'auto-attention aide les réseaux de neurones à lever l'ambiguïté des mots, à baliser des parties du discours, à résoudre des entités, à apprendre des rôles sémantiques [15].

La *Figure 12 re*présente l'architecture du Transformer :

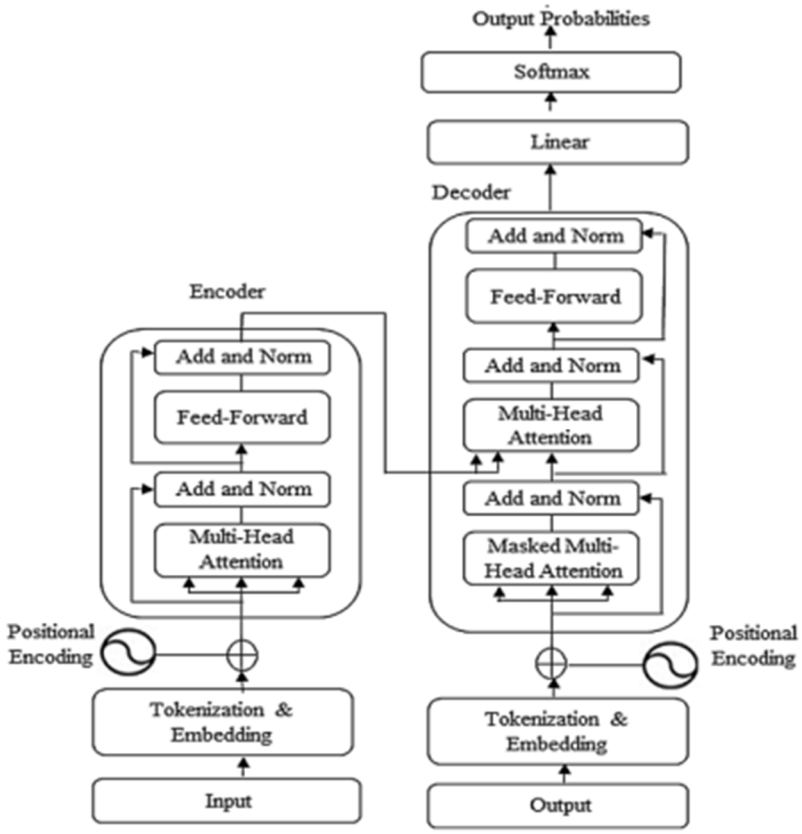



**Figure 12 : Architecture du transformer [9]**

Le Transformer est composé d'un encodeur et un décodeur :

***Encodeur :*** L'encodeur se compose de deux sous-couches sont : *multi head attention* et *positionwise feed forward* . [15]

*PositionWise Feed-Forward* Networks Le réseau d'anticipation de position transforme la représentation à toutes les positions de séquence en utilisant le même MLP. C'est pourquoi nous l'appelons *position wise* .

***Décodeur :*** En plus des deux modules de l'encodeur, le décodeur insère un troisième module juste avant l'auto-attention. Ce sous-module modifie l'existence auto- attention pour ignorer les jetons suivants. Par conséquent, le décodeur est considéré comme unidirectionnel. L'auto-attention ultérieure est également modifiée pour devenir un encodeur-décodeur mécanisme d'attention tel qu'il assiste sur l'encodeur. Le décrit précédemment la connexion résiduelle et la normalisation de couche sont appliquées après chaque couche [15].

## 2.5. Les modèles de traitement du langage naturel basés sur des transformateurs

Voici quelques modèles de NLP pour l'approche abstractive utilisant les réseaux de neurones dans le cadre de l'apprentissage profond à l'aide des transformers :

### 2.5.1. BERT

BERT est un framework d'apprentissage automatique open source pour le NLP. Il est conçu pour aider les ordinateurs à comprendre la signification d'un langage ambigu dans un texte en utilisant le texte environnant pour établir le contexte.

BERT est basé sur Transformers, un modèle d'apprentissage en profondeur dans lequel chaque élément de sortie est connecté à chaque élément d'entrée, et les poids entre eux sont calculés dynamiquement en fonction de leurs connexions. (En PNL, ce processus est appelé attention).

Historiquement, les modèles de langage n'ont pu lire la saisie de texte que de manière séquentielle, soit de gauche à droite, soit de droite à gauche, mais pas les deux. BERT est différent car il est conçu pour lire dans les deux sens en même temps. Cette capacité, rendue possible par l'introduction d'un transformateur, est appelée bidirectionnalité.

BERT excelle dans plusieurs fonctionnalités qui rendent cela possible, notamment :

- Tâches de génération de langage basées sur des séquences telles que : répondre aux questions, Résumé récapitulatif, Prédiction de phrases, Générer des réponses de dialogue.
- Tâches de compréhension du langage naturel telles que : résoudre l'ambiguïté et la coréférence (mots qui se ressemblent ou se ressemblent mais ont des significations différentes), Désambiguïsation du sens des mots, Raisonnement en langage naturel, Classification des sentiments.

Le BERT devrait avoir un impact majeur sur la recherche vocale et la recherche de texte, et la technologie NLP de Google a jusqu'à présent été sujette à des erreurs. Le BERT devrait également améliorer considérablement le référencement international, car sa compréhension du contexte l'aide à interpréter les modèles partagés par différentes langues sans comprendre pleinement la langue. Plus largement, BERT a le potentiel d'améliorer considérablement les systèmes d'IA à tous les niveaux.[19]

### 2.5.2. GPT-2

GPT-2 est un grand modèle de langage basé sur un transformateur qui prédit le mot suivant en fonction de tous les mots précédents dans le texte. La diversité des ensembles de données rend cette cible simple englobant des démonstrations naturelles de nombreuses tâches dans divers domaines. GPT-2 est une mise à l'échelle directe de GPT, avec plus de paramètres et formé sur plus de la quantité de données.

