OFFRE VERBATIM

-

Classification de verbatim issus de sondages

GHOUFRAOUI Hamza
Stagiaire data scientist au pôle data chez I-team, La Poste Groupe
Étudiant en master 2 IS, université de Nantes

28 août 2024

Tuteur en entreprise : TORTEAU Philippe
Référent universitaire : STAMM Aymeric

Table des matières

Remerciements

Je souhaite remercier toutes les personnes qui ont contribué au bon déroulement de mon stage.

En premier lieu, je remercie Philippe TORTEAU, mon tuteur en entreprise pour m'avoir accordé sa confiance afin de mener à bien mon projet.

Je tiens aussi à remercier Aymeric STAMM, mon référent universitaire et tous les professeurs du Master d'ingénierie statistique de l'université de Nantes pour cette formation.

Je remercie également Arnaud LE MOING et Perrine LECALLIER pour leur encadrement et leur accompagnement tout au long de ce stage.

Enfin, je remercie chaleureusement toute l'équipe du pôle data avec qui ce fut un réel plaisir de travailler.

Table des sigles et des abréviations

C3S : **C**entre de Solution, Siège et Support
CDEM : Centre de Développement Expertise Mutualisé
VM : Machine Virtuelle
NLP : Natural Language Processing (Traitement automatique du langage naturel)
BERT : Bidirectional Encoder
BART : Modèle pré entraîné pour la génération, la traduction et la compréhension du langage naturel
BSCC : Branche Services Courrier-Colis
LBP : La Banque Postale
BGPN : Branche Grand Public et Numérique
SI : Service Informatique
IA : Intelligence Artificielle
EMRG : Équipe de Mobilité et Recrutement Groupe
TF-IDF : Term Frequency-Inverse Document Frequency
GenIA : Intelligence Artificielle GÉNérative
LLM : Large Language Model
CBOW : Continu Bag Of Word
LSTM : Long Short-Term Memory
RNN : Recurrent Neural Networks
POC : Proof Of Concept
ARI : indice de Rand ajusté
FMI : indice de Fowlkes - Mallow

Introduction

Contexte de l’étude

Dans le cadre de mon stage en tant que Data Scientist au groupe La Poste, j'ai eu l'opportunité de travailler sur un projet axé sur l'analyse de verbatim issus de sondages provenant de différents services. Ce projet s'inscrit dans une démarche globale visant à améliorer la satisfaction client en exploitant les données textuelles collectées par le service en question.
L'analyse de ces réponses fournies par les clients, permet de mieux comprendre leurs attentes, ressentis et besoins.

Le traitement du langage naturel (NLP) est au cœur du projet. Le **NLP** est une branche de l'intelligence artificielle qui vise à permettre aux ordinateurs de comprendre, interpréter et générer le langage humain. En appliquant des techniques avancées telles que la tokenisation, la vectorisation, l’embedding et l'utilisation de modèles de langage pré entraînés (BERT , BART) , nous transformons les textes bruts en données exploitables. Ces techniques permettent par exemple d'analyser le sentiment d’un texte.

L'objectif principal de ce projet étant de classer les verbatim en différentes catégories, on utilise des **méthodes de classification** telles que k-means, les réseaux de neurones LSTM… Ces approches permettent de regrouper les verbatim en fonction de leurs similarités sémantiques, offrant ainsi une vision des principaux thèmes émergents des réponses fournies par les clients.

En conclusion, ce projet illustre l'importance du NLP et des algorithmes de data science dans l'exploitation des données textuelles pour une meilleure compréhension des retours et des besoins clients. Cela représente une étape essentielle pour toute organisation désireuse de rester attentive aux retours de ses clients et de s'ajuster à leurs attentes.

Présentation de l'entreprise

Le groupe La Poste est une société anonyme à capitaux entièrement publics, opérant aussi bien en France qu'à l'international, dans 63 pays répartis sur 5 continents. Ses activités sont variées et vont bien au-delà des services traditionnels de courrier et de colis. La Poste est également un acteur majeur dans les secteurs de la banque et de l'assurance, offrant des solutions financières à une clientèle diverse. De plus, elle s'engage activement dans le développement d'outils numériques, renforçant ainsi sa présence dans le domaine de la transformation digitale et des services en ligne. Cette diversification stratégique permet à La Poste de répondre aux besoins évolutifs de ses clients et de se positionner comme un acteur clé dans plusieurs industries.

Figure 1 : logo La Poste 2020

Cette diversification est due à la stratégie de l'entreprise : en 2010, La Poste a démarré son plan stratégique "**La Poste 2020 : Conquérir l'avenir**" qui consiste à la diversification de ses activités pour pallier la baisse de celle du courrier.

Figure 2 : logo des différentes filiales de La Poste

Ainsi La Poste s'est développée dans le secteur de la livraison de colis avec ses filiales colissimo et dpd, elle est aussi arrivée sur le marché de la banque et de l'assurance avec ses filiales La Banque Postale , Ma French Bank, et enfin elle s'est engagée dans la télécommunication avec la filiale La Poste Mobile.

Suite à cette diversification de ses activités due à la stratégie adoptée en 2010, La Poste tourne une nouvelle page de son histoire en 2020 avec la stratégie **La Poste 2030 : Engagé pour vous** , un plan visant à poursuivre sa transformation pour atteindre un modèle économique durable et autoporteur. Ce plan stratégique vise à répondre aux transitions démographiques, territoriales, écologiques et numériques. L'objectif est de devenir la première plateforme européenne de lien et d'échanges, alliant humanité et digital, tout en étant verte et citoyenne.

La Poste s'engage à maintenir un modèle d'entreprise résilient et multi-activités, en se concentrant sur ses quatre branches principales : Services-Courrier-Colis, GeoPost, Grand Public et Numérique, ainsi que La Banque Postale. Chacune de ces branches a des objectifs stratégiques spécifiques pour 2030, visant à renforcer leur impact et à contribuer au développement global du groupe.

Après avoir examiné les stratégies passées et futures du groupe La Poste, nous allons maintenant détailler les différentes branches qui composent le groupe.
Le groupe La Poste est structuré en **4 branches principales** chacune spécialisée dans un ou plusieurs domaines.

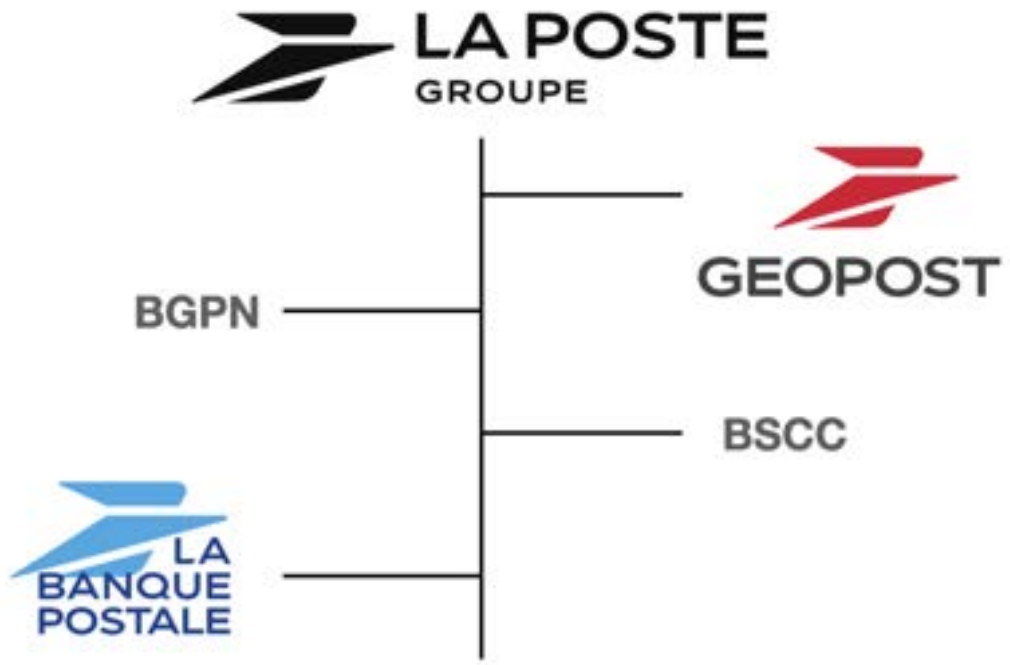


Figure 3 : Les 4 branches du groupe La Poste

- **BSCC : Branche services courrier-colis**

La branche Services-Courrier-Colis de La Poste est le pilier historique et fondamental du groupe. Elle est responsable de la distribution quotidienne de courrier et de colis, desservant des millions de foyers et d'entreprises en France et à l'international. Face à l'essor du commerce en ligne, elle développe des services de livraison rapides et durables, tout en réduisant son empreinte écologique grâce à des initiatives telles que la logistique urbaine verte.

- **LBP : La Banque Postale**

La Banque Postale, la branche bancaire et financière du groupe La Poste, est une banque citoyenne accessible à tous. Elle offre une gamme complète de services bancaires, d'assurance et de gestion d'actifs, en mettant l'accent sur la finance durable et responsable. La Banque Postale soutient des projets qui contribuent à la transition écologique et sociale, et propose des produits financiers adaptés aux besoins des particuliers, des entreprises et des collectivités.

- **BGPN : Branche grand public et numérique**

La branche Grand Public et Numérique de La Poste est dédiée à la fourniture de services numériques et à l'accompagnement des particuliers dans leurs démarches quotidiennes. Elle propose des solutions de confiance numérique, telles que la signature électronique, l'identité numérique et les coffres-forts numériques, facilitant ainsi les interactions administratives sécurisées. En favorisant l'inclusion numérique, cette branche développe des produits et services accessibles à tous, renforçant le lien entre le digital et les services de proximité traditionnels de La Poste.

- **Geopost (anciennement DPDgroup)**

GeoPost, opérant sous la marque DPDgroup, est la branche internationale de livraison express du groupe La Poste. GeoPost est l'un des principaux acteurs mondiaux du marché de la livraison de colis, offrant des services dans plus de 230 pays. Cette branche se distingue par sa capacité à innover, en utilisant des technologies avancées pour garantir des livraisons efficaces et une excellente traçabilité des envois.

Pour finir, faisons un tour des chiffres impressionnants affichés par le groupe :

- En 2022, La Poste a généré un chiffre d'affaires de 35,4 milliards d'euros, elle se classe dans le top 50 des plus grandes entreprises françaises. 41% de son chiffre d'affaires est réalisé à l'international ce qui prouve la force du groupe à étendre ses activités hors de son territoire.

- Le groupe emploie environ 250 000 collaborateurs à travers le monde, dont la majorité est en France. Cet effectif important reflète la diversité des métiers.

- Le groupe est le premier employeur de France avec 10 609 recrutements en CDI en 2021.

- Chaque jour, La Poste traite et distribue environ 18 millions de courriers et plus de 1 million de colis. Ces chiffres montrent l'ampleur des opérations logistiques.

- Il s'agit du plus grand réseau commercial local en France, accueillant 1,3 million de clients chaque jour dans ses points de vente.

Présentation du pôle data

Mon stage a été effectué au pôle data du cdem au C3S de l'entité i-team. Tout d'abord voyons une brève présentation d'i-team.

I-Team est un service innovant au sein du groupe La Poste. En tant que service transversal, i-team soutient les besoins informatiques de l'ensemble de l'organisation. En effet elle mutualise les services SI du groupe tel que les services SI RH , SI Finances ... assurant le bon fonctionnement et l'intégration de diverses solutions technologiques. Avec une équipe robuste d'environ 1 000 professionnels IT, I-team gère les systèmes et applications indispensables aux opérations de La Poste.

Grâce à ces efforts, I-Team joue un rôle vital dans la transformation numérique de La Poste, en alignement avec les objectifs stratégiques du groupe pour rester compétitif et réactif dans un marché en constante évolution

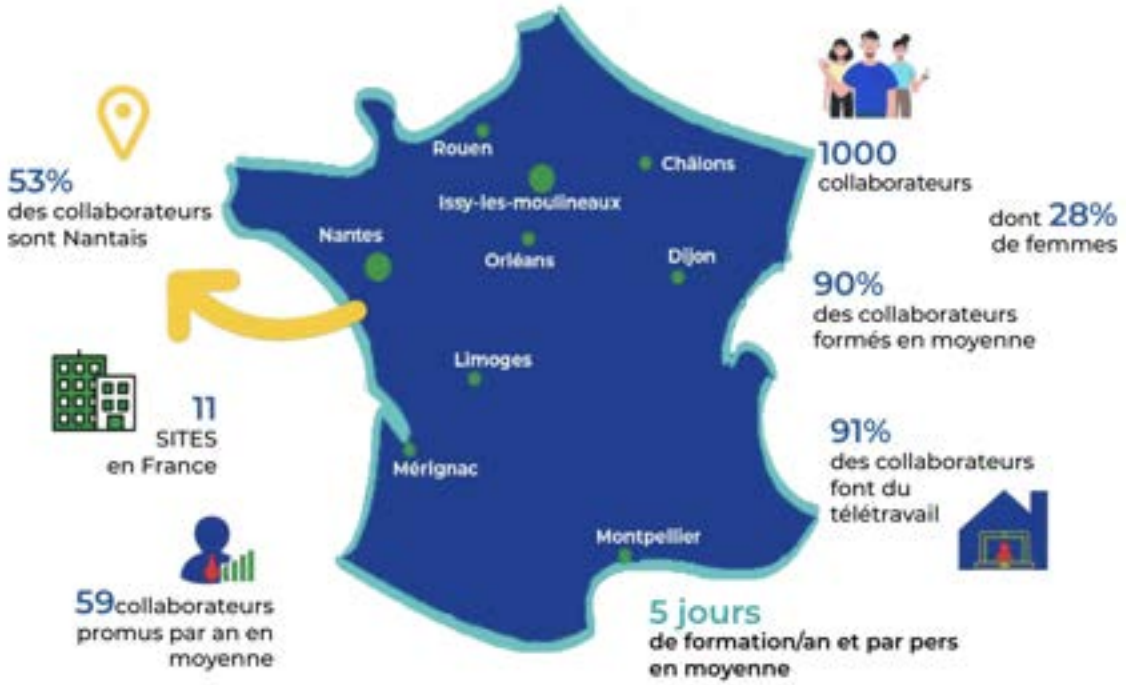


Figure 4 : Infographie I-Team

Personnellement, je me situe plus précisément dans la direction C3S , plus particulièrement dans le département CDEM, et encore plus particulièrement dans le **pôle DATA**, qui s'occupe de projets de traitement et analyse de données, et aussi des projets IA.

En avril 2024 dans le cadre de mon master ingénierie statistique j'ai rejoint le pôle DATA sous l'encadrement de Mr TORTEAU Philippe en tant que data scientist stagiaire , le pôle data est une équipe de 7 personnes s'occupant de projet d'analyse de data, d'anonymisation des données, d'intelligence artificielle et du data lake. L'équipe est composée de data scientist/analyst, de data engineer , d'un product owner, le tout aidé par un développeur et sous la gouvernance d'un manager.

Le pôle DATA gère également un data catalog : c'est une plateforme centralisée qui répertorie et organise les ensembles de données disponibles au sein de l'entreprise. Il offre une visibilité complète sur les données, y compris leur origine, leur structure, leur usage autorisé et les politiques de confidentialité associées. Cela facilite non seulement la découverte et l'accès aux données pertinentes pour les projets, mais aussi la gouvernance des données à travers l'organisation.