GPT-2 présente un large éventail de fonctionnalités, y compris la capacité de générer des échantillons de texte synthétisés de manière conditionnelle d'une qualité sans précédent, Il faut initialiser le modèle avec l'entrée et le laisser produire de longues continuations. De plus, GPT-2 surpasse les autres modèles de langage formés sur des domaines spécifiques, par exemple : "les livres", sans utiliser ces ensembles de données de formation spécifiques à un domaine. Pour les tâches linguistiques telles que la réponse aux questions, la compréhension de la lecture, la synthèse et la traduction, GPT-2 commence à apprendre ces tâches à partir de texte brut sans utiliser de données de formation spécifiques à la tâche. Bien que les scores de ces tâches en aval soient loin d'être à la pointe de la technologie, ils montrent que, avec suffisamment de données et de calculs (non étiquetés), ces tâches peuvent bénéficier de techniques non supervisées. [22]

### 2.5.3. GPT-3

GPT-3 est un modèle de prédiction de langage. Cela signifie qu'il dispose d'un modèle d'apprentissage automatique de réseau neuronal qui peut prendre le début du texte comme entrée et le transformer en ce qu'il prédit comme étant le résultat le plus utile. Ceci est accompli en formant le système sur le vaste corps de texte Internet pour repérer les modèles. Plus précisément, GPT-3 est la troisième version d'un modèle axé sur la génération de texte basée sur une pré-formation sur une énorme quantité de texte.

Lorsqu'un utilisateur fournit une entrée de texte, le système analyse la langue et utilise un prédicteur de texte pour créer la sortie la plus probable. Même sans beaucoup de réglage ou de formation supplémentaire, le modèle génère un texte de sortie de haute qualité qui ressemble à ce que les humains produisent. [3]

### 2.5.4. BART

BART est un modèle séquence à séquence formé comme un auto-encodeur de débruitage. Cela signifie qu'un modèle BART affiné peut prendre une séquence de texte en entrée et produire une séquence de texte différente en sortie. Ce type de modèle est pertinent pour la traduction automatique, le question-réponse, la synthèse de texte, ou la classification des séquences (catégorisation des phrases ou des jetons de texte d'entrée).

Les auteurs de BART décident d'utiliser certaines techniques de bruit existantes et de nouvelles techniques de préformation. Les schémas de bruit qu'ils utilisent sont :

Masquage de jeton :

- les jetons aléatoires dans une phrase sont remplacés par [MASQUE]. Le modèle apprend à prédire le jeton unique en fonction du reste de la séquence.
- Suppression de jetons : les jetons aléatoires sont supprimés. Le modèle doit apprendre à prédire le contenu du jeton et à trouver la position à partir de laquelle le jeton a été supprimé.
- Remplissage de texte : un nombre fixe de jetons contigus sont supprimés et remplacés par un seul jeton [MASQUE]. Le modèle doit apprendre le contenu des jetons manquants et le nombre de jetons.
- Permutation des phrases : les phrases (séparées par des points) sont permutées de manière aléatoire. Cela aide le modèle à apprendre l'implication logique des phrases.
- Rotation du document : le document est réorganisé pour commencer avec un jeton aléatoire. Le contenu avant le jeton est ajouté à la fin du document. Cela donne un aperçu de la façon dont le document est généralement organisé et à quoi ressemble le début ou la fin d'un document.

Dans l'architecture de BART, la première partie de BART utilise l'encodeur bidirectionnel de BERT pour trouver la meilleure représentation de sa séquence d'entrée, qui a été formé à l'aide d'une simple technique de masquage de jetons. BART renforce l'encodeur BERT en utilisant des types de mécanismes de masquage plus difficiles dans sa préformation,

lorsqu’on a la représentation au niveau du jeton et de la phrase d'une séquence de texte d'entrée, le décodeur GPT doit les interpréter pour les mapper avec la cible de sortie [16]

### 2.5.5. T5

L'idée derrière le modèle T5 est l'apprentissage par transfert. Le modèle est initialement formé à l'apprentissage par transfert sur des tâches contenant un texte volumineux, puis affiné sur des tâches en aval afin que le modèle acquiert des compétences générales et des informations à appliquer à des tâches telles que la synthèse T5 à l'aide de la génération de séquence à séquence. L’encodeur de modèle T5 prend en entrée une séquence de jetons qui correspondent à une séquence d'ensembles. Dans le bloc codeur existe un bloc avec deux sous-composants, la couche d'auto-attention et le réseau d'anticipation. Le décodeur et l'encodeur sont en structure, sauf qu'il existe un mécanisme d'attention généralisé après chaque auto- couche d'attention. Cela permet au modèle de fonctionner uniquement sur les sorties précédentes. Le final Le bloc décodeur produit une sortie qui est introduite dans une autre couche. Cette dernière couche est une couche dense où la fonction d'activation est softmax. Les poids de la sortie de cette couche sont introduits dans la matrice d'intégration d'entrée. [17]

### 2.5.6. PEGASUS

Dans ce modèle, les lignes significatives sont supprimées du texte d'entrée et compilées dans des sorties distinctes. En outre, il est préférable de choisir uniquement des phrases pertinentes plutôt que de choisir des phrases au hasard. Cette méthode est la méthode préférée pour résumer avec la méthode abstractive car elle est similaire à la tâche d'interprétation de l'intégralité du document et de génération de résumés. Il est utilisé pour former le modèle de transformateur basé sur des données textuelles produit le modèle PEGASUS. [18]

## 2.6. Comparaison entre les modèles

Pour faire le résumé abstractif, nous avons fait des comparaisons entre les modèles concernant ce type de résumé, voici un tableau (*Tableau 1)* comparatif qui présente les caractéristiques de ces modèles