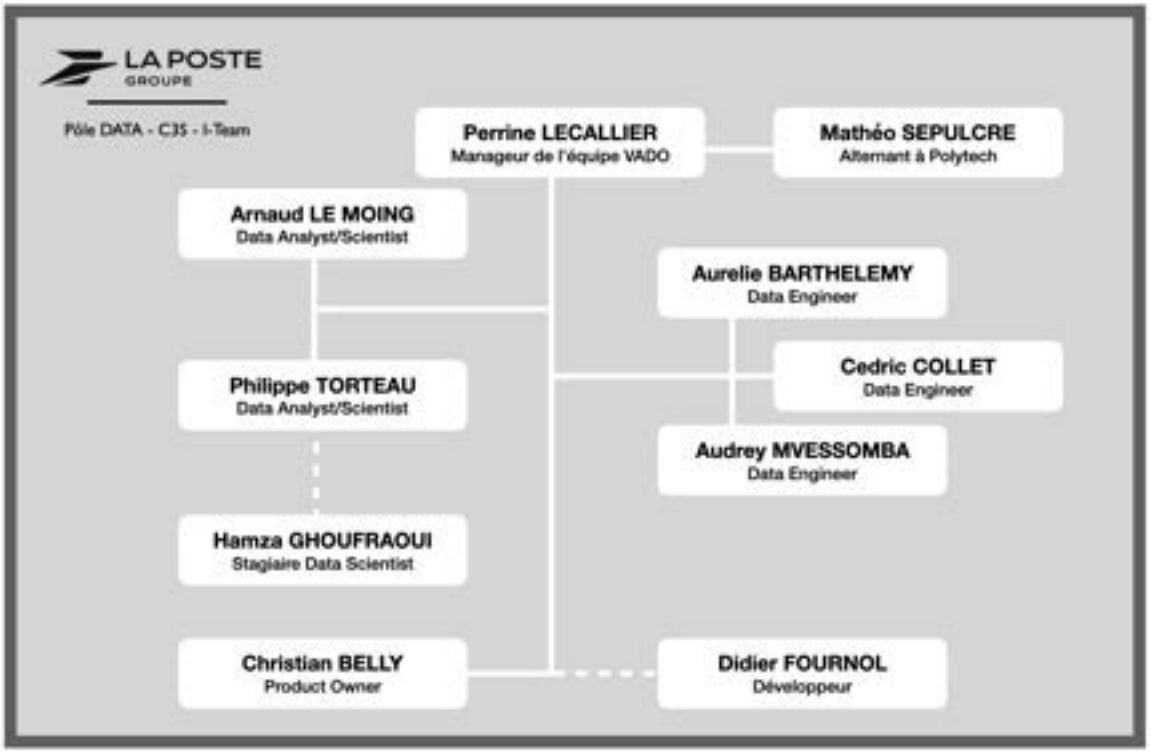


Figure 5 : Organigramme du pôle DATA

La Poste a amorcé son virage vers l'intelligence artificielle en 2016 avec l'acquisition de Probayes. Fondée par des chercheurs de l'Inria et du CNRS, Probayes est spécialisée dans les solutions d'IA sur mesure, notamment pour la détection des fraudes à la carte bancaire. En 2021, La Poste a renforcé son engagement en rachetant Open Value, un cabinet de conseil en big data et IA. Grâce à ces acquisitions stratégiques, La Poste se positionne parmi les cinq premières entreprises françaises en matière de **transformation data et IA,** tant pour ses clients que pour ses propres services.

C'est dans le cadre de la transition vers l'intelligence artificielle que le pôle a été chargé de développer l'offre verbatim.

Origine du projet

Au printemps 2022, le **service EMRG** (Équipe de Mobilité et Recrutement Groupe) de La Poste, qui est responsable des candidatures internes, a lancé un appel d'offres pour analyser et classifier les verbatim issus de sondages. Ce service est spécialisé dans l'accompagnement des employés dans leur évolution de carrière, offrant un suivi personnalisé, des conseils et des épreuves pour les préparer aux entretiens. Cependant, l'analyse manuelle des verbatim s'est révélée extrêmement

chronophage. Pour gagner en efficacité, le service a décidé d'automatiser ce processus en utilisant des outils d'intelligence artificielle et de classification pour trier et évaluer ces retours, permettant ainsi une évaluation plus rapide et objective du service.

En réponse à l'appel d'offres lancé par le service EMRG au printemps 2022, le pôle Data a été chargé de développer une solution pour analyser et classifier les verbatim issus des sondages internes.

Le pôle Data a dû relever plusieurs défis, en particulier sur la question de savoir s'il fallait adopter une approche supervisée ou non supervisée. Deux solutions ont finalement émergé. La première est une **solution exploratoire** utilisant une méthode non supervisée, permettant d'identifier les thèmes récurrents dans les verbatim sans connaître ces thèmes à l'avance. Cette approche est particulièrement utile pour découvrir de nouvelles tendances ou sujets inattendus dans les réponses des sondés.

La seconde solution vise le **suivi des thèmes** identifiés sur plusieurs mois, reposant cette fois sur une méthode supervisée. Dans ce cas, les thèmes sont déjà connus et l'objectif est de suivre leur évolution dans le temps. Cependant, cette méthode présente des limitations. Elle est énergivore, nécessitant une base d'apprentissage conséquente qui est rarement disponible. De plus, elle manque de flexibilité : si les questions des sondages changent ou si certains thèmes disparaissent, la méthode et la base d'apprentissage doivent être entièrement retravaillées, ce qui compromet le fonctionnement du modèle.

Traitement des verbatim

Afin de lancer l'offre verbatim, les membres du pôle data (notamment Arnaud LE MOING et Philippe TORTEAU) se sont penchés sur l'élaboration d'une solution exploratoire qui repose sur une classification non supervisée. Arnaud et Philippe ont chacun développé une solution de leur côté. Avant de détailler les solutions développées, il est nécessaire de de détailler et de comprendre comment il est possible de classifier des textes.

Pour pouvoir classifier efficacement les textes, il est essentiel de passer par les étapes de tokenization et de vectorisation. La **tokenization** consiste à découper le texte en unités de base, appelées tokens, qui peuvent être des mots, des phrases, ou même des caractères. Ensuite, la **vectorisation** transforme ces tokens en vecteurs numériques. Ces vecteurs représentent les caractéristiques du texte de

manière que les algorithmes de classification (et de machine learning) puissent les comprendre et les traiter.

Dans les solutions développées par mes collègues, deux approches de vectorisation ont été utilisées. La première méthode, basée sur le **TF-IDF** (Term Frequency-Inverse Document Frequency), calcule l'importance de chaque mot dans un document par rapport à un corpus de documents. La seconde méthode utilise des **embeddings**, qui sont des représentations vectorielles plus sophistiquées des mots, capturant leurs significations contextuelles. Ces deux méthodes de vectorisation seront un peu plus expliquées en détail ultérieurement.

Avant de passer par ses deux étapes de tokenization et vectorization, il est nécessaire de **nettoyer** le texte. Nettoyer un texte c'est éliminer toutes les ponctuations, passer tout le texte en minuscule et enlever tous les accents.

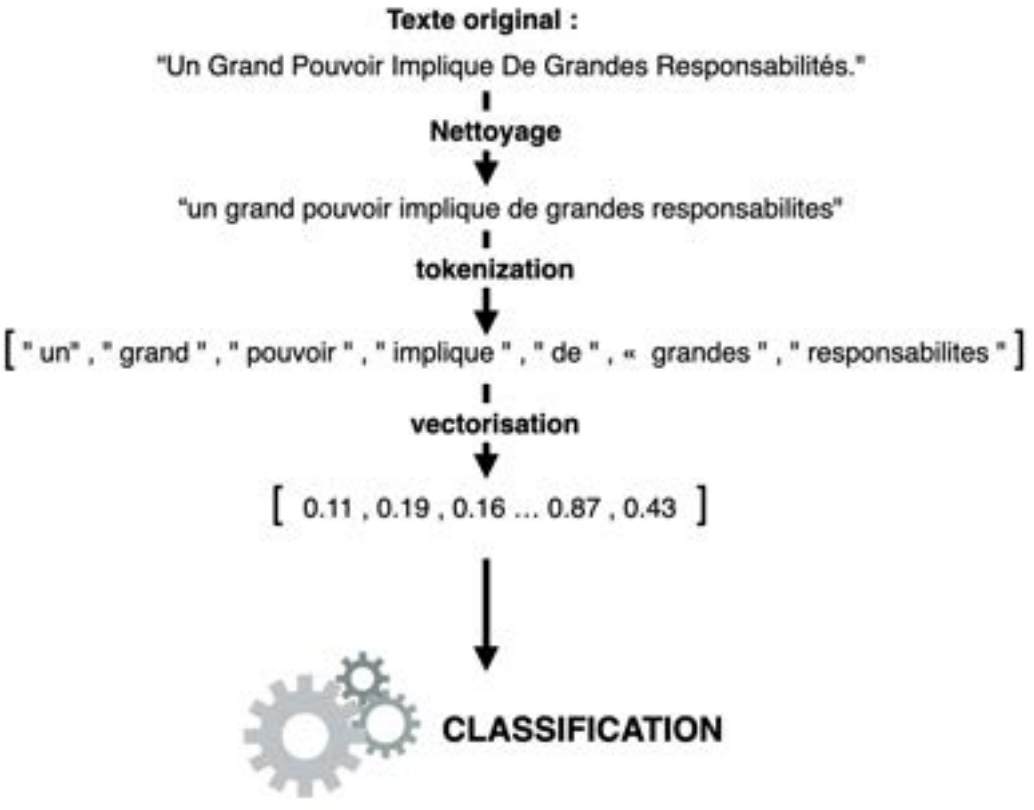


Figure 6 : Traitement appliqué aux textes en amont de la classification

Une fois le traitement du texte réalisé , une nouvelle problématique arriva sur la table. Une courte lecture de chaque laïus interpella mes collègues. En effet, il arrive souvent que les gens répondant au sondage intègrent souvent plusieurs

idées dans une seule réponse. Ainsi, un même verbatim peut contenir plusieurs thèmes distincts, ce qui complique considérablement la classification. Initialement, l'idée de découper les textes en fonction de la ponctuation a été utilisée, mais elle a rapidement été abandonnée. En effet, les répondants aux sondages ne prennent pas toujours le temps de ponctuer correctement leurs réponses, considérant cette tâche comme fastidieuse et ennuyeuse. Leur priorité est souvent de répondre rapidement pour passer à autre chose. C'est dans ce contexte que l'utilisation de l'intelligence artificielle générative s'est imposée comme une solution pour gérer cette problématique.

Intelligence artificielle générative

Pour découper les verbatim, nous utilisons des outils d'intelligence artificielle générative, plus précisément des modèles de langage **LLM** (Large Language Models). Avant de voir leur utilisation dans notre contexte, définissons ce qu'est un LLM.

Un LLM est un modèle d'IA entraîné sur une énorme quantité de données textuelles, il est capable de comprendre et de générer du langage naturel. Ces modèles reposent la plupart du temps sur du **deep learning** , et afin de permettre ce type d'apprentissage en profondeur, ils sont construits sur des réseaux de neurones : ce type spécifique de réseaux neuronaux est appelé **modèle transformers**. Ces modèles sont capables d'apprendre le contexte, ce qui est particulièrement important pour le langage humain. Les modèles transformers utilisent une technique mathématique appelée auto-attention pour détecter la manière dont les éléments d'une séquence sont reliés entre eux. Ils sont donc plus robustes pour comprendre le contexte que d'autres types d'apprentissage automatique. Ils comprennent, par exemple, comment la fin d'une phrase est reliée au début, et comment les phrases d'un paragraphe sont reliées entre elles.
Plus de détails mathématiques et techniques sur les modèles transformers sont disponibles en annexe (voir *An.1 : Les modèles transformers*).

Maintenant que nous avons vu ce qu'était un LLM et aussi ce qu'il y'a sous le capot de ces modèles, comment utiliser un LLM sur notre base de données en passant par python ?

Figure 7 : Logo Ollama

Afin d'utiliser des LLM sur nos machines nous sommes passés par une plateforme qui propose des LLM pouvant être utilisés localement. Cette plateforme se nomme **Ollama**.

L'avantage d'Ollama c'est qu'il exécute les modèles directement sur la machine tout en gardant les données en local, ce qui améliore fortement la confidentialité et la rapidité des traitements. La confidentialité est capitale pour nous, car les clients pour qui nous travaillons ne veulent sûrement pas que leurs données se retrouvent sur Internet, d'où la nécessité d'utiliser un service comme Ollama qui garde toutes les données en local.

L'utilisation d'Ollama en Python est très intuitive. Il suffit de l'installer avec une commande pip, puis de charger le modèle que l'on veut utiliser. Ensuite, il suffit de lui envoyer des prompts (requêtes) pour l'utiliser dans diverses tâches.

Pour découper nos verbatims, nous utilisons un gros modèle de langage disponible sur ollama compatible avec notre infrastructure. Voici un exemple de découpage réalisé par le modèle 3.1:70B :
Le laïus " l'accompagnement est très bien mais j'aurais aimé plus de préparations aux entretiens" a été découpé en : "très bon accompagnement " , "meilleure préparation aux épreuves" .

Solution exploratoire

Nous savons désormais comment traiter nos verbatims, que ce soit en les rendant exploitables par des algorithmes de classification ou en séparant les idées lorsqu'un verbatim en contient plusieurs. Tout logiquement, la suite est l'élaboration de la solution en faisant un choix sur l'algorithme de classification à utiliser, mais aussi sur le choix de la méthode de vectorisation. Les deux solutions proposées par Philippe TORTEAU et Arnaud LE MOING sont détaillées ci-dessous :

La solution développée par Philippe qu'on appellera solution P repose sur une classification Kmeans ++ et sur de l'embedding en tant que technique de vectorisation.

La solution développée par Arnaud (solution A) utilise une classification ascendante hiérarchique (CAH) et la vectorisation TF-IDF

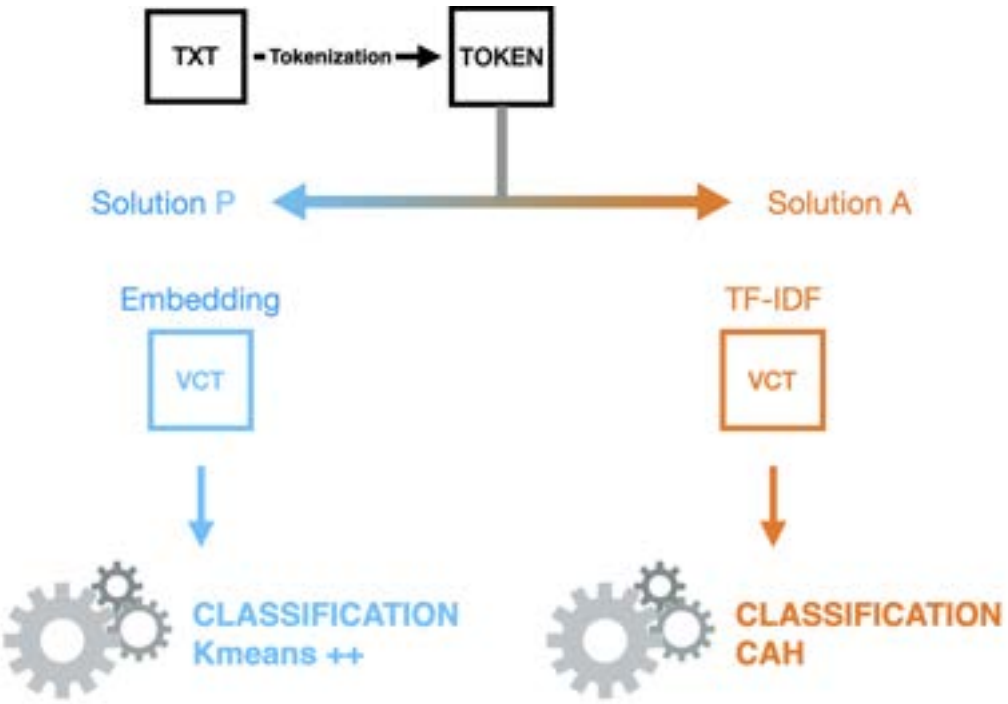


Figure 8 : Les deux solutions exploratoires développées

Pour commencer, voyons plus en profondeur les deux méthodes de vectorisation utilisées par les deux solutions.

En 1954, Zellig Harris, un linguiste, a publié "Distributional Structure". Dans cette publication, il propose l'hypothèse suivante : « les différences de sens sont corrélées aux différences de distribution », reformulée plus tard par John R. Firth sous la forme suivante : « **On ne connaît un mot que par ses fréquentations** ». Cette idée est connue sous le nom d'hypothèse distributionnelle. En résumé, cette hypothèse stipule que les mots apparaissant dans un même contexte linguistique ont un sens proche. Le contexte linguistique fait référence aux mots situés avant et après le mot observé.

Par exemple :

> Dans la phrase « J'ai raté le train à cause des embouteillages », le mot "train" est observé dans le contexte linguistique « J'ai raté le ... à cause des embouteillages ».

De plus, on peut remplacer le mot observé par un autre mot. Si le sens de la nouvelle phrase reste sémantiquement proche de l'originale, cela signifie que le nouveau mot a un sens proche de celui du mot d'origine.

Par exemple :

> Si l'on remplace "train" par "chat", la phrase « J'ai raté le chat à cause des embouteillages » n'a aucun sens ; cela montre que les mots "chat" et "train" ne sont pas proches sémantiquement. En revanche, si l'on remplace "train" par "bus", la phrase « J'ai raté le bus à cause des embouteillages » garde un sens similaire à la phrase d'origine, indiquant que les mots "bus" et "train" ont un sens proche.

C'est dans les années 80 avec l'essor de l'ordinateur et les prémices de l'intelligence artificielle que Gerard SELTON dans sa publication *Introduction to modern information retrieval* introduit le concept de modèle vectoriel, l'idée étant de représenter un texte sous forme de vecteur numérique.
La première idée pour réaliser cela fut de compter les mots, voici un exemple :

Prenons les 3 phrases suivantes :

- Phrase 1 : " Le bus est en retard "
- Phrase 2 : " Le train est rapide "
- Phrase 3 : " Le bus est rapide et plein "

Pour vectoriser ces phrases en comptant les mots, nous faisons comme suit :

	bus	en	est	et	le	plein	retard	rapide	train
Phrase 1	1	1	1	0	1	0	1	0	0
Phrase 2	0	0	1	0	1	0	0	1	1
Phrase 3	1	0	1	1	1	1	0	1	0

Cette méthode de vectorisation fournit des vecteurs de la taille du vocabulaire présent dans l'ensemble des phrases, donc ici des vecteurs de taille 9, ainsi on obtient ces vecteurs pour les 3 phrases :

count_vect_phrase1 = [1,1,1,0,1,0,1,0,0]
count_vect_phrase2 = [0,0,1,0,1,0,0,1,1]
count_vect_phrase3 = [1,0,1,1,1,1,0,1,0]

Le souci avec cette méthode c'est que les mots ont tous le même poids , par exemple dans la phrase 2 le déterminant "Le" et le mot "rapide" ont le même poids dans le vecteur alors que le mot rapide a beaucoup plus de significativité sémantique. Face à cela SELTON propose une nouvelle méthode, le TF-IDF.

TF-IDF (term frequency - inverse document frequency) ,

- TF (Term Frequency) : C'est la fréquence d'un mot dans un document spécifique. Par exemple, dans la phrase 1 , le mot "bus" représente 20 % de tous les mots.
- DF (Document Frequency) : Cela fait référence à la fréquence des documents qui contiennent le mot en question. Par exemple, le mot "bus" apparaît dans deux documents sur trois, le DF de "bus" est de 66 %.

Le TF-IDF effectue un arbitrage entre ces deux mesures. Un mot qui apparaît dans très peu de documents mais qui est très présent dans un document spécifique se verra attribuer un poids plus élevé. En revanche, un mot qui apparaît dans tous les documents aura un poids plus faible, car il est considéré comme moins discriminant.

Mathématiquement ca donne ca :

Pour calculer la fréquence du terme il existe plusieur méthodes, on peut utiliser La fréquence « brute » d'un terme (comme utilisé dans les exemples) , elle se calcule très facilement tel que suit : le nombre d'occurrences de ce terme dans le document considéré, divisé par le nombre de mots du document, voici sa formule:

$$\mathrm{TF}(t,d) = \frac{f_{t,d}}{n_d}$$

où :

• $f_{t,d}$ représente le nombre d'occurrences du terme t dans le document d,
• n_d représente le nombre total de termes dans le document d.

Un autre schéma de pondération très utilisé pour calculer le TF est la normalisation par le max, elle permet de prendre en compte la longueur du document, et on définit alors le TF comme suit :

$$\mathrm{TF}_{\mathrm{norm}}(t,d) = K + (1-K)\frac{\mathrm{TF}(t,d)}{\max_{t' \in d} \mathrm{TF}(t',d)}$$

où :

• $\mathrm{TF}(t,d)$ est la fréquence brute du terme t dans le document d,
• $\max_{t' \in d} \mathrm{TF}(t',d)$ est la fréquence maximale de n'importe quel terme t' dans le document d.
• K est une constante fixée ; la plupart du temps, on prendra k = 0.5

Et pour calculer l'IDF , la Fréquence inverse de document :

$$\mathrm{IDF}_i = \log\left(\frac{|D|}{|\{d_j : t_i \in d_j\}|}\right)$$

où :

- $|D|$: nombre total de documents dans le corpus ;
- $|\{d_j : t_i \in d_j\}|$: nombre de documents où le terme t_i apparaît (c'est-à-dire lorsque $n_{i,j} \neq 0$).

Finalement la norme TF-IDF s'obtient en multipliant les deux mesures TF et IDF.
La norme TF-IDF d'un mot i dans un document j est donné par :

$$\mathrm{tfidf}_{i,j} = \mathrm{tf}_{i,j} \cdot \mathrm{idf}_i$$

En reprenant l'exemple vue plus haut on a cette fois ci les résultats suivants :

	bus	en	est	et	le	plein	rapide	retard	train
Phrase 1	0.42	0.55	0.32	0	0.32	0	0	0.55	0
Phrase 2	0	0	0.39	0	0.39	0	0.55	0	0.66
Phrase 3	0.38	0	0.3	0.5	0.3	0.5	0.38	0	0

tfidf_vect_phrase1 = [0.42, 0.55, 0.33, 0. , 0.33, 0. , 0. , 0.55, 0.]
tfidf_vect_phrase2 = [0. , 0. , 0.39, 0. , 0.39, 0. , 0.5 , 0. , 0.66]
tfidf_vect_phrase3 = [0.39, 0. , 0.3 , 0.51, 0.3 , 0.51, 0.39, 0. , 0.]

Cette fois ci contrairement à la méthode de comptage on capture plus les caractéristiques sémantiques des phrases, par exemple dans la phrase 2 les mots qui donne le plus de sens à cette phrase sont "train" et "rapide" , de ce fait le poids associé à ces mots dans le vecteurs (0.55 et 0.66) sont supérieur au poids associé au déterminant "le" (0.39).

Dans l'exemple ci-dessus, c'est la fonction **TfidfVectorize**r de la librairie Sklearn qui a été utilisée.

En conclusion, la méthode TF-IDF permet de donner plus de poids aux termes spécifiques et de réduire l'importance des mots courants. Cependant, le TF-IDF présente certaines **limites**. Il ne capture pas le contexte sémantique des mots, se contentant de les traiter comme des items indépendants. De plus, il est sensible aux variations dans les formulations et n'est pas capable de saisir les synonymes

ou les relations entre les mots. Ces limitations peuvent réduire l'efficacité de la classification, notamment dans un contexte où le sens des mots est crucial.

Pour surmonter ces obstacles, il existe des méthodes capturant les relations sémantiques entre les mots en les représentant dans un espace vectoriel, où des termes similaires sont proches les uns des autres, permettant ainsi une meilleure compréhension du contexte linguistique et améliorant l'efficacité de la classification du langage naturel. Ces méthodes sont les techniques **d'embedding.**

L'Embedding est une technique avancée de vectorisation. Contrairement aux autres techniques de vectorisation, elle capture les relations sémantiques entre les termes. Cela signifie que les termes ayant des significations similaires ou possédant des contextes linguistiques similaires seront représentés par des vecteurs proches dans l'espace. Comment cela est-il possible ?

En utilisant un réseau de neurones, où l'objectif est de minimiser une fonction perte qui force les vecteurs de mots contextuellement similaires à se rapprocher.
Il existe deux façons de réaliser cela :

- L'entraînement par le procédé CBOW (Continu Bag Of Word) :

Ici le but est de prédire un mot en fonction de son contexte, pour ce faire voici un schéma détaillant la procédure :

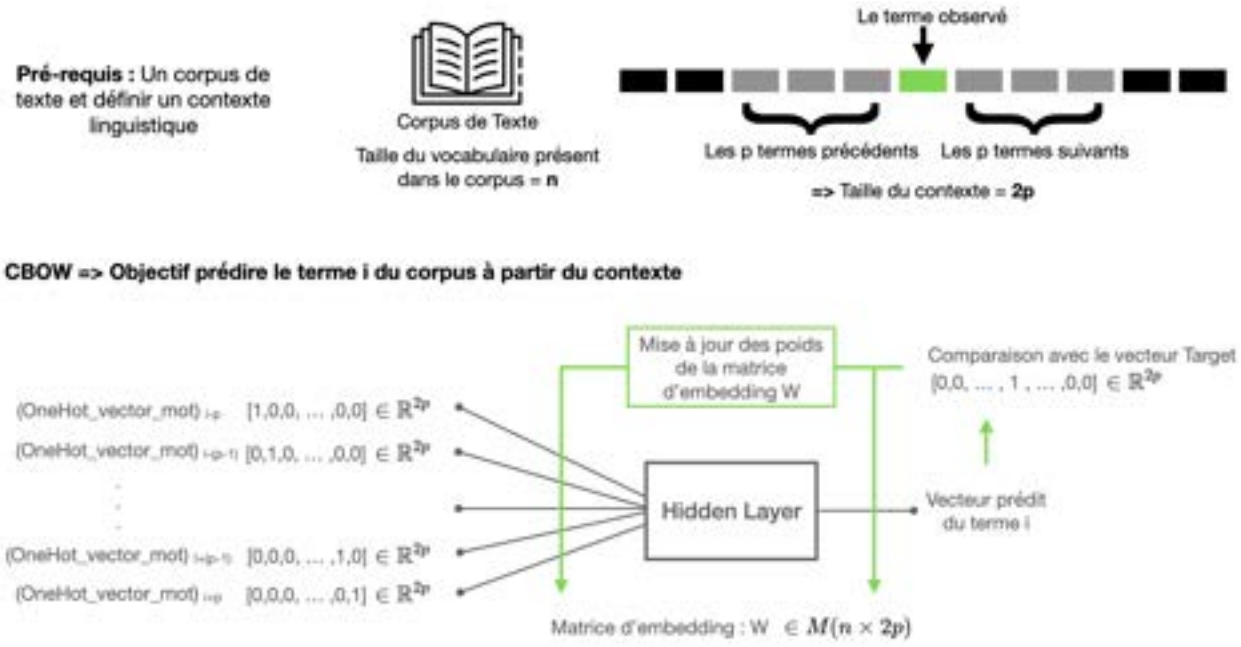


Figure 9 : Procédure CBOW

Le schéma montre la méthode CBOW, une méthode fréquemment employée pour former des modèles d'embedding de mots.

La procédure est telle que suit :

- Préparation des données : Tout d'abord, il est nécessaire d'avoir un corpus de texte et de définir un contexte linguistique, c'est-à-dire un ensemble de mots qui entourent le mot cible à prédire. En général, la taille de ce contexte est définie par un paramètre p, qui correspond au nombre de mots précédents et suivants pris en compte.

- Représentation des mots : Chaque terme du contexte est vectorisé en un vecteur one-hot, qui est une représentation binaire dans un espace de dimension n, où n correspond à la taille du vocabulaire. Dans cette représentation , le vecteur comprend un 1 à la position correspondant au mot en question et des 0 ailleurs.

- Matrice d'embedding : Les vecteurs one-hot sont ensuite multipliés par une matrice d'embedding W pour les transformer en vecteurs denses. Cette matrice W est initialisée aléatoirement et est actualisée au cours de l'entraînement du modèle.

- Pour prédire le mot cible, on combine les vecteurs denses obtenus (souvent moyennés) et les passe à travers une couche cachée (Hidden Layer) afin de prédire le vecteur cible, qui correspond à la représentation one-hot du mot que l'on souhaite prédire.

- Mise à jour des poids: Une fois la prédiction faite, elle est comparée au vecteur target (la vraie représentation one-hot du mot cible). L'erreur de prédiction est utilisée pour ajuster les poids de la matrice d'embedding, améliorant ainsi la capacité du modèle à prédire les mots cibles à partir de leur contexte dans les itérations suivantes.

Une procédure symétrique à la CBOW visant à prédire les mots du contexte en fonction d'un mot en entrée est la méthode dite : **Skip - Gram**. Le fonctionnement des deux méthodes est très similaire, voici un schéma qui explique la méthode :

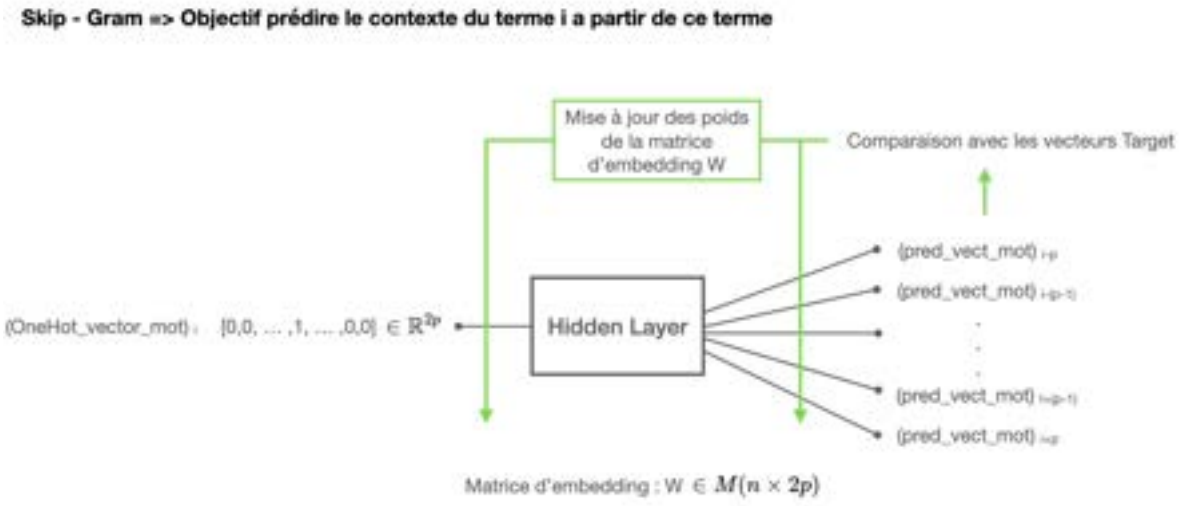


Figure 10 : Procédure Skip Gram

Après avoir vu comment fonctionnent les embeddings et la manière dont les modèles sont entraînés, nous allons voir comment les utiliser dans le cas de la vectorisation d'une phrase.