**Tableau 1 : les caractéristiques des modèles transformers**

| | Model de base | Les langues | Caractéristique | Données de préformation | Nombre paramètres | Taille du vocabulaire | Architecture | Pré-entraineme nt |
|---|---|---|---|---|---|---|---|---|
| **BERT** | Encodeur | a des modèles pour différentes langues | Apprendre le contexte d'un mot en fonction de tout son environnement (gauche et droite du mot). | A été formé sur 16GO de données textuelles | 1270M | 30K | Bidirectionnelle | non supervisé |
| **GPT-2** | Décodeur | disponible que pour l'anglais, mais peut être affiné sur des corpus spécifiques | il peut prédire le mot suivant | A été formé sur 40 GO de données textuelles | 1.5mds | 50K | unidirectionnell | non supervisé |
| **GPT-3** | Décodeur | Anglais, allemand, Russe et japonais | A une architecture similaire à GPT-2 mais est encore plus grand | A été formé sur un ensemble de données open source appelé « Common Crawl » et d'autres textes d'openAI tels que des entrées de Wikipédia | 175mds | / | Unidirectionnell e | non supervisé |
| **BART** | Encodeur-Décodeur | Plus de 70 langues | Combiner entre l'architecture de BERT et GPT | Formé sur un ensemble de données de 160Go | 400M | 29K | Encodeur bidirectionnel et décodeur autoregressif | non supervisé |
| **T5** | Encodeur-Décodeur | 110 langues | Format de text en text | Formé sur un ensemble de données de 7To | 11mds | 32K | Bidirectionnel, unidirectionnell e et Seq2Seq | non supervisé et supervisées |
| **PEGASUS** | Encodeur-Décodeur | Plus de 70 langues | Générration des phrases vides utilise MLM et GSG | / | 568M | 500k | Bidirectionnel, unidirectionnell e et Seq2Seq | Auto Supervisé |

## 2.7. Réglage fin

Le réglage fin est une façon d'appliquer ou d'utiliser l'apprentissage par transfert. Plus précisément, le réglage fin est un processus qui prend un modèle qui a déjà été formé pour une tâche donnée, puis réglé le modèle pour lui faire effectuer une tâche similaire.

En supposant que la tâche d'origine est similaire à la nouvelle tâche, l'utilisation d'un réseau de neurones artificiels déjà conçu et formé nous permet de tirer parti de ce que le modèle a déjà appris sans avoir à le développer à partir de zéro.

Pour affiner notre modèle nous avons changé quelque paramètre pour améliorer sa puissance. Par exemple, nous devons choisir le nombre de couches que nous utilisons, les

types de couches que nous utilisons, l'ordre dans lequel placer les couches, le nombre des nœuds à inclure dans chaque couche, décider du degré de régularisation à utiliser, quoi définir notre taux d'apprentissage, des exemples de réglage fin comme changer :

- Nombre de couches.
- Types de couches.
- Ordre des calques.
- Nombre de nœuds dans chaque couche.
- Quelle quantité de régularisation utiliser.
- Taux d'apprentissage.

C'est ce qui rend l'approche de réglage fin si attrayante. Si nous pouvons trouver un modèle formé qui fait déjà bien une tâche, et que cette tâche est similaire à la nôtre au moins d'une manière éloignée, alors nous pouvons tirer parti de tout ce que le modèle a déjà appris et l'appliquer à notre tâche spécifique.

Dans notre cas, nous avons éliminé quelque couche avec *Freezing layer*. Cette technique consiste à :

- Rendre les couches non entraînables.
- accélérer la formation du réseau de neurones en gelant progressivement les couches cachées.
- Réduire le temps de calcul pour l'entraînement tout en ne perdant pas grand-chose du côté de la précision.

## 3.8. Les méthodes d'évaluation des résultats

### 3.8.1. Évaluation des résumés automatiques avec ROUGE

La métrique la plus couramment utilisée pour l’évaluation de résumé automatique est ROUGE [25]. Cette métrique est utilisée en traduction et résumé automatique, en cela qu’elle mesure la similarité lexicale entre un texte et son résumé.

En d’autres termes, ROUGE mesure la proportion de mots en commun au sein du résumé et de la référence. ROUGE mesure également les proportions de suites de n mots que l’on appelle n-grammes. On note ces mesures ROUGE-n où n est le nombre de mots considérés.

Ainsi, ROUGE-1 mesure les proportions de mots, ROUGE-2 mesure les proportions de paires de mots (des bigrammes, ou 2-grammes) etc.

Soit D un document de N mots D = {$w_1, \ldots, w_N$ }, on a l'ensemble des n-grammes de D :

$$gram_n(D) = \{(w_1, \ldots, w_n),(w_2, \ldots, w_{n+1}), \ldots, (w_{N-n+1}, \ldots, w_N)\} \quad (1.1)$$

En particulier, on peut dénombrer le nombre de n-grammes en commun pour un résumé automatique S et sa référence R :

$$match_n(S, R) = |gram_n(S) \cap gram_n(R)| \quad (1.2)$$

On peut alors choisir de diviser le nombre de n-grammes communs par le nombre de mots dans le résumé automatique, c'est ce qu'on appelle la précision (1.3), ou par le nombre de mots du résumé de référence (1.4).

Enfin, la F-mesure (F) est la moyenne harmonique entre la précision et le rappel (1.5).

$$ROUGE - n(P) = \frac{match_n(S,R)}{gram_n(S)} \quad (1.3)$$

$$ROUGE - n(R) = \frac{match_n(S,R)}{gram_n(R)} \quad (1.4)$$

$$ROUGE - n(F) = 2 \times \frac{ROUGE-n(P) \times ROUGE-n(R)}{ROUGE-n(P) + ROUGE-n(R)} \quad (1.5)$$

Une autre variante est couramment utilisée : ROUGE-L, qui mesure la longueur de la plus grande sous-séquence commune. De même que pour ROUGE-n, on peut utiliser la précision en divisant par la longueur du résumé automatique.

- le rappel en divisant la par longueur du résumé de référence.
- la F-mesure, la moyenne harmonique des deux.

Dans le cadre de l'évaluation de résumés automatiques, les métriques les plus couramment utilisées sont ROUGE-1(F), ROUGE-2(F) et ROUGE-L(F).

### 3.8.2. Limites de la métrique ROUGE

L'usage de la métrique ROUGE pour l'évaluation de résumé abstractif est critiqué dans la mesure où ne sont pris en compte que les correspondances lexicales entre les mots. En

d’autre terme, un mot qui ne serait pas le même que celui du résumé de référence sera pénalisé, qu’il soit très proche en sens ou au contraire très éloigné.

En particulier, ceci pénalise les reformulations ou l’usage de synonymes qui sont pourtant des comportements pertinents au sein d’un résumé abstractif. Ceci est particulièrement problématique du fait qu’il n’existe pas un unique résumé pertinent, mais que plusieurs alternatives sont possibles.