Supposons que nous ayons un modèle d'embedding pré-entraîné et que nous voulons obtenir le vecteur pour le mot "bus". Il suffit tout simplement de consulter la matrice d'embedding pour extraire le vecteur de 2p dimensions associé à "bus". Si l'on souhaite obtenir l'embedding pour une phrase comme "Le bus roule", on a juste à faire la moyenne des vecteurs des mots "Le", "bus", et "roule" pour obtenir la vectorisation de la phrase.

Dans la pratique pour la solution P , Philippe a fait le choix d'utiliser **USE** (universal sentence encoder) une solution d'embedding développée par tensorflow , elle produit des vecteurs de grande dimension (512).

Après avoir exploré les différentes techniques de vectorisation, nous disposons maintenant d'une représentation numérique des textes, essentielle pour permettre à nos algorithmes de classification de fonctionner efficacement. Dans la section suivante, nous allons aborder les méthodes de classification que nous employons pour identifier et catégoriser les thèmes présents dans les verbatim, en utilisant les vecteurs obtenus lors de la phase de vectorisation.

Avant de continuer il est nécessaire de comprendre comment marche le concept de "vecteurs proches". En effet, la vectorisation du langage naturel fournit des vecteurs de grande dimension, (comme USE qui fournit des vecteurs de taille 512) et calculer des distances dans un espace vectoriel de dimension élevée est une tâche ardue.
Pour cela, la **distance cosinus** est souvent privilégiée. Cette mesure évalue l'angle entre deux vecteurs, ce qui la rend particulièrement efficace pour juger de la similitude entre vecteurs dans de tels espaces contrairement aux autres distances, comme la distance euclidienne, qui peuvent devenir obsolètes dans des espaces de grande dimension.

Mathématiquement nous avons :

Soient A et B deux vecteurs de dimensions n.

- On a la Similarité cosinus :

$$\text{similarité cosinus}(\mathbf{A}, \mathbf{B}) = \cos(\theta) = \frac{\mathbf{A} \cdot \mathbf{B}}{\|\mathbf{A}\|\|\mathbf{B}\|}$$

Avec :

$$\mathbf{A} \cdot \mathbf{B} = \sum_{i=1}^{n} a_i b_i$$

$$\|\mathbf{A}\| = \sqrt{\sum_{i=1}^{n} a_i^2}$$

$$\|\mathbf{B}\| = \sqrt{\sum_{i=1}^{n} b_i^2}$$

- Donc Distance cosinus

$$\text{distance cosinus}(\mathbf{A}, \mathbf{B}) = 1 - \cos(\theta)$$

Grâce à la distance cosinus, il est désormais possible d'identifier les vecteurs proches les uns des autres, ce qui est crucial pour assurer le bon fonctionnement de la classification. C'est cette distance que nous choisirons pour tous nos algorithmes de classification non supervisée.

Après avoir appris à découper nos verbatim en idées, à les rendre exploitables grâce à la vectorisation, et à calculer une distance qui nous permet d'identifier les

vecteurs proches les uns des autres, il est maintenant possible de définir la classification de ces textes vectorisés.

Pour la solution P, une fois nos verbatim découpés puis vectorisés par embedding , nous utilisons une classification **K-means ++**.

K-means++ est une version améliorée de K-means, conçue pour surmonter une des principales limites de K-means : la sensibilité à l'initialisation des centroïdes. L'objectif de cet algorithme est de choisir les centroïdes initiaux de manière à minimiser les risques de convergence vers un minimum local sous-optimal, ce qui conduit à une meilleure performance globale.

Les étapes de cet algorithme sont très similaire à un K-means habituel :

- Initialisation des Centroïdes :

Alors que l'algorithme K-means initialise les centres des clusters de manière aléatoire, K-means ++ utilise quant à lui une stratégie d'initialisation plus intelligente. Le premier centroïde est choisi au hasard parmi les points de données. Ensuite, les centroïdes suivants sont choisis avec une probabilité proportionnelle au carré de la distance du point de données le plus proche d'un centroïde déjà sélectionné. Cette méthode assure que les centroïdes initiaux sont bien répartis dans l'espace des données, ce qui tend à conduire à une convergence plus rapide et à des clusters de qualité.

Les autres étapes d'assignation, de mise à jour des centroïdes et d'itérations restent les mêmes que pour l'algorithme K-means. C'est pour la phase d'initialisation des centroïdes, qui nous garantit une convergence plus rapide et des clusters plus fiables, que la solution P utilise cet algorithme.

Nous avons maintenant passé en revue la plupart des outils utilisés par la solution P (à l'exception de l'analyse de sentiments et du wordcloud, qui seront expliqués par la suite). Examinons maintenant le fonctionnement de cette solution.

Dans le but d'avoir une classification robuste et performante, Philippe a mis en place un algorithme qui exécute **p itérations**, avec p fixé par l'utilisateur, une classification K-means++. Ensuite, grâce à l'initialisation non aléatoire des centroïdes que réalise le K-means++ on s'assure d'une concordance entre les clusters formés au cours des différentes itérations, ainsi la consolidation est possible, on peut donc déterminer qu'un individu appartient à un cluster donné si, sur les p exécutions, c'est ce cluster qui lui a été le plus souvent attribué.

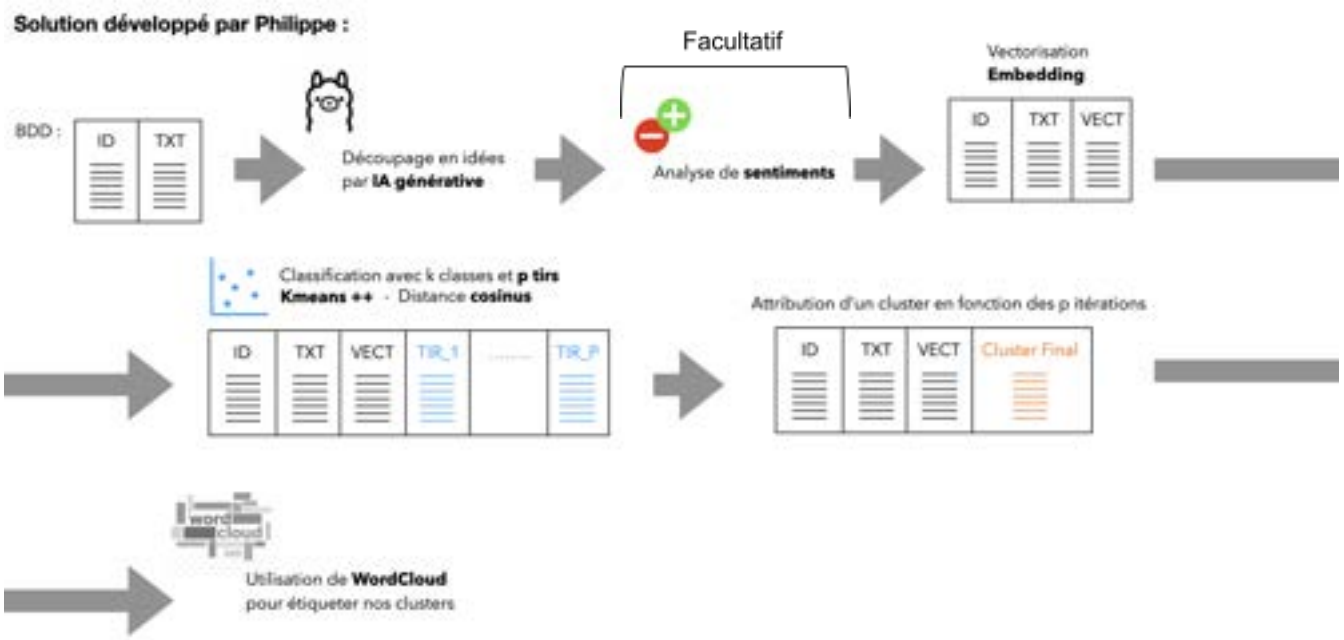


Figure 11 : Schéma détaillé de la solution P

Sur la figure ci-dessus, en plus des étapes déjà vues et expliquées , nous ajoutons une analyse de sentiments. Elle peut être requise (sans être obligatoire) à cette étape afin de séparer les verbatim positifs des négatifs. La classification sera ensuite faite d'une part sur les positifs puis de l'autre sur les négatifs. S'ajoute également une étape avec l'utilisation de word cloud pour étiqueter les clusters formés par la classification.

Le but de l'**analyse des sentiments** est de déterminer le sentiment de chaque idée d'un laïus découpé en amont par genIA. En effet, l'utilisation de l'analyse de sentiments pour classifier des verbatims issus de sondages est particulièrement pertinente, car elle permet de distinguer efficacement les textes positifs des textes négatifs. Cette distinction a pour but d'améliorer la qualité de la classification, car les verbatim portant des sentiments opposés expriment souvent des idées et des opinions très différentes tout en contenant des mots souvent très proches. En séparant les textes selon leur polarité émotionnelle, on obtient des groupes de données plus homogènes ce qui facilite l'identification de thèmes. Cela conduit à une classification plus précise.

Pour réaliser cette analyse de sentiments, nous utilisons un modèle pré-entraîné appelé **BARThez**. Cette partie sera détaillée dans la section consacrée à mes missions, car j'ai travaillé sur cette analyse. En résumé, cette tâche nous permet de déterminer si les textes sont objectifs ou subjectifs. Les textes objectifs sont

classés comme neutres puisqu'ils ne dégagent aucun sentiment, tandis que pour les textes subjectifs, le modèle identifie également la polarité du sentiment exprimé.

Une fois que tous nos clusters sont formés, nous utilisons des **word clouds** (nuages de mots). Cela nous permet de visualiser les termes les plus fréquents dans chaque groupe ce qui aide à identifier les thèmes dominants ou la thématique générale de chaque cluster. Un word cloud est une représentation visuelle des mots les plus utilisés dans un texte ou un ensemble de textes où la taille de chaque mot est proportionnelle à sa fréquence d'apparition.

La fonction, qui génère les word clouds, nous donne la possibilité de supprimer les stop words, c'est-à-dire les mots très courants comme "le", "et", "de"... qui n'apportent pas de valeur significative à l'analyse. En éliminant ces mots, le word cloud se concentre sur les termes réellement pertinents pour comprendre la thématique du groupe. Voici un exemple ci dessous d'un wordcloud :

Sur ce word cloud, on voit que la thématique générale du cluster est le temps d'attente et plus précisément sur l'idée de réduire le temps d'attente.

Figure 12 : Exemple de word cloud

Enfin, pour conclure, nous allons parler brièvement de la solution A, développée par Arnaud. Les seules différences entre cette solution et la solution P sont qu'elle utilise la norme TF-IDF pour vectoriser les textes et une classification ascendante hiérarchique. Après une analyse et une comparaison des résultats entre les deux solutions, il a été décidé de retenir la solution P comme **solution exploratoire finale**. En effet, les deux méthodes convergent vers les mêmes thèmes et il y avait une concordance générale des clusters. Cependant, en raison de la complexité d'utilisation de la solution A et des meilleures performances (en termes de rapidité) de la solution P, c'est cette solution qui a été retenue.

Solution de suivi des thématiques

Une autre demande du service EMRG était de suivre **l'évolution des retours** clients dans le temps en observant les différentes thématiques au fil des mois. Cette demande a pu être satisfaite grâce à la solution exploratoire, qui a identifié les différents thèmes présents dans nos verbatims , ce qui nous a permis d'utiliser une classification supervisée. En effet, sans connaître les thèmes représentés dans nos verbatim, il n'est pas du tout aisé de réaliser une classification de type supervisé qui, pour rappel, demande de connaître les étiquettes à l'avance. L'utilisation de ce type de classification est plus adaptée pour répondre à cette demande, car après s'être doté d'une base d'apprentissage, le modèle demande très peu de présence et de ressource humaine pour réaliser une nouvelle classification, contrairement à la solution non supervisé où une présence humaine est obligatoire pour pouvoir étiqueter les clusters.

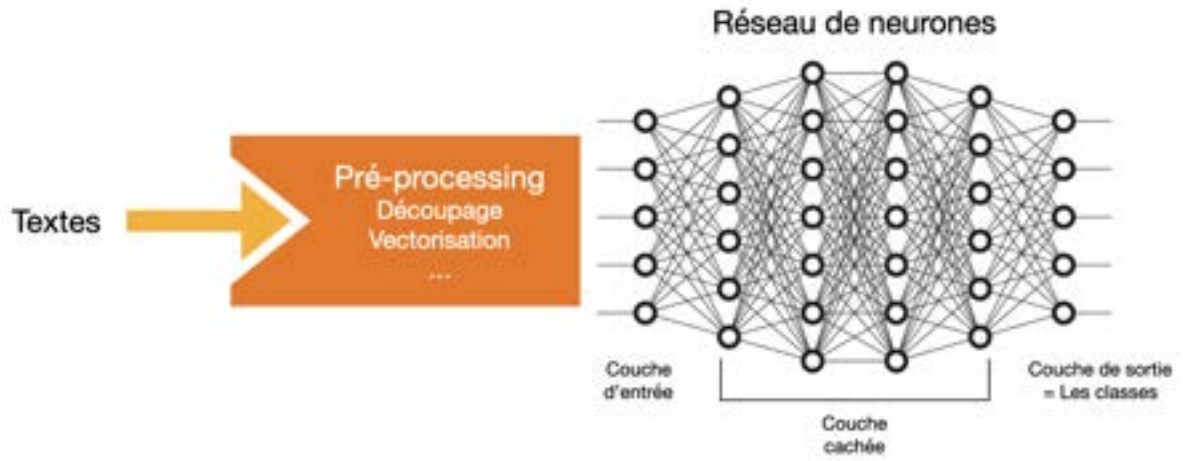


Figure 13 : Classification de texte par réseau de neurones

Ainsi, le but ici est de prédire la classe d'appartenance d'un texte. Pour se faire, nous utilisons un pré-processing qui consiste, comme pour la solution exploratoire, à rendre les verbatim plus exploitables (découpage par genIA) et à les transformer en vecteurs de chiffres (tokenization - vectorisation) puis nous utilisons un **réseau de neurones**. Après avoir transformé les textes en vecteurs numériques, ces chiffres circulent le long des connexions entre les neurones. La somme des poids entrant dans chaque neurone subit une transformation via une fonction d'activation, qui génère une sortie utilisée comme entrée pour la couche suivante, et ainsi de suite. Chaque poids agit comme un paramètre à ajuster pour obtenir la meilleure classification possible. La qualité de la classification finale est évaluée par une fonction de perte, appelée LOSS qui, dans notre cas, est basée sur l'entropie croisée. La LOSS dépend de l'ensemble des poids, ce qui en fait un

problème d'optimisation : il s'agit de trouver la configuration de poids qui minimise cette fonction de perte, généralement par descente de gradient.

Le pré-processing des textes est en 4 étapes , après découpage en idée par genIA on a :

- Elimination des Stop Word (les mots très courants)
- Tokenization
- Embedding

Ainsi, nous utilisons un réseau de neurones composé d'une couche d'entrée , de cellules **LSTM** (Long Short-Term Memory) et enfin d'une couche de sortie qui est une couche DENSE de k neurones (k=nombre de classes). Nous utilisons le mode d'activation SOFTMAX sur cette couche afin d'obtenir une probabilité d'appartenance aux différentes classes. La classe d'affectation d'un individu sera celle présentant la plus grande probabilité d'affectation.
Plus de détails (techniques et mathématiques) sur le réseau de neurones et les cellules LSTM sont disponibles en annexe (AN.2 : Réseau de neurones : cellule LSTM)

Après l'entraînement de notre modèle (avec la méthode train-test split), celui-ci est prêt à être utilisé pour classer nos verbatim. Cependant, étant donné que la classification est ici supervisée, cette méthode présente plusieurs limites dues à divers facteurs. Ces limites seront abordées dans la section suivante.

Limites des Méthodes Utilisées

La classification supervisée des textes issus de sondages présente plusieurs limites, particulièrement dans un contexte où le client ne dispose pas d'une base d'apprentissage au préalable. En effet, pour que cette méthode fonctionne, il est nécessaire de disposer d'un ensemble de données étiquetées où chaque texte est déjà classé selon des catégories imposées. Or, la plupart du temps, les clients n'ont pas cette base d'apprentissage et ne possèdent souvent pas les connaissances en data science nécessaires pour en comprendre l'importance. Cela crée un obstacle majeur, car sans cette **base d'apprentissage**, il est impossible de construire un modèle de classification supervisée efficace.

De plus, pour que la classification supervisée soit efficace et utile, le client doit connaître les thèmes abordés dans les verbatim. Cependant, il arrive que la grille de classification proposée par le client ne couvre pas tous les thèmes présents dans les textes. C'est pour cela que nous proposons souvent de faire un premier poc avec la solution exploratoire afin d'explorer tous les thèmes présents dans les

verbatim . Cette limitation peut entraîner une mauvaise classification des verbatim ou laisser des aspects importants sans catégorie spécifique. Par ailleurs, si de nouveaux thèmes émergent au fil du temps, par exemple d'un mois à l'autre, la classification actuelle ne pourra pas les identifier. Ajouter une nouvelle catégorie nécessite alors de reconstruire entièrement le modèle, ce qui est un **processus long et coûteux**. De même, si la question posée dans le sondage change, tout le travail de modélisation doit être refait, ce qui limite encore plus la **flexibilité** de cette méthode.

En conclusion, la classification supervisée peut être contraignante et manquer de **souplesse** dans le contexte de l'analyse de textes issus de sondages.

La solution exploratoire, bien que puissante pour la classification de textes issus de sondages, présente également plusieurs limites. L'un des défis majeurs réside dans la détermination du **nombre de classes** à initialiser lors de l'utilisation de l'algorithme K-means ++. En effet, trouver le bon nombre de clusters n'est pas une tâche aisée, même avec des méthodes comme celle du coude ou de la silhouette. Une mauvaise estimation peut entraîner des regroupements de verbatim peu cohérents où des thèmes communs sont répartis entre plusieurs clusters ou, au contraire, des clusters contenant des thèmes trop similaires.

Un autre problème vient du fait que des textes exprimant la même idée de manière différente peuvent être séparés en différents clusters, ce qui réduit la cohérence de la classification. Il est alors nécessaire de **regrouper** ses groupes dans un même groupe unique. Nous avons aussi la nécessité d'une vérification humaine pour s'assurer que les textes appartenant à un même groupe sont bien cohérents entre eux et que l'étiquette attribuée à ce groupe reflète correctement le contenu des verbatim. Cette étape de validation est indispensable, mais elle demande un investissement en temps et en expertise.

Enfin, en plus des groupes de textes nécessitant d'être regroupés, il peut arriver que de **nouvelles classes** doivent être créées après coup. Cela arrive lorsque les clusters générés ne correspondent pas parfaitement aux attentes ou à la réalité des thèmes présents dans les verbatim. Ce besoin fréquent d'intervention humaine, que ce soit pour vérifier la classification ou pour étiqueter les groupes manuellement à l'aide d'outils comme les word clouds, souligne la **dépendance** de cette méthode à une expertise humaine continue, limitant ainsi son automatisation et son efficacité à grande échelle.

L'objectif principal de mon stage est de réduire ces limites. Dans la section suivante, nous explorerons les idées que j'ai mises en œuvre pour y parvenir.

Mes missions

Mes Outils de travail

Pour mener à bien mon travail, j'ai disposé d'un ordinateur personnel ainsi que d'un accès à des **machines virtuelles** (VM) via CyberArk. Ces VM m'ont permis de coder en toute sécurité et d'utiliser des outils spécifiques tels que Jupyter. La majorité de mes développements s'est effectuée en Python, principalement sous forme de notebooks. Ces outils ont été essentiels pour la mise en œuvre des solutions durant mon stage.

Les VM auxquelles j'ai eu accès durant mon stage offrent de nombreux avantages particulièrement en termes de sécurité et de puissance de calcul. Hébergées dans un **environnement sécurisé** , ces VM garantissent que toutes les données et le code sont protégés contre les accès non autorisés. De plus, ces machines sont équipées de 32 VCPU, ce qui offre une bonne **puissance de calcul**. Cette puissance nous permet d'utiliser des modèles d'intelligence artificielle générative connus pour être particulièrement gourmands en ressources, mais aussi une bonne puissance de calcul sur de l'exécution de script python. Par contre le manque de GPU se fait sentir sur l'utilisation de genIA. En effet, l'exécution de scripts faisant appel à une genIA pouvant durer plus de 7 jours, l'utilisation de GPU conçues pour exécuter des opérations massivement parallèles laisse espérer des gains de temps importants. Par chance, en juin 2024, des GPU ont été ajoutées aux VM de toute l'équipe ouvrant ainsi de nouvelles portes (on le verra dans la suite) et nous faisant gagner beaucoup de temps de traitement (jusqu'à un facteur de 30).

Dès le début de mon stage, la chose qui m'a été demandée est la mise en place **d'outils de benchmark** : créer un score qui pourrait mesurer la distance entre les résultats d'une classification automatique et un classement fait par un humain, ou entre deux classifications. Et cela que ce soit pour comparer des partitions avec le même nombre de classes, ou avec un nombre de classes différent. Pour ce faire, j'ai considéré plusieurs méthodes :

- **Indice de Jaccard** (J) :

L'indice de Jaccard mesure la similarité entre deux ensembles en calculant le ratio entre la taille de l'intersection et la taille de l'union de ces ensembles. Dans le contexte du clustering, il est utilisé pour comparer la similarité entre deux partitions en se basant sur les paires d'éléments.

J est défini par la formule suivante :

$$J = \frac{|A \cap B|}{|A \cup B|}$$

où :

- $|A \cap B|$ représente le nombre de paires de points qui sont dans le même cluster dans les deux partitions,
- $|A \cup B|$ représente le nombre total de paires de points considérées (dans le même cluster ou non).

L'indice de Jaccard prend des valeurs entre 0 et 1 :

- Un indice de Jaccard de 1 indique une correspondance parfaite entre les deux partitions.
- Un indice de Jaccard de 0 indique qu'il n'y a pas de correspondance entre les deux partitions.

- **indice de Rand ajusté** (ARI) :

Une mesure qui permet d'évaluer à quel point les groupements de données obtenus à partir d'un algorithme de clustering sont similaires aux groupements de données réels , tout en tenant compte du hasard.
l'ARI est défini par la formule suivante :

$$ARI = \frac{\sum_{ij}\binom{n_{ij}}{2} - \left[\sum_i \binom{a_i}{2} \sum_j \binom{b_j}{2}\right] / \binom{n}{2}}{\frac{1}{2}\left[\sum_i \binom{a_i}{2} + \sum_j \binom{b_j}{2}\right] - \left[\sum_i \binom{a_i}{2} \sum_j \binom{b_j}{2}\right] / \binom{n}{2}}$$

où :

- n est le nombre total d'éléments,
- n_{ij} est le nombre d'éléments communs dans la classe i de la première partition et la classe j de la deuxième partition,
- $a_i = \sum_j n_{ij}$ est le nombre d'éléments dans la classe i de la première partition,
- $b_j = \sum_i n_{ij}$ est le nombre d'éléments dans la classe j de la deuxième partition.

L'ARI est compris entre -1 et 1 :

- Un ARI de 1 indique une correspondance parfaite entre les deux partitions.
- Un ARI de 0 indique que la partition obtenue n'est pas meilleure qu'une partition aléatoire
- Un ARI négatif indique que la partition obtenue est pire qu'une partition aléatoire.

- **indice de Fowlkes - Mallow** (FMI) :

Un indice qui évalue la qualité des clusters en fonction des similitudes par paires entre les points de données au sein et entre les clusters.

Le FMI est défini par la formule suivante :

$$FMI = \frac{TP}{\sqrt{(TP + FP)(TP + FN)}}$$

où :

- TP (Vrais Positifs) est le nombre de paires de points qui sont dans la même classe dans les deux partitions,
- FP (Faux Positifs) est le nombre de paires de points qui sont dans la même classe dans la première partition mais dans des classes différentes dans la seconde,
- FN (Faux Negatifs) est le nombre de paires de points qui sont dans des classes différentes dans la première partition mais dans la même classe dans la seconde.

L'indice FMI est compris entre 0 et 1 :

- Un FMI de 1 indique une correspondance parfaite entre les deux partitions.
- Un FMI de 0 indique qu'il n'y a pas de correspondance entre les deux partitions.

- **Différence absolue d'entropie** :

La différence absolue d'entropie est une mesure utilisée pour comparer la qualité des partitions obtenues par un algorithme de classification par rapport à une partition de référence, elle peut être définie comme suit :

$$\Delta H = |H(P) - H(Q)|$$

où :

- ΔH est la différence absolue d'entropie.
- $H(P)$ est l'entropie de la partition prédite P.
- $H(Q)$ est l'entropie de la partition de référence Q.

L'entropie H d'une partition peut être calculée avec la formule suivante :

$$H(P) = -\sum_{i=1}^{k} \frac{|C_i|}{n} \log\left(\frac{|C_i|}{n}\right)$$

où :

- k est le nombre de clusters dans la partition P.
- $|C_i|$ est le nombre d'éléments dans le cluster C_i.
- n est le nombre total d'éléments dans l'ensemble de données.

Une différence d'entropie proche de zéro indique que la partition obtenue est similaire à la partition de référence.

Tous ces outils de benchmark nous seront très utiles afin de comparer plusieurs méthodes de classifications.

Amélioration de la solution exploratoire

Pour commencer, j'ai commencé par travailler sur l'analyse de sentiments : est-ce possible de l'améliorer ? Comprendre le modèle qu'on utilisait déjà (BARThez) afin de voir si c'est possible de l'optimiser, existe-il d'autres modèles plus efficaces ?

Avant d'aller plus loin, nous allons expliciter dans le détail le modèle utilisé. Il s'agit du modèle pré-entraîné **BARThez**, développé par des élèves de l'école polytechnique. Le modèle repose sur l'utilisation de BART et de Fine-Tuning grâce auxquels il a été entraîné à reconnaître le sentiment d'une phrase. Ils ont utilisé des données textuelles issues d'internet et de la littérature française.

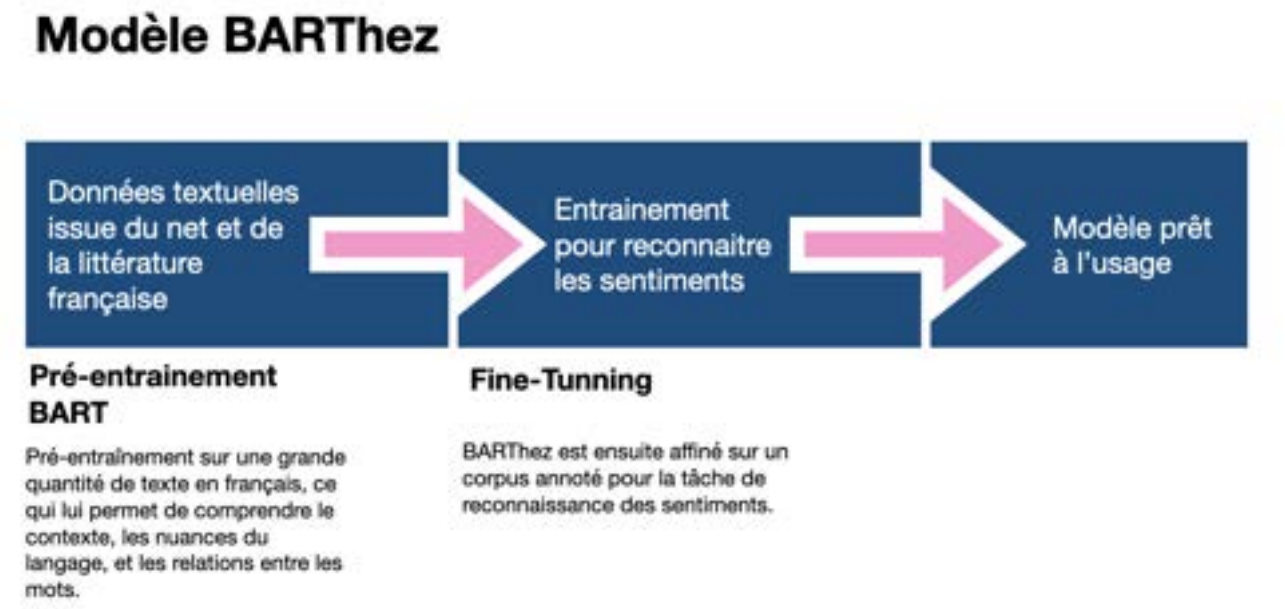


Figure 14 : Modèle BARThez

BARThez s'inspire du modèle **BART** (Bidirectional and Auto-Regressive Transformers) qui est un modèle de langage développé par Facebook AI, conçu pour les tâches de NLP comme la traduction automatique, le résumé de texte …
BART utilise deux architectures : un encodeur bidirectionnel et un décodeur auto-régressif, une combinaison qui lui permet de comprendre le contexte des phrases tout en générant du texte de manière séquentielle.

Un **encodeur bidirectionnel** est une architecture utilisée dans les modèles de traitement de NLP qui lit un texte dans les deux sens : de gauche à droite et de droite à gauche, le but est de capturer le contexte complet autour de chaque mot, en tenant compte des mots qui le précèdent et qui le suivent.

Un **décodeur auto-régressif** est une architecture utilisée pour générer du texte de manière séquentielle, lors de chaque étape le modèle prédit le mot suivant en se basant uniquement sur les mots précédemment générés.
Le squelette commun et utilisé dans les deux architectures est basé sur les Transformers une architecture basée sur des réseaux de neurones.