En réponse, la métrique Meteor [26] propose de prendre en compte les synonymes dans la mesure du score. Pour autant, cette métrique ne s’est pas imposée dans la communauté du résumé automatique.

### 3.8.3. Mesure de l’abstractivité des résumés

Pour les travaux sur le résumé abstractive, il est pertinent d’évaluer également dans quelles mesures les résumés générés sont effectivement abstractifs. En particulier, on peut mesurer la propension du modèle générer de nouveaux mots (i.e. absent de la source) ou à l’inverse mesurer le taux de copie. En particulier, on attend d’un modèle abstractif que son taux de copie soit proche de celui observé au sein des données de références. On peut mesurer le taux de copie avec la métrique ROUGE-n(P) du résumé par rapport au document source (à la place du résumé de référence lors de l’évaluation) [27].

### 3.8.4. Évaluation Humaine

Compte tenu des critiques apportées aux métriques automatiques, il est intéressant de soumettre les méthodes de résumé automatique à l’évaluation humaine.

## 2.9. Conclusion

Dans ce chapitre, nous avons présenté les différentes approches utilisées pour faire le résumé, ensuite nous avons exploré quelques méthodes qui ont été proposées pour résoudre le problème de production de résumés automatiques. Nous avons parlé aussi de la nouveauté de RNN “ les transformer” qui sont les plus utilisés pour faire des résumés de texte automatique dernièrement, nous avons cité ces derniers par leurs caractéristiques et leur fonctionnement.

# Chapitre 3

# Conception et développement

## 3.1. Introduction

La synthèse a été continuée d'être un sujet de recherche brûlant dans le domaine de la science des données. Le résumé est une technique qui réduit la taille d'un document tout en préservant son sens. C'est l'un des domaines les plus étudiés au sein de la communauté du traitement automatique du langage naturel (NLP).

Après avoir introduit les approches neuronales pour le Traitement Automatique du Langage Naturel au chapitre 1 et 2, nous en présentons dans ce chapitre les applications dans le domaine du résumé automatique de texte.

Lorsque vous souhaitez que l'apprentissage automatique transmettre le sens d'un texte, il peut reformuler les informations ou simplement vous montrer les parties les plus importantes du contenu. Ca se que nous appelons le résumé abstractif.

Les modèles d'apprentissage en profondeur de pointe actuels tels que GPT-3, GPT-2, BERT, etc. nous aident à générer des résumés humanisés paraphrasés en termes de lisibilité. Nous allons concentrer sur l'obtention de résultats acceptables avec cette dernière approche.

## 3.2. L’environnement de développement

Nous présenterons l’environnement de travail qui inclut les outils de développement (logiciels et technologies exploités). Nous allons ensuite décrire les différents langages utilisés dans ce projet.

### 3.2.1 Google colab

Colaboratory, souvent raccourci en "Colab", est un produit de Google Research. Colab permet à n'importe qui d'écrire et d'exécuter le code Python de son choix par le biais du navigateur. C'est un environnement particulièrement adapté à la machine learning, à l'analyse de données et à l'éducation.

Colab permet :

- d’améliorer vos compétences de codage en langage de programmation Python.
- de développer des applications en Deep Learning en utilisant des bibliothèques Python populaires telles que Keras, TensorFlow, PyTorch, OpenCV et Transformers.
- d’utiliser un environnement de développement (Jupyter Notebook) qui ne nécessite aucune configuration.
- disponible gratuitement.
- utilise le GPU comme environnement virtuel en ligne.

### 3.2.2. Python

Python est le langage de programmation open source le plus employé par les informaticiens. Ce langage s’est propulsé en tête de la gestion d’infrastructure, d’analyse de données ou dans le domaine du développement de logiciels. En effet, parmi ses qualités, Python permet notamment aux développeurs de se concentrer sur ce qu’ils font plutôt que sur la manière dont ils le font. Il a libéré les développeurs des contraintes de formes qui occupaient leur temps avec les langages plus anciens. Ainsi, développer du code avec Python est plus rapide qu’avec d’autres langages.

### 3.2.3. Hugginge face

Le Hugging Face Hub est une plate-forme avec plus de 50 000 modèles, 5 000 ensembles de données et 5 000 démos dans lesquelles les utilisateurs peuvent facilement collaborer dans leurs flux de travail ML. Le Hub fonctionne comme un lieu central où tout le monde peut partager, explorer, découvrir et expérimenter l'apprentissage automatique open source.

Nous avons créé, entrainé et partagé notre modèle « *Bert2Gpt2* » sur la plateforme Hugging face.

## 3.3. Le modèle Transformer appliqué au résumé

Le modèle Transformer a été développé dans le cadre de la traduction neuronale avant d'être rapidement étendu à d'autres tâches de NLP. En particulier, l'objectif même du modèle Transformer éliminer les calculs séquentiels afin de mieux paralléliser les opérations pertinentes pour le résumé automatique en cela qu'il permet de traiter des séquences d'entrée, et de sortie, plus longues par rapport aux modèles encodeur-décodeur classiques.

Nous proposons également d'effectuer un résumé abstractif qui génère des nouvelles séquences à partie de texte entré. Le résumé abstractif utilise des modèles de traitement du langage naturel (NLP) pour créer des résumés de texte à l'aide des modèles pré-entrainer basé sur l'architecture des transformers. C'est comme si un humain lisait un article et demandait de quoi il s'agissait.