Grâce à l'encodeur et au décodeur, BART utilise un entraînement non supervisé en utilisant une approche dite denoising autoencoder, où le modèle apprend à reconstruire un texte original à partir d'une version bruitée (en masquant certains mots ou en réorganisant les mots d'une phrase). Ceci permet au modèle d'apprendre à comprendre la structure du langage et les relations entre les mots.
Cette phase de pré-entraînement est commune à BART et BARThez. Suite à cette phase d'apprentissage, BARThez utilise une technique d'affinage dite **fine-tuning**. Il est affiné sur un corpus annoté pour la reconnaissance des sentiments. Le corpus contient des textes avec des étiquettes indiquant le sentiment, lors de cette phase, le modèle ajuste des paramètres pour améliorer sa capacité à analyser les sentiments d'un texte. Après tout ça, le modèle est prêt à l'usage.

Après avoir préparé le modèle, il est possible de l'appliquer directement à nos textes afin de réaliser une analyse des émotions. Le modèle produit un **vecteur en sortie** à chaque texte entrant, avec le premier index représentant l'intensité négative et le second index représentant l'intensité positive. En effectuant une simple opération sur ces valeurs, il est possible d'évaluer la neutralité ou non d'une phrase en mesurant l'équilibre entre ces deux intensités et aussi il est possible de déterminer le sentiment global de la phrase, qu'il soit négatif, positif ou neutre.

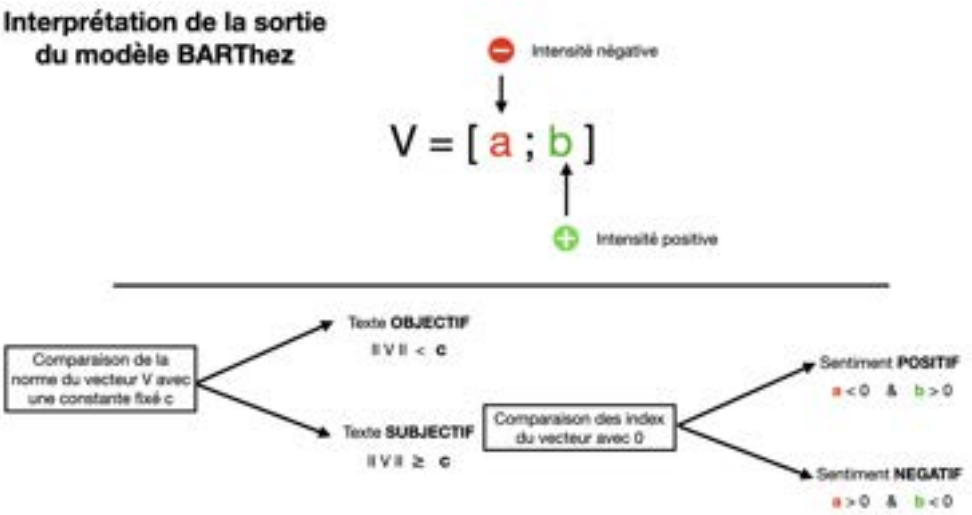


Figure 15 : Interprétation de la sortie du modèle BARThez

Le schéma montre l'interprétation de la sortie du modèle BARThez pour l'analyse de sentiment. Pour un texte donné en entrée, le modèle génère un vecteur V=[a ; b], où a représente l'intensité négative et b l'intensité positive d'une phrase. La première étape consiste à comparer la norme $\|V\|$ du vecteur à une **constante c fixé** par nous même: si $\|V\|$ est inférieure à c, le texte est jugé objectif, sinon il est subjectif. Pour les textes subjectifs, on détermine le sentiment en comparant les valeurs de a et b : si a<0 et b>0, le sentiment de la phrase est positif, sinon si a>0 et b<0 le sentiment est négatif.

Avant mon arrivée, Philippe utilisait ce modèle avec une constante c fixée arbitrairement, sans véritable évaluation pour déterminer si c'était le meilleur choix. Dans le but d'améliorer l'analyse des sentiments, nous avons décidé d'utiliser des scores pour mesurer l'efficacité des différentes classifications réalisées avec BARThez en testant plusieurs valeurs de c, ainsi que d'autres modèles d'analyse de sentiments. Le tableau récapitulatif des scores ci-dessous montre les résultats obtenus. Ces scores ont été calculés en utilisant une classification de référence provenant de la base de données des retours d'un sondage EMRG, où 500 individus ont été annotés manuellement : 0 pour neutre, -1 pour négatif, et 1 pour positif.

Score :	Indice de Jaccard	ARI	FMI
BARThez c = 2	0.79	0.55	0.71
BARThez c = 1.14	**0.86**	**0.66**	**0.75**
BARThez c = 0.7	0.42	0.23	0.28
CamemBERT	0.56	0.37	0.30
FlauBERT	0.65	0.50	0.43
TextBlob*	0.21	0.18	0.11

(*): utilisé avec un module de traduction

Nous avons essayé plusieurs valeurs de la constante c pour le modèle BARThez, et d'autres modèles tels que camemBERT , flauBERT et TextBloB. Les deux premiers reposent sur une architecture BERT et textBloB sur du BART à l'instar de BARThez. Ce dernier est un modèle entraîné sur un large corpus de texte en anglais, on utilise donc un module de traduction en anglais afin de l'utiliser.

Les résultats présentés dans le tableau montrent clairement que le modèle le plus performant est **BARThez avec une constante fixée à 1.14**. J'ai déterminé cette

valeur en testant 100 textes neutres, c'est-à-dire des textes n'exprimant aucun sentiment, et en explorant toutes les valeurs de c comprises entre 0.5 et 3(par incréments d'un centième). La valeur de 1.14 s'est révélée être celle qui classait le plus grand nombre de textes comme neutres. BARThez avec une constante de 2, qui était le choix initial de Philippe, offre également de bonnes performances, comme le montre le tableau. En revanche, d'autres modèles, tels que camemBERT et flauBERT, se révèlent moins efficaces que BARThez. Enfin, Text Blob, utilisé après traduction des textes, affiche des performances très faibles, probablement en raison des pertes de sens et d'informations contextuelles qui peuvent survenir lors de la traduction.

Traçons la matrice de confusions du classement réalisé par BARThez avec c= 1.14

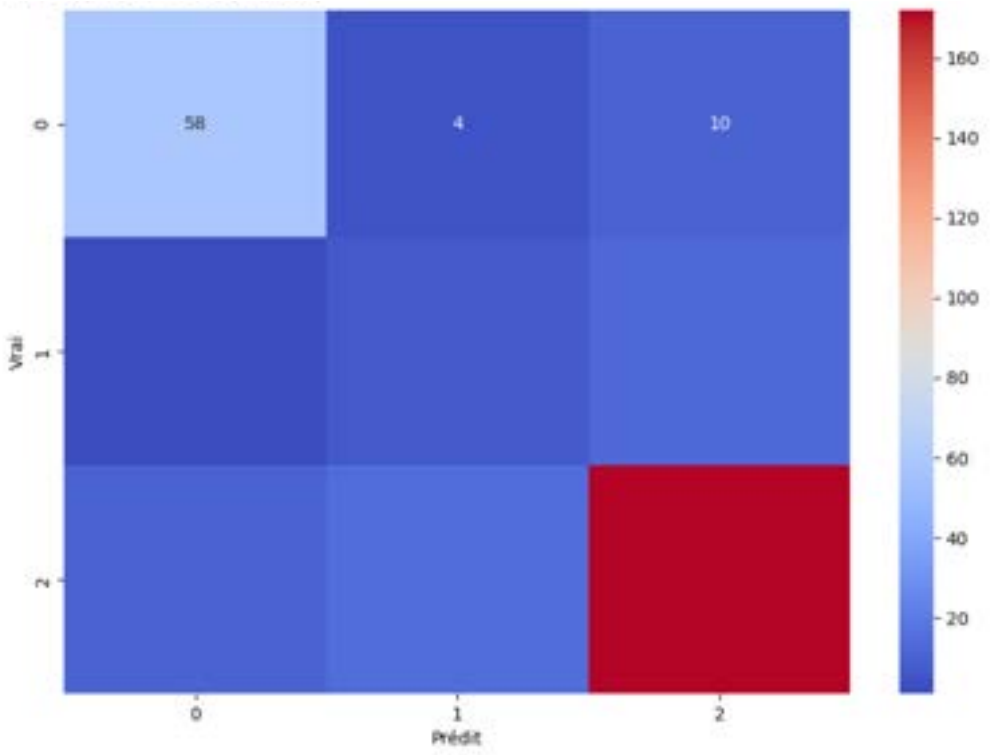


Figure 15 : Matrice de confusion

On remarque grâce à la matrice de confusion, que le modèle ne se trompe pas trop sur la neutralité des textes mais il persiste des confusions entre les négatifs et les positifs.

Nous avons donc trouvé la meilleure manière d'analyser les sentiments d'un texte. Toujours dans une dynamique d'amélioration de la solution exploratoire, nous avons eu l'idée, avec Philippe et Arnaud, de développer une nouvelle approche basée sur les formes fortes. En effet, le principal problème de cette solution réside dans le choix du nombre de classes à initialiser pour le k-means. La méthode du coude et le score de silhouette sont difficilement, voire pas du tout, interprétables

dans notre contexte. Une idée a donc émergé : pourquoi ne pas commencer par effectuer une première classification dont nous ne conservons que les clusters dits forts, c'est-à-dire ceux composés d'individus s'étant regroupés de manière significative. Ensuite, dans un second temps, nous pourrions réaliser une seconde classification avec un nombre de classes inférieur à celui utilisé lors de la première classification, en utilisant uniquement les individus qui ne se sont pas regroupés dans les clusters forts.

La forme forte est un concept développé par Yves Diday en 1971. Il fait référence à un ensemble d'éléments qui sont classés ensembles à plusieurs reprises au cours de plusieurs essais ou itérations, l'idée est que les éléments qui résistent aux variations aléatoires et restent groupés au sein d'une même classe font preuve d'homogénéité et d'une très forte stabilité. C'est-à-dire que si un ensemble d'éléments est régulièrement classé de manière identique sous différentes conditions, alors cet ensemble peut-être considéré comme une forme forte.

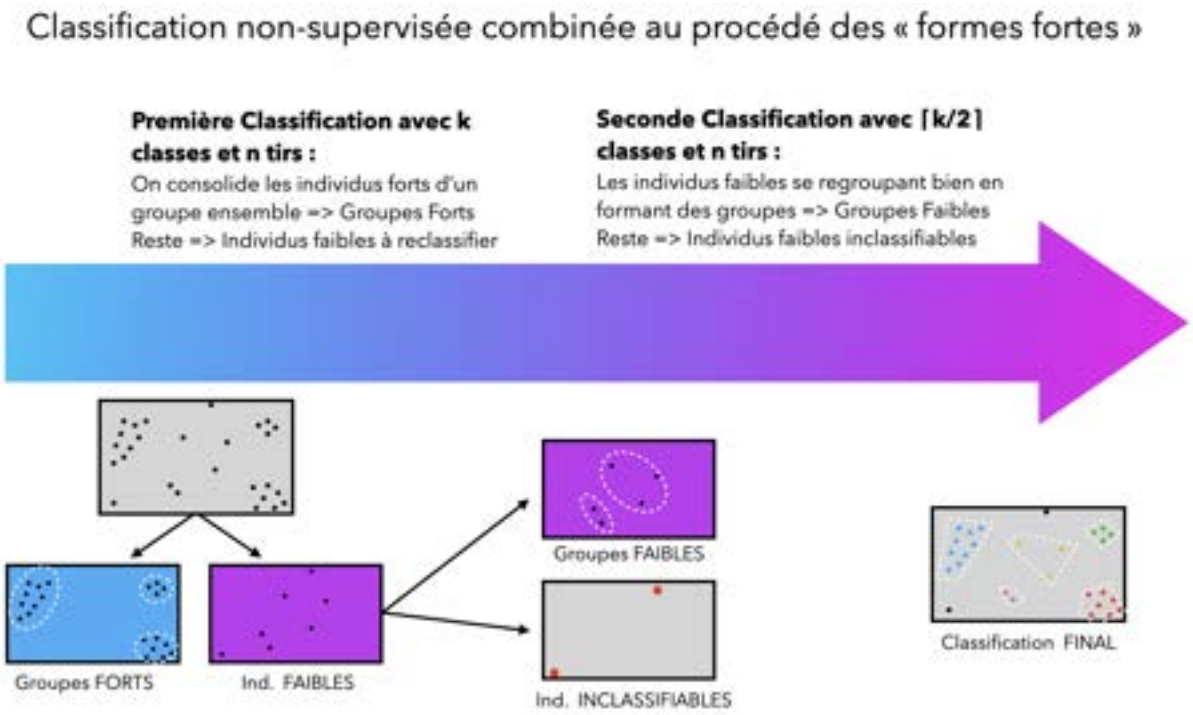


Figure 16 : Classification non-supervisée combinée aux « formes fortes »

Ce schéma montre le fonctionnement de la méthode de classification non supervisée combinée au procédé des formes fortes pour améliorer la qualité des regroupements. La démarche se compose de deux étapes :

- Première Classification avec k classes :
  Dans un premier temps, un algorithme de classification (k-means++) est utilisé pour diviser les données en k classes. L'objectif est de consolider les individus qui se regroupent fortement, formant ainsi des **groupes forts**. Ces groupes sont caractérisés par une forte cohésion entre les individus, ce qui signifie qu'ils partagent des caractéristiques similaires de manière significative. Les individus qui ne s'intègrent pas bien dans ces groupes forts sont classés comme **individus faibles** et sont destinés à être reclassés dans une seconde étape.

- Seconde Classification avec ⌈k/2⌉ classes **:**
  Dans cette étape, les individus faibles issus de la première classification sont regroupés à nouveau, cette fois avec un nombre réduit de classes, soit [k/2]. Le but est de former des **groupes faibles** en rassemblant les individus qui, bien que moins cohésifs que ceux des groupes forts, montrent tout de même des similarités suffisantes pour être regroupées. Les individus qui ne parviennent toujours pas à s'intégrer dans ces nouveaux groupes sont considérés comme **individus non classifiables**.

Le processus final permet d'obtenir une classification plus fine et précise, où les données sont réparties en groupes forts et faibles, tout en identifiant les individus qui ne peuvent être classés de manière cohérente. Cette approche améliore la qualité de la classification en concentrant l'attention sur les regroupements les plus significatifs et en réduisant les incertitudes liées à des individus mal classifiés dans une approche traditionnelle.

Après avoir exploré la théorie des formes fortes et mis en place une méthode basée sur ce concept, il est temps de passer à la phase suivante. J’ai développé une fonction automatisée qui intègre ces principes, permettant ainsi de réaliser tout ce processus sans intervention manuelle. Cette fonction garantira non seulement une meilleure cohérence dans l’application de la méthode, mais aussi un gain de temps.

Dans l’idée de gagner du temps et de minimiser le nombre d’interventions humaines, j’ai décidé d'ajouter l’étape de production des word clouds de chaque cluster formé par l’algorithme, simplifiant ainsi l’étiquetage des groupes créés.

Voici deux schémas qui détaillent et qui expliquent la fonction en détail :

Algorithme : Classification non supervisée procédé des formes fortes

Require : Une base de données contenant les laïus et les laïus vectorisé , le nombre de classes voulue pour la classification (**k**) , le nombre de tirs à effectuer (**p**)

1: **Première Classification** (k classes et p tirs) : p fois un kmeans sur nos données => p affectations à une classe pour chaque individus de la bdd

2: On met en évidence nos **Groupes forts** : Une classe est considérée comme un groupe fort si au moins 2 % de la population totale a été regroupée dans cette classe au moins 70 % du temps sur les p tirs.

3: On classe chaque individus qui a été classé au moins 50% du temps sur les p tirs dans un même groupe fort et le reste sont des **ind.Faibles** a reclassifier

4: **End if** le pourcentage d'individus faibles est inférieur à 10%

5: **else**

6: **Seconde Classification** ([k/2] classes et p tirs) : p fois un kmeans sur les ind.Faibles

7: On classe chaque individus qui a été classé au moins 50% du temps sur les p tirs dans un même groupe

8: Si un ind.Faible n'a pas été classé dans un même groupe au moins 50 % du temps sur les p tirs, alors aucune classe ne lui est attribuée.

9: **End**

10: Utilisation de **WordCloud** pour étiqueter chaque groupe

11: Si besoin, regrouper les groupes aux thèmes communs

Figure 17 : Algorithme de la fonction

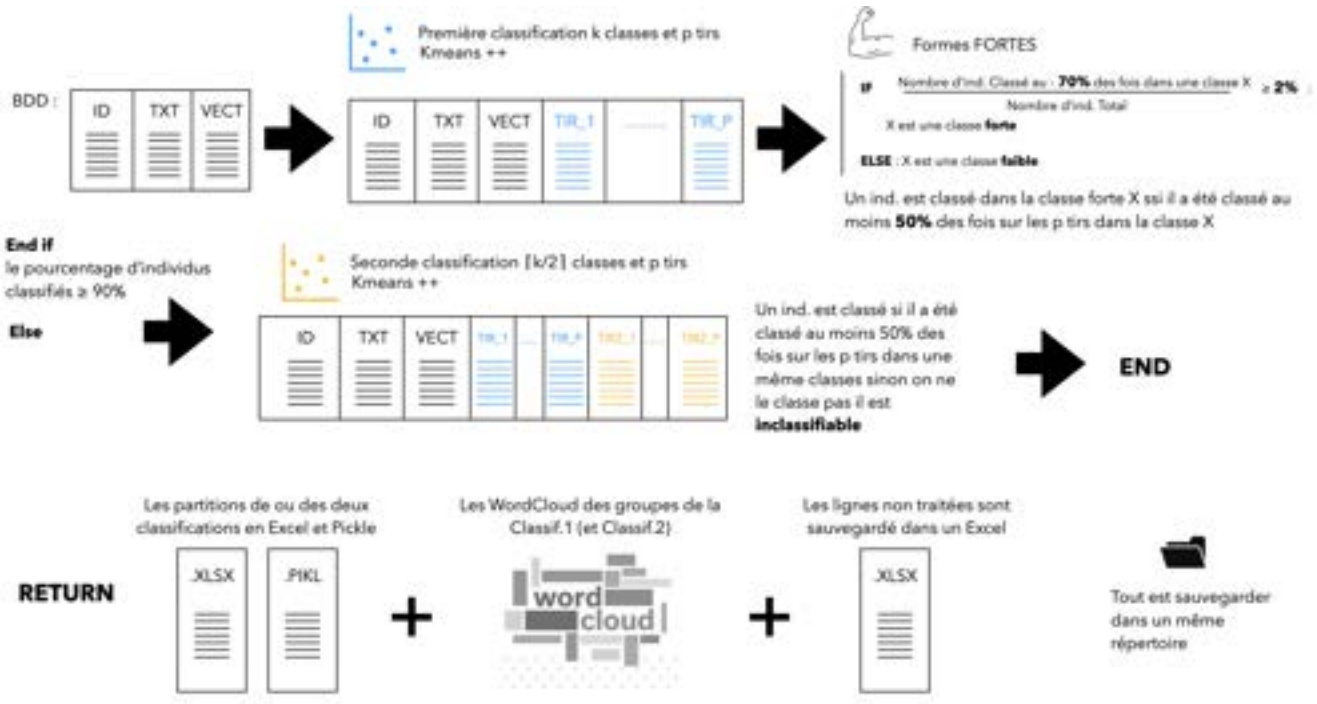


Figure 18 : Schéma détaillé de la fonction

Ces images décrivent la fonction qui implémente l'algorithme des formes fortes pour la classification non supervisée. Elle prend en entrée une base de données de textes vectorisés et réalise d'abord un premier K-means ++ avec k classes, itéré p fois. Les individus fortement regroupés sont identifiés comme appartenant à des groupes forts. Ensuite, si moins de 90% des individus sont classés à la fin du premier tir, un second tir de p classification est effectué sur les individus restants, en utilisant un nombre de classes égal à la partie entière supérieure de k divisée par deux. Les résultats finaux incluent les partitions des deux classifications et les word clouds pour étiqueter les groupes. Enfin les individus non classifiables sont sauvegardés séparément.

Pour finir, il reste une étape où nous devons fusionner les groupes partageant des thèmes similaires en un seul groupe. Et il reste à créer une partition finale en combinant la classification obtenue lors du premier tirage avec celle réalisée au second tirage.

Dans l'optique de maximiser le gain de temps, j'ai également implémenté une technique de **parallélisme** pour l'exécution des tâches de classification. Elle permet de diviser les opérations en plusieurs sous-tâches qui sont ensuite exécutées simultanément sur les différents cœurs du CPU. Ainsi, plutôt que de lancer les itérations de classification de manière séquentielle, elles sont toutes effectuées en parallèle, réparties sur les 32 cœurs disponibles de notre machine virtuelle. Grâce à cette approche, nous avons considérablement réduit le temps nécessaire pour exécuter les p itérations d'une classification, rendant le processus bien plus rapide.

Maintenant que nous avons développé notre nouvelle méthode de classification, il est essentiel de la tester pour évaluer son efficacité. Pour ce faire, nous allons la comparer à une classification de référence composée de 500 textes provenant de la base de données EMRG. Cette comparaison nous permettra de générer des scores de classification, que nous confronterons ensuite aux scores de la solution d'origine développée par Philippe, ainsi qu'à ceux obtenus par d'autres méthodes de classification. Cette **analyse comparative** nous aidera à déterminer la performance de cette nouvelle approche.

Voici un tableau récapitulatif des scores obtenus par les différentes méthodes, ces scores sont obtenus en confrontant les partitions prédictive obtenues avec la partition de référence.

Nous allons comparer 5 méthodes, la solution P qui utilise plusieurs itérations d’un k-means ++ , la solution fraîchement développé avec les formes fortes , DBscan , ensuite une variante de la solution des formes fortes où au lieu d’utiliser un

k-means ++ on utilise une autre méthode de classification, k-médoïde, et enfin la dernière méthode est le spectral clustering

Score :	Indice de Jaccard	ARI	Diff-entropie
Solution P	0.56	0.43	0.22
K-means ++ Forme Forte	**0.68**	**0.55**	**0.13**
DBscan	0.0	0.0	1.98
K-médoïde Forme Forte	0.41	0.25	0.30
Spectral Clustering	0.18	0.15	0.37

Il s'est avéré que le clustering DBSCAN ne fonctionne pas du tout dans notre contexte. La raison principale est la dimensionnalité élevée des textes vectorisés, ce qui explique l'inefficacité totale de DBSCAN. De même, le clustering spectral a également produit des résultats décevants avec un indice de Jaccard de seulement 0,18. En ce qui concerne la classification par k-médoïde, les résultats obtenus ne surpassent pas ceux de la solution P. Parmi toutes les méthodes testées, la seule qui a offert de meilleurs résultats est celle utilisant les formes fortes avec k-means ++ .

Mise en pratique de la solution "forme forte"

Après avoir vu que la méthode des formes fortes que j'ai développée avec l'aide de Philippe et Arnaud offrait les meilleurs résultats en terme de classification, j'ai été missionné de réaliser un poc de cette solution pour analyser les verbatim de La Banque Postale, LBP qui pourrait devenir un nouveau client de l'offre verbatim. Ils ont ainsi demandé une exploration à l'aveugle des réponses à un sondage datant de juin 2024. On m'a donc confié cette mission, et pour ce faire j'ai utilisé la méthode et la fonction tout récemment développée. Je vais ici dérouler et expliquer toutes les étapes.

Tout d'abord la demande du client était de classer d'une part les laïus qui expriment une amélioration et d'autre part des textes exprimant un ressenti (une appréciation). C'est dans ce contexte là que nous avons décidé d'utiliser l'ia generative qui, en plus de découper tous nos laïus en idées, pourrait faire la part entre les idées qui expriment une amélioration et celles qui expriment une appréciation. Pour ce faire, Arnaud a rédigé un nouveau prompt spécialement conçu pour accomplir cette tâche, en utilisant le même LLM fourni par Ollama. Ce

nouveau prompt, très détaillé, permet de créer pour chaque laïus un dictionnaire associé comportant deux entrées : la première regroupe les idées exprimant des améliorations, tandis que la seconde rassemble celles exprimant des appréciations (il est possible qu'une des deux entrées reste vide). Une fois cette étape terminée, les textes sont formatés, puis deux nouvelles bases de données sont créées : l'une contenant les textes axés sur les améliorations et l'autre sur les appréciations. Ensuite, nous appliquons l'embedding USE pour vectoriser les textes. Les laïus liés aux améliorations sont ensuite passés dans la fonction de classification, tandis que les appréciations sont soumises à une analyse de sentiments. Cette dernière permet de séparer les appréciations en deux bases de données distinctes : l'une pour les appréciations positives et l'autre pour les négatives, qui sont ensuite chacune classifiées séparément..

Nous allons détailler la mise en pratique de la classification des laïus améliorations. Tout d'abord, nous avons 2330 textes. Nous nous posons la question de savoir combien de classes choisir pour le premier tir de classification. Nous utilisons la méthode du coude et de la silhouette dont voici le résultat :

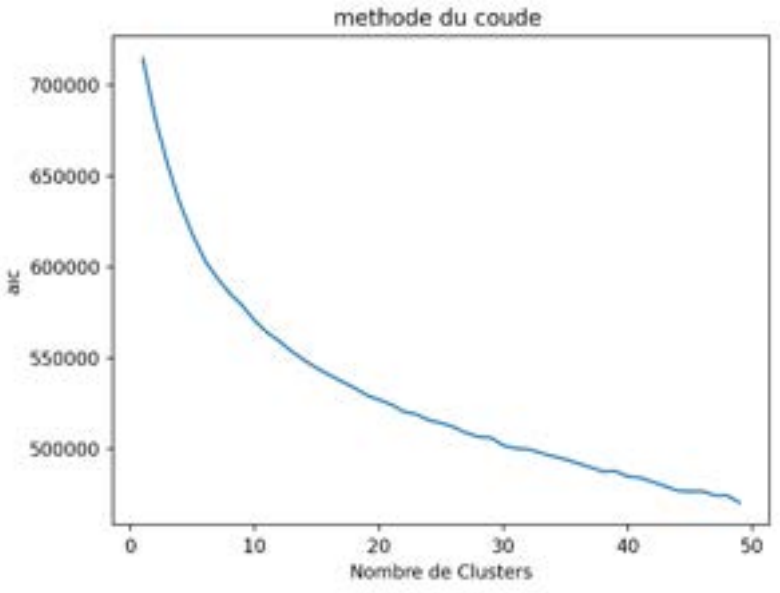


Figure 19 : Méthode du coude

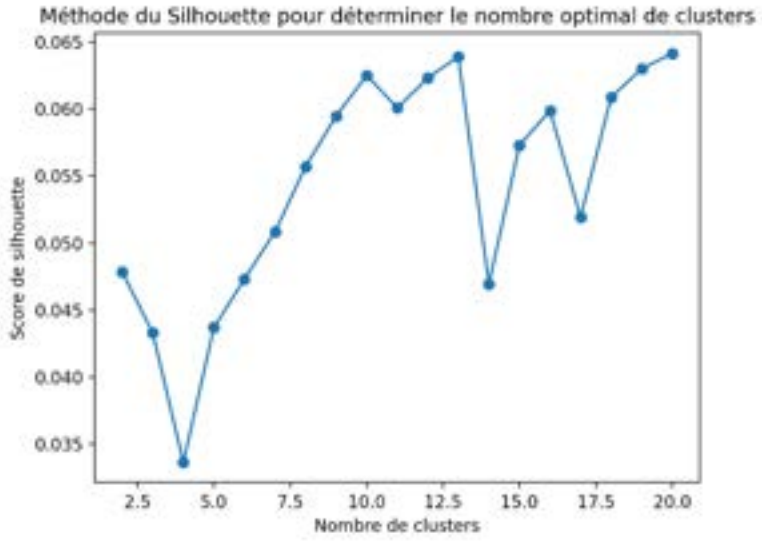


Figure 20 : Méthode Silhouette

Comme on peut le voir, les deux méthodes ne nous permettent pas de savoir efficacement le nombre de classes à initialiser et c'est pour cette raison que nous décidons de choisir un nombre de classe k assez haut. Nous prendrons donc k =12 pour le premier tir de classification et décidons de faire 7 itérations, (donc p=7), Voici le résultat retourné par la fonction :

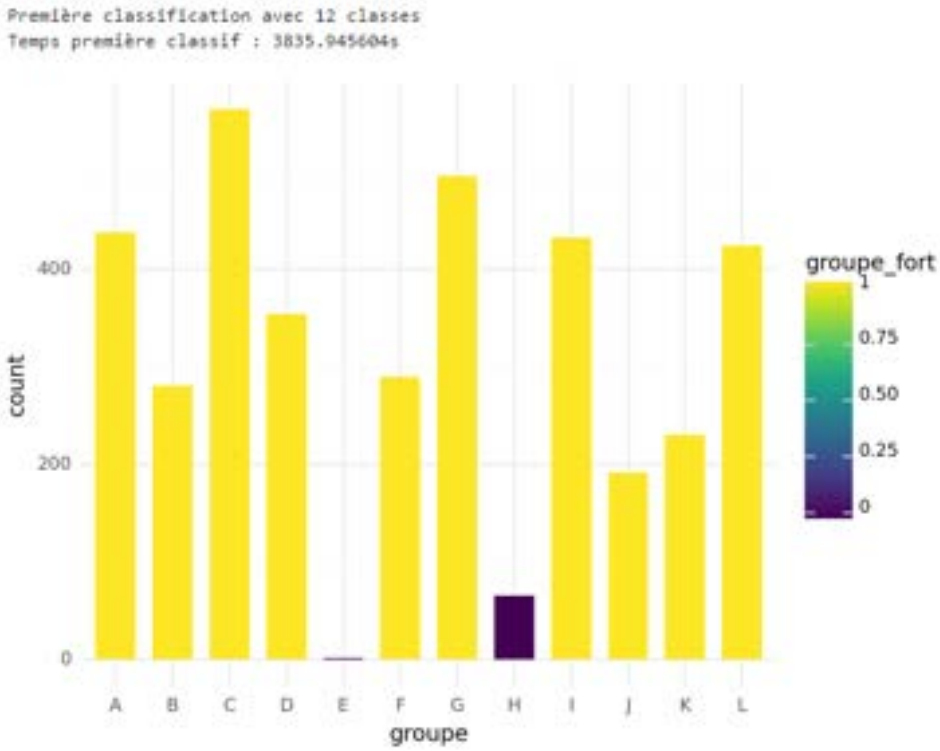


Figure 21 : Résultat de la fonction

On a donc au total 10 groupes forts et 6 groupes faibles , le but est maintenant d'attribuer un thème à chacun de ces groupes puis de voir si des regroupements sont possibles, pour ensuite créer notre partition finale. Voici deux tableaux qui récapitulent toutes les étiquettes données à nos groupes , les Word clouds qui nous ont permis d'étiqueter ces groupes sont disponibles en Annexe (voir An.3 : WordCloud verbatim LBP - Améliorations).