## 3.4. Architecture de notre modèle

Dans notre travail nous avons utilisé l'architecture « Encoder-Decoder Model », nous avons travaillez avec BERT comme un encodeur et GPT-2 comme un décodeur, et les réunis dans un seul model « *BERT2GPT-2* ». Nous avons suivi ces étapes pour la réalisation notre modèle de résumé automatique :

### 3.4.1. Création de modèle

Pour créer un modèle, nous devons passe sur deux phases nécessaires comme elle est présentée dans la *figure13* :

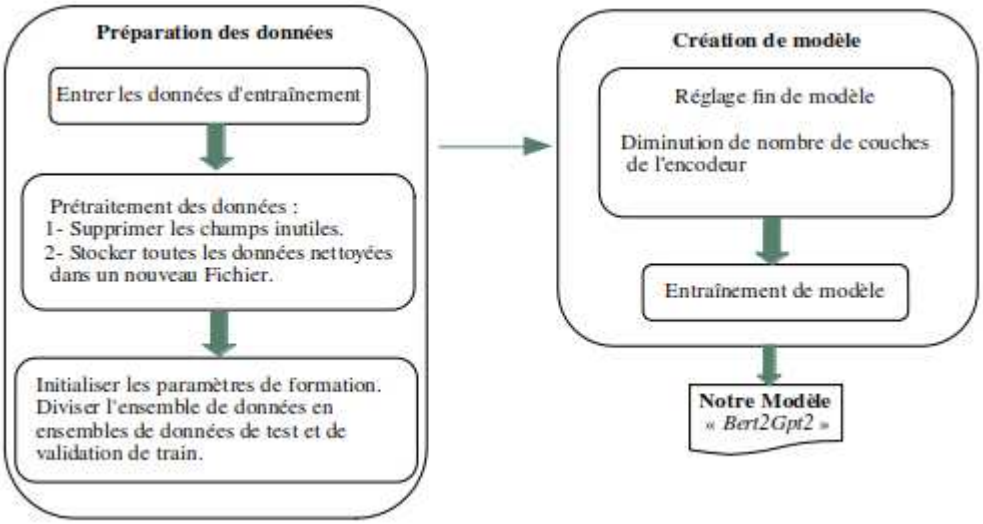



**Figure 13: Création de modèle**

### 3.4.1.1. Préparation des données

Avant de créer un modèle, nous avons préparé un ensemble des données pour l'entrainement, en utilisant les deux célèbres data-set français « *mlsum-fr* » [31] et «*orange-sum*» [29] pour entrainer notre modèle à notre langue choisis.

```
from datasets import load_dataset
dataset = load_dataset("orange_sum",'abstract')
```

**Figure 14 : Chargement de dataset**

Puis, nous avons nettoyé tous les champs et caractères inutiles.

```
def clean(row):
  row['text']= row['text'].replace('\n',' ').replace('\t',' ').replace(',','').replace('\'','').replace('\"','')
  row['text']= " ".join(row['text'].split())
  row['target']= row['target'].replace('&lt;','<').replace('&gt;','>')
  return row
```

**Figure 15 : Nettoyage des data**

Ensuite, nous avons devisé les données en deux parties une pour l'entrainement et une autre de validation.

```
dataset['train'].to_csv('train.csv', index=False, header=True)
dataset['test'].to_csv('test.csv', index=False, header=True)
```

Figure 16 : division des données

#### 3.4.1.2. Création de modèle

Après la bonne préparation des données, nous avons passé à la création de notre modèle. Pour le créer, nous avons choisi de réunir les deux modèles NLP (BERT et GPT2) dans un modèle « Encodeur-Décodeur ».

Nous avons commencé par affiner le modèle encodeur « BERT » par une diminution de nombre des couches « Freezing-layer » afin d'obtenir des meilleurs résultats et diminuer le temps d'entrainement.

```
for name, param in list(model.encoder.named_parameters())[:-50]:
    print('I will be frozen: {}'.format(name))
    param.requires_grad = False
```

Figure 17 : Freezing-layer (réglage fin)

Puis nous avons fait l'entrainement de notre modèle sur les données entrées en sauvegardons tous les résultats d'entrainement pour obtenir un modèle entrainé qui comprend la langue française pour l'utiliser ultérieurement dans la génération de texte français.

```
from transformers import EncoderDecoderModel, BertTokenizer

# creer le bert2gpt2 d'un model BERT and GPT2 .
model = EncoderDecoderModel.from_encoder_decoder_pretrained("bert-base-cased", "gpt2")

# sauvegarder  le modele
model.save_pretrained("bert2gpt2")
```

Figure 18 : Scripte de création de modèle

### 3.4.2. Génération de résumé automatiquement

La *figure 14* représente l'utilisation de notre modèle entrainé :

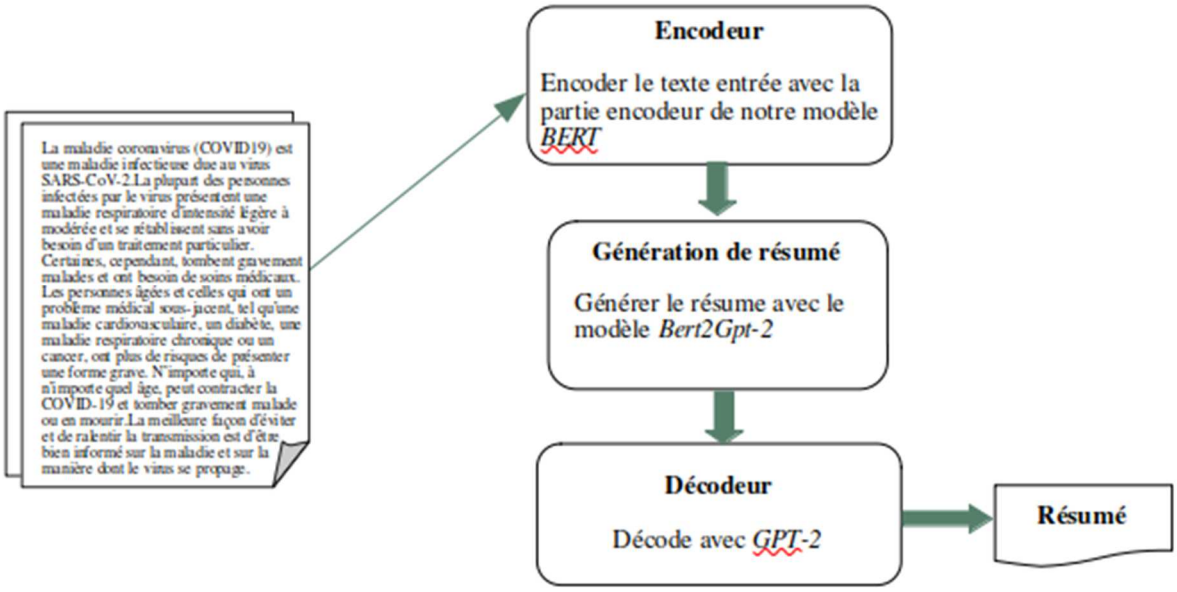



**Figure 19 : Génération de résumé avec notre modèle**

- **Encodeur :** Dans notre cas, nous avons utilisé le modèle pour la génération de résumé automatique, pour l'utiliser nous devons entrée un texte français, et le passer au encodeur « BERT » pour l'encodé, il transforme la phrase en vecteurs pour que la machine les comprend.
- **Génération de résumé :** Dans cette tâche, nous utilisons le modèle « *Bert2Gpt2* » pour générer un résumé en précisant les paramètres de génération.
- **Décodeur :** la dernière étape est de décodé le résultat obtenu de la génération pour obtenir un résumé lisible.

```
from transformers import RobertaTokenizerFast, EncoderDecoderModel

#initialisation de tokenizer
tokenizer = RobertaTokenizerFast.from_pretrained("Chemsseddine/bert2gpt2SUMM-finetuned-mlsum")
#aider les token special
tokenizer.bos_token = tokenizer.cls_token
tokenizer.eos_token = tokenizer.sep_token
#initialisation du modele
model = EncoderDecoderModel.from_pretrained("Chemsseddine/bert2gpt2SUMM-finetuned-mlsum")
```

**Figure 20 : Appel à notre modèle**

Après l'appel au modèle, on l'utilise pour la génération du résumé :

```python
# generate summary
def generateSumm(txt):
  # encoder le texte entrée
  input_ids = tokenizer.encode(txt, return_tensors='pt')
  #generation de resume a l'aide de texte encodé
  summary_ids = model.generate(input_ids,#le texte encodé
            min_length=80,#la longuer minimum du sequence de sortie
            max_length=200,#la longuer maximale du sequence de sortie
            num_beams=10,
            repetition_penalty=2.5,
            length_penalty=1.0,
            early_stopping=True,
            no_repeat_ngram_size=2,#Pour éviter les répétitions du même texte,
            use_cache=True,
            do_sample = True,
            temperature = 0.8,
            top_k = 50,
            top_p = 0.95)
  #decodé la sequence de generé par le modele
  summary_text = tokenizer.decode(summary_ids[0], skip_special_tokens=True)
  return summary_text
```

**Figure 21 : Génération du résumé**

## 3.6.Évaluation avec ROUGE

Nous avons travaillé sur quelques modèles pour savoir la précision de ses derniers dans la génération de résumé, Nous avons entrainé les modèles suivant a notre dataset et utilisé la métrique ROUGE pour l'évaluation :

- ***BARThez :*** est le premier modèle seq2seq pré-entraîné à grande échelle pour le français. Basé sur BART, BARThez est particulièrement adapté aux tâches génératives. [29]
- ***PEGASUS :*** Dans PEGASUS, les phrases importantes sont supprimées/masquées d'un document d'entrée et sont générées ensemble comme une séquence de sortie à partir des phrases restantes, similaire à un résumé extractif. *[18]*
- ***T5 :*** Ce modèle a été affiné en français pour la synthèse abstraite de texte.

```
from datasets import load_metric
metric = load_metric("rouge")
fake_preds = ["comment ", "ca va"]
fake_labels = ["comment", "ca va"]
metric.compute(predictions=fake_preds, references=fake_labels)
```

**Figure 22 : Exemple d'applications de la métrique ROUGE**

**Tableau 2 : comparaison avec la métrique ROUGE entres les modèles entrainé**

| | **Rouge-1** | **Rouge-2** | **Rouge-L** | **Rouge-LSum** |
|---|---|---|---|---|
| ***BARThez*** | 24.2547 | 10.0483 | 19.7 | 20.0966 |
| ***PEGASUS*** | 25.31 | 9.613 | 19.08 | 19.55 |
| ***T5*** | 20.2108 | 7.8633 | 16.9554 | 17.3178 |
| ***NotreModele*** | 25.7023 | 8.5872 | 18.6776 | 19.821 |

Dans ce tableau comparatif, nous avons entrainé ces modèles avec *mlsum*, *orangeSum* et nos données médicales. Pour les comparer avec notre modèle, s'il apprend bien les articles médicaux français, pour réaliser des bons résumés abstractifs, et savoir ces précisions.

Nous pouvons voir que la précision de notre modèle est élevé par rapport aux autres modèles pré-entrainé au texte français comme (Barthez, T5, Pegasus), cette précision peut être augmente si nous entrainons de nouveau notre modèle sur des nouvelles données, dû la forte précision de notre modèle nous pouvons conclure que notre modèle a une priorité aux autres et il est prêt à faire la tache de résumé automatique de texte français.

## 3.7. Implémentation d'un Prototype

Nous avons implémenté une interface pour faciliter l'utilisation de notre modèle pour faire un résumé du texte abstractive.


**Figure 23 : Page principale de notre application**

La *figure 23* présente notre interface qui a le but à faire le résumé abstractif elle se compose de :

- Deux champs de texte le premier est pour le texte entré et le deuxième pour le résumé obtenu.
- Un bouton nettoyer aider l'utilisateur a supprimé le texte du champ.
- Un bouton soumettre quand nous cliquons sur ce bouton l'application commence a généré le résumé automatique.
- Deux glisseur un pour la longueur maximale et le deuxième pour la longueur minimale.

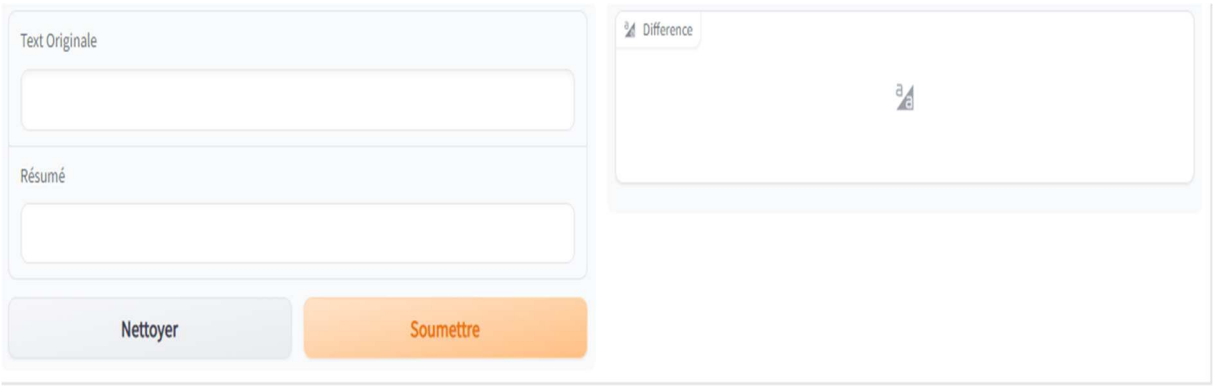



**Figure 24 : Page de comparaison des résultats avec le texte original**

Dans la *figure 24* la deuxième tabulation de notre interface dans cette interface nous pouvons comparer le résultat obtenu avec le texte original pour savoir comment le programme a fait le résumé automatique.

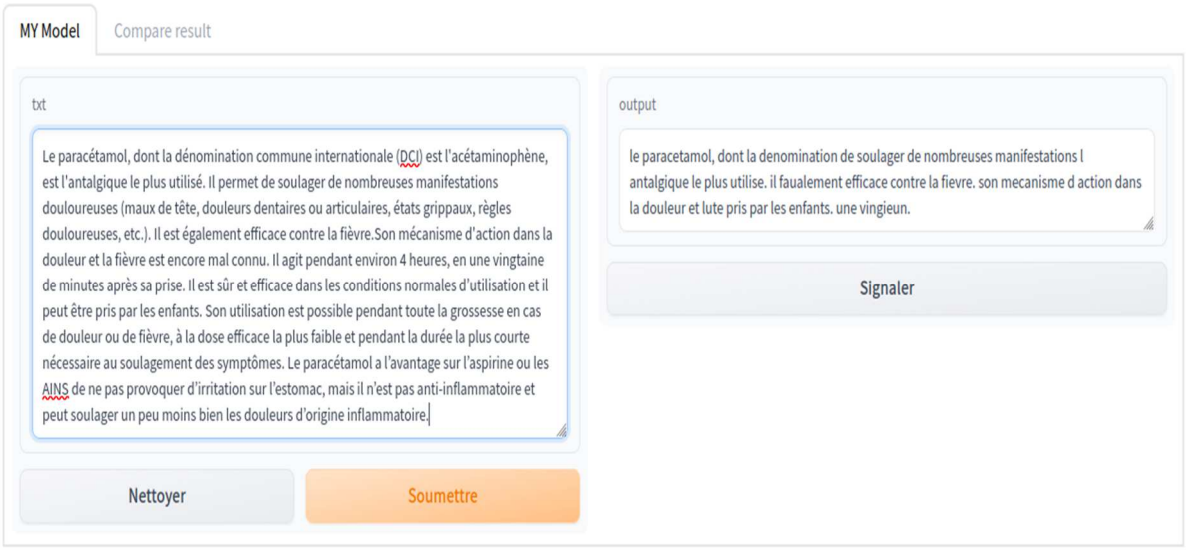



**Figure 25: Page principale de notre application (exemple d'application)**

Dans la *figure 25* nous avons présenté un exemple de résumé de texte entré à l'aide de notre interface et dans le deuxième coté son résumé.


**Figure 26 : Page de comparaison des résultats avec le texte original (Exemple d'application)**

Dans la capture précédente nous avons montré la comparaison entre le texte entré et le texte de sortie et pour savoir la différence entre les deux nous avons définis les mots originale supprime par le modèle par un moins (-) en rouge et les phrase modifié en vert.

## 3.7. Conclusion

Nous avons présenté dans ce chapitre les logiciels utilisé pour faire notre projet ainsi que le plan de travail expliqué étape par étape tel que la création de notre modèle et l'entrainement leur architecture est paramétrage nous avons fait dans ce chapitre aussi le réglage fin pour affiner notre modèle pour faciliter l'entrainement,

Nous avons présenté les différentes méthodes d'évaluation de résume automatique et les utilisons dans notre implémentation pour savoir la précision de notre résultat,

Nous avons vu aussi les différentes méthodes d'évaluation de résumé automatique, nous avons présenté des captures de notre application contenant l'interface qui consiste à faire le résumé et la deuxième est utilisée pour comparer le texte source avec le résumé généré s'il est similaire lexicalement avec son texte source. Nous finissons par la présentation de notre interface implémentée avec des explications sur leurs composants et comment l'utilisateur peut l'utiliser facilement.

# Conclusion et perspectives

Notre monde est parachuté par la collecte et la diffusion d'énormes quantités de données, et ça pose un vrai problème dû à l'augmentation anormale de ces documents. Parfois, ils sont inutiles et prennent beaucoup de temps à lire. Sous l'influence de la nécessité de consulter rapidement un grand nombre de documents comme le domaine médical "covid-19" qu'il est trop urgent et sensible, donc l'idée d'assigner la tâche de résumer des documents à des machines plutôt qu'à des humains devient de plus en plus urgente aujourd'hui.

## Travail réalisé

Nous avons vu que les modèles de langage basé sur Transformer, tels que BERT et GPT-2, qui ont été préformés sur de grands ensembles de données, peuvent être facilement affinés pour obtenir de bons résultats pour un résumé abstractif en utilisant uniquement un minimum de données. Cette approche exploite la puissance de l'apprentissage par transfert qui a été observé sur de nombreuses autres tâches de traitement du langage naturel avec les architectures Transformer. Étant donné que cette approche nécessite le minimum de données, elle peut être appliquée dans divers autres domaines étroits et langages à faibles ressources. Les résumés produits par l'approche proposée sont cohérents avec les documents d'entrée (dans la plupart des cas) et ont une grande fluidité, comme prévu d'un modèle basé sur GPT (bien qu'il y ait des problèmes avec l'exactitude factuelle de certains résumés générés). Des travaux récents d'OpenAI et Salesforce suggèrent qu'il s'agit d'un problème prédominant indépendant des modèles de résumé abstractif.

Pour cela, nous avons réalisé ce travail qui consiste à faire une recherche sur le traitement de langage naturel, plus précisément le résumé abstractif de texte français. Nous avons récolté tous les informations lié à ce domaine pour lancer notre travail et le renforcer, nous avons commencé par la récolte des données d'entrainement, en suite la création et l'entrainement de notre modèle et finissons par le partager sur la plateforme hugging face pour rendre son appel et son utilisation simple dans la génération résumé.

Nous avons obtenu comme résultat un model Encodeur-Décodeur composer de deux modèle : l'encodeur Bert et le décodeur GPT2. Ce modèle peut réaliser le résumé de texte abstractive, nous avons testé les précisions des autres modèles Transformer pour les

comparés avec la précision de notre modèle et cela veut dire que notre modèle est bon pour le traitement de résumé automatique du texte français.

## Perspectives et travaux à venir

Nous avons testé notre model avec les différents métrique d'évaluation des différents modèles NLP, les résultats étaient satisfaisants. Dans le futur, on compte d'obtenir des très bons résultats d'évaluation par l'agrandissement et l'enrichissement de notre modèle avec des nouveaux données médicales pour qu'il apprend bien ce domaine afin d'avoir des résumé cohérent, et utilisé aussi des nouveaux réglages fin pour améliorer l'entrainement du modèle.

Enfin, du fait que notre modèle peut être d'ordre général, des expériences devraient être menées sur d'autres domaines (article de football, de presse, etc.) et d'autres langues (arabe, allemande, etc.).

# Références bibliographiques


[1] Ethem alpaydin : Introduction in Machine learning : Fourth Edition / Cambridge, Massachusetts : the MTT press, 2020.

[2] Gregory Gelly : Réseaux de neurones récurrents pour le traitement automatique de la parole, 2017.

[3] Robert Dale: Natural language generation: The commercial state of the art in 2020, Natural Language Engineering Journal: 27, 2021.

[4] Chloé-Agathe Azencott: Introduction au Machine Learning, 2012.

[5] Claude Touze : Les réseaux de neurones artificiels, introduction au connexionnisme, 2016.

[6] El Mahdi Brakni: Réseaux de neurones artificiels appliqués a la méthode électromagnétique transitoire InfiniTEM, 2011.

[7] Paul Tardy : Approches neuronales pour le résumé abstractif de transcriptions de parole, 2021

[8] Klaas Landsman: The Fine-Tuning Argument: Exploring the Improbability of Our Existence, 2016.

[9] Vaswani A, Shazeer N, Parmar N, Uszkoreit J, Jones L, Gomez AN, Kaiser L, Polosukhin I : Attention is All you Need. ArXiv abs/1706.03762, 2017.

[10] Atif Khan and Namoie Salim : A review on abstractive summarization methods : Journal of Theoretical and Applied, 2014.

[11] Nesreen Al-sharman, B.S., M.S : Automatic Text Summarizations, 2018.

[12] Narendra Andhale, L.A. Bewoor : An Overview of Text Summarization Techniques, 2016.

[13] N. Moratanch, Dr. S. Chitrakala : A Survey on Abstractive Text Summarization. International Conference on Circuit, Power and Computing Technologies [ICCPCT], 2016.

[14] Filippo Maria Bianchi , Enrico Maiorino, Michael C. Kamp□meyera, Antonello Rizzi, Robert Jenssen : An overview and comparative analysis of Recurrent Neural Networks for Short Term Load Forecasting, 2017.

[15] Victor Risne, Adele Siitova: Text summarization using transfer learning Extractive and abstractive summarization using BERT and GPT-2 on news and podcast data, 2019.

[16] Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Ves Stoyanov, Luke Zettlemoyer : BART: Denoising Sequence-to-Sequence Pre-training for Natural Language Generation, Translation, and Comprehension, Facebook AI, 2020.

[17] Anushka Gupta, Diksha Chugh, Anjum, Rahul Katarya : Automated News Summarization Using Transformers,Delhi Technological University, New Delhi,India . 2021.

[18] Jingqing Zhang, Yao Zhao, Mohammad Saleh, Peter J. Liu : PEGASUS: Pre-training with Extracted Gap-sentences for Abstractive Summarization, 2019.

[19] Jiacheng Xu , Zhe Gan , Yu Cheng , Jingjing Liu : Discourse-Aware Neural Extractive Text Summarization ,The University of Texas at Austin 2 Microsoft Dynamics 365 AI Research, 2020.

[20] Dzmitry Bahdanau, Kyunghyun Cho, Yoshua Bengio : Neural Machine Translation by jointly learning to align and translate, 2015.

[21] Yang Liu and Mirella Lapata : Text Summarization with Pretrained Encoders , Institute for Language, Cognition and Computation, School of Informatics, University of Edinburgh, 2019.

[22] Bowen Tan, Virapat Kieuvongngam, Yiming Niu : Automatic Text Summarization of COVID-19 Medical Research Articles using BERT and GPT-2, 2020.

[23] Arbi Med Zied: Traitement automatique des langages naturels et le problème d'extraction des métadonnées, 2015.

[24] Corentin Hardy : Contribution au développement de l'apprentissage profond dans les systèmes distribués, 2019.

[25] Chin-Yew Lin : ROUGE: A Package for Automatic Evaluation of Summaries, 2004.

[26] Denkowski & Lavie : Meteor for Multiple Target Languages using DBnary, 2015.

[27] Paul Tardy : Approches neuronales pour le résumé abstractif de transcriptions de parole, 2022.

[28] Louis Martin, Benjamin Muller, Pedro Javier Ortiz Suárez, Yoann Dupont, Laurent Romary, Éric Villemonte de la Clergerie, Djamé Seddah, Benoît Sagot : CamemBERT : a Tasty French Language Model, 2019.

[29] Moussa Kamal Eddine, Antoine J.-P. Tixier, Michalis Vazirgiannis : BARThez: a Skilled Pretrained French Sequence-to-Sequence Model, 2021.

[30] Corentin Blanc, Alexandre Bailly, Elie Francis , Thierry Guillotin , Fadi Jamal ,Bechara Wakim, Pascal Roy: FlauBERT vs. CamemBERT: Understanding patient's answers by a French medical chatbot, 2022.

[31] Thomas Scialom , Paul-Alexis Dray , Sylvain Lamprier , Benjamin Piwowarski, Jacopo Staiano : MLSUM: The Multilingual Summarization Corpus, 2020.