Thèmes attribué aux groupe forts :

Groupe FORT	Thème	Effectif
N°1	Meilleure attitude des téléconseillers	360
N°2	Plus à l'écoute des clients	101
N°3	Réduire le temps d'attente	396
N°4	Amélioration divers services bancaires	208
N°5	Apporter plus de solutions aux problèmes client	172
N°6	Répondre plus vite	163
N°7	Meilleure attitude des téléconseillers	130
N°8	Améliorer les aptitudes des téléconseillers	218
N°9	Apporter de meilleurs renseignements	96
N°10	Améliorer la qualité du service	256

Thèmes attribué aux groupe faibles :

Groupe FAIBLE	Thème	Effectif
N°1	Améliorer la qualité de la communication (téléphone)	89
N°2	Amélioration divers services bancaires	30
N°3	Amélioration divers services bancaires	61
N°4	Amélioration divers services bancaires	55
N°5	Amélioration divers services bancaires	20
N°6	Améliorer la qualité de la communication (téléphone)	112

Comme on peut le voir, on peut regrouper certains groupes et on va donc consolider les deux partitions en une partition finale.
Partition finale en 7 Classes (les laïus données en exemple sont anonymisés) **:**

- **Attitude des téléconseillers** (groupe fort 1 + fort 2 + fort 7)

Effectif = 591, exemple de laïus :

- « Être plus courtois envers les clients »
- «Traiter les clients anciens avec plus de considération »
- « Écouter les clients plus attentivement »
- « Diminuer le ton agressif »

- « Recruter des gens polis et sympathiques »

- **Aptitude des téléconseillers** (groupe fort 8)

Effectif = 218, exemple de laïus :

- « Former les téléconseillers sur les transactions internationales »
- « Avoir des téléconseillers plus professionnels qui respectent leurs engagements (appel en retour) »
- « Meilleure connaissance des produits par les téléconseillers »

- **Temps d'attente** (groupe fort 3 + fort 6)

Effectif = 560, exemple de laïus :

- « Répondre plus rapidement aux appels »
- « Réduire le temps d'attente »
- « Réduire le temps d'attente avant de parler à un conseiller »

- **Service bancaires** (groupe fort 4 + faible 2 + faible 3 + faible 4 + faible 5)

Effectif = 374, exemple de laïus :

- « Disposer d'un service d'urgence 24h/24 pour les fraudes »
- « Envoyer les relevés de compte à la date prévue »
- « Supprimer ou simplifier le certicode »

- **Qualité du service** (groupe fort 10 + faible 1 + faible 6)

Effectif = 457, exemple de laïus :

- « Améliorer l'attente en ligne »
- « Améliorer globalement le service »
- «Améliorer la qualité de l'appel »

- **Retour aux problèmes client** (group fort 5)

Effectif = 172, exemple de laïus :

- « Répondre efficacement aux problèmes des clients »
- « Donner la solution au premier appel »
- « Améliorer l'efficacité de la résolution des problèmes »

- **Renseignements** (groupe fort 9)

Effectif = 96, exemple de laïus :

- « Fournir plus de détails sur les procédures à suivre »
- « Fiabiliser les informations données par le téléconseiller »

- « Expliquer clairement les incidences »

On réalise également le diagramme en baton de cette partition, un outil idéal pour visualiser la répartition des groupes.

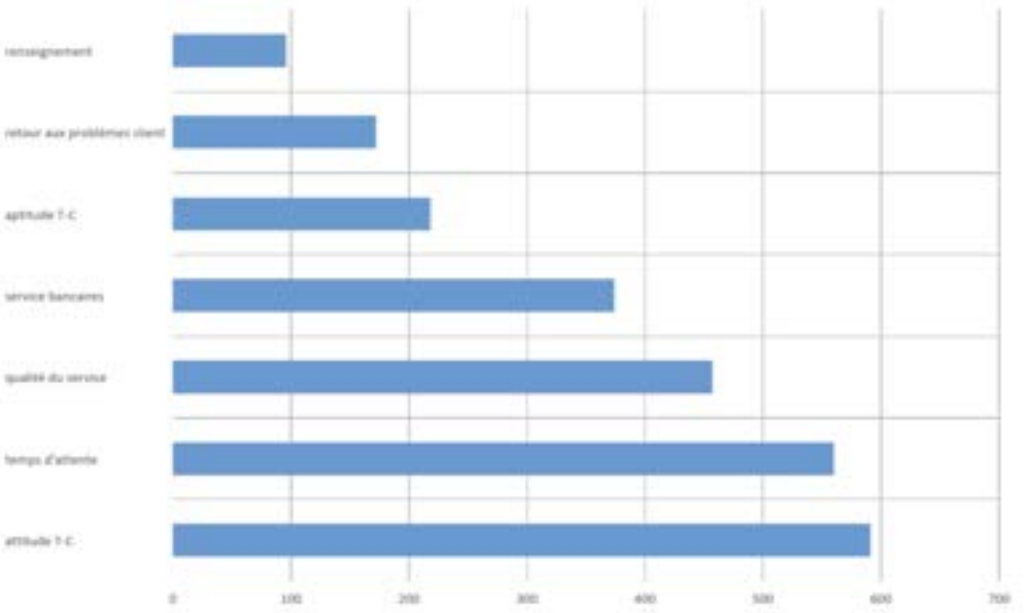


Figure 22 : Diagramme en bâton de la partition finale

Nous avons ainsi détaillé toute la procédure permettant de créer la partition finale pour les laïus axés sur les améliorations. De la même manière, nous avons appliqué une approche similaire pour les appréciations positives et négatives, bien que nous ne détaillerons pas cette analyse ici. Une fois ce travail achevé, j'ai présenté les résultats à Arnaud et Philippe, qui les ont validés. Grâce à leur approbation, j'ai ensuite pu exposer cette solution au client, qui s'est montré également satisfait. En conséquence, le client nous a demandé de réaliser un second POC avec une base de données plus récente que celle fournie, afin de valider les thématiques identifiées et d'analyser leur évolution dans le temps.

Pour conclure sur la méthode, on peut dire que la solution que j'ai développée, ainsi que la fonction associée, fonctionne très bien. Les retours ont été positifs, et il n'a pas été nécessaire d'intervenir manuellement, ce qui témoigne de la robustesse de notre approche. De plus, grâce au parallélisme et à l'automatisation des word clouds, nous avons considérablement gagné en efficacité et en temps.

Solution supervisé : s’affranchir d’une base d’apprentissage

La solution supervisée, bien qu'efficace pour de nombreuses tâches, se heurte à une contrainte majeure : la nécessité de disposer d'une base d'apprentissage préalablement étiquetée, ce qui peut s'avérer complexe et chronophage à mettre en place. Pour surmonter cette limitation, nous allons explorer l'utilisation de modèles capables de classifier des textes avec des thèmes imposés sans nécessiter de telles bases, voire d'exploiter l'IA générative pour accomplir cette tâche. L'arrivée des GPU a considérablement accéléré les processus d'entraînement et d'inférence, rendant l'IA générative plus accessible et plus rapide à implémenter dans ce type de solution.

Le premier modèle qu’on va essayer est le modèle **MoritzLaurer**. Il permet de classer nos laïus dans des groupes déjà étiquetés, sans nécessité une base d'entraînement et une phase d'entraînement. Le modèle marche comme suit :
On lui fournit une hypothèse et des classes , par exemple :

- Hypothèse : “ce texte traite de”
- Classes : “d'écologie” , “de politique” , “de technologie”

Ensuite le modèle affecte à chaque texte une probabilité d'appartenance pour chacunes des classes, dans notre cas on affecte un texte à la classe où la probabilité d’appartenance est la plus grande.
Ce Modèle a deux problèmes , la sensibilité à l'hypothèse posée et aux classes définies, et il peut aussi avoir du mal à se décider (les probabilité d’appartenance peuvent être proches d’une classe à l’autre).

La seconde méthode qu’on met en place est d'utiliser notre **IA générative** afin de classer nos textes dans des thèmes imposés. Pour ce faire, nous rédigeons un nouveau prompt qui permet cela. Comme pour le modèle MoritzLaurer, un problème assez contraignant est l'hypersensibilité au prompt. En effet, un prompt qui diffère très peu peut créer deux partitions complètement différentes. L'arrivée des GPU dans les VM nous a fortement aidés car sans les GPU, les temps de traitement auraient été très longs. L’idée n`aurait même pas été validée sans les GPU dû à un coup en temps trop élevé qui aurait entravé l'utilisation à grande échelle.

Dans le but de mesurer l’efficacité des deux méthodes par rapport à la classification supervisée, on a décidé de comparer deux partitions obtenues par nos méthodes (Moritz-Laurer et modèle 3.1:70B d’Ollama) à la partition obtenue par le réseau de neurones LSTM, on utilisera l’indice de Jaccard et le FMI afin de comparer nos méthodes.

On a utilisé 1500 verbatim d'une base de données du service EMRG déjà classés par la solution supervisée dans 4 classes : "accompagnement", "retour" ,"Process", "épreuves", pour classer avec nos modèles on va utiliser plusieurs stratégies, la première stratégie consiste tout simplement à classer nos textes dans les 4 thèmes, la seconde elle, est un peu plus complexe , on découpe nos thèmes en plusieurs sous-thèmes, puis on classe nos textes dans les sous thèmes, pour ensuite après les consolider, et enfin obtenir la partition demandé avec nos thèmes.

Les résultats obtenus par les deux méthodes sont présentés dans le tableau ci-dessous.

Score :	Indice de Jaccard	FMI
Moritz-Laurer Stratégie 1	0,57	0.44
Moritz-Laurer Stratégie 2	0.43	0.22
Ollama : 3.1:70B Stratégie 1	0.65	0.51
Ollama : 3.1:70B Stratégie 2	**0.73**	**0.58**

Globalement, les résultats montrent que le modèle Ollama 3.1:70B, surtout lorsqu'il est utilisé avec la stratégie 2, offre des performances nettement supérieures à celles de la méthode Moritz-Laurer. Le modèle de Ollama semble non seulement mieux capturer la structure thématique des textes, mais aussi générer des clusters de meilleure qualité. Cela suggère que le recours à des sous-thèmes dans la stratégie de classification pourrait être une approche prometteuse pour améliorer la précision et la qualité des partitions obtenues avec le modèle ollama : 3.1:70B .

S'affranchir d'une base d'apprentissage traditionnelle reste un défi de taille dans le domaine de la classification supervisée. Cependant, avec les avancées récentes en intelligence artificielle, il devient de plus en plus envisageable de contourner cette dépendance. Les méthodes explorées dans cette étude montrent qu'il est possible de classer des textes sans passer par une base d'apprentissage, tout en obtenant des résultats proches à ceux des approches supervisées classiques.

En améliorant et en perfectionnant ces méthodes, nous pourrons peut-être un jour nous affranchir totalement de la nécessité d'une base d'apprentissage. Cela offrirait une flexibilité considérable pour répondre aux demandes des clients, qui souhaitent classer leurs verbatim dans des catégories spécifiques et imposées. L'avenir de la classification sans apprentissage semble prometteur et il est possible qu'avec des innovations futures, nous parvenions à répondre efficacement à ces exigences sans recourir à des méthodes supervisées traditionnelles.

Conclusion

En conclusion, on à vue qu’en partant d'une solution non supervisée initiale imparfaite et, en y apportant quelques améliorations ciblées, d'augmenter à la fois l'efficacité et la rapidité des processus. Concernant la solution supervisée, là où le manque d'une base d'apprentissage semblait être un obstacle insurmontable, nous avons finalement réussi à développer des méthodes alternatives efficaces qui pourront bientôt on l'espère se substituer à cette base.

Nous espérons que l'offre Verbatim continuera de rencontrer le même succès qu'elle connaît actuellement, en attirant toujours plus de clients. Avec Arnaud et Philippe aux commandes, il est certain que cette solution ne pourra que s'améliorer avec le temps. De plus, avec l'essor de l'intelligence artificielle et l'intérêt grandissant de La Poste pour ces technologies, les perspectives d'évolution sont très prometteuses.

Retour d’expérience

Mon expérience en tant que data scientist stagiaire à La Poste a été extrêmement enrichissante et formatrice, dépassant de loin mes attentes. Dès mon arrivée, j'ai été accueilli au sein d'une équipe de professionnels compétents, passionnés, et toujours prêts à partager leurs connaissances. Ce climat bienveillant et stimulant m'a permis de m'intégrer rapidement et de me sentir à l'aise pour poser des questions et échanger sur des idées, ce qui a grandement favorisé mon apprentissage.

L'un des aspects les plus marquants de mon stage a été l'opportunité de plonger dans le domaine de l'intelligence artificielle, en particulier dans le traitement du langage naturel. J'ai pu travailler sur des projets concrets qui m'ont permis d'appliquer des techniques de machine learning et d'IA avancées, telles que les modèles de langage, les embeddings, les transformers, que j'avais jusqu'alors abordés principalement de manière théorique. Ce contact direct avec des problématiques réelles m'a permis de consolider mes compétences en Python, notamment dans la manipulation de grandes quantités de données, l'utilisation de bibliothèques spécialisées en NLP.

J'ai également eu l'occasion d'apporter ma pierre à l'édifice en contribuant de manière significative à l'équipe grâce à mes connaissances en data science, plus particulièrement dans le domaine de la classification. Au cours de mon stage, j'ai pu proposer et développer une nouvelle méthode de classification, qui s'est révélée plus précise et mieux adaptée aux besoins spécifiques de l'entreprise. Cette expérience m'a apporté une grande satisfaction, car elle m'a permis de voir l'impact direct de mon travail au sein de l'équipe et de l'entreprise.

Au-delà des compétences techniques, ce stage m'a également permis de développer une meilleure compréhension du travail en équipe dans un contexte professionnel. Travailler au sein d'une équipe expérimentée m'a non seulement appris l'importance de la collaboration, mais m'a aussi montré comment un projet se déroule du début à la fin dans un environnement professionnel. J'ai pu observer et participer à des processus de prise de décision collective, de gestion de projet, et de résolution de problèmes, ce qui m'a donné une perspective précieuse sur les dynamiques de travail au sein d'une grande entreprise comme La Poste.

ANNEXES

An.1 : Les modèles transformers

Les modèles de langage naturel reposent sur des avancées majeures dans le domaine de l'intelligence artificielle, en particulier les réseaux de neurones et les architectures Transformers.

Les Transformers représentent une avancée cruciale dans les réseaux de neurones, introduite par Vaswani et al. en 2017 dans l'article intitulé Attention Is All You Need. Ce modèle a remplacé les architectures séquentielles comme les RNN (Recurrent Neural Networks) et les LSTM , qui souffraient de problèmes de parallélisation et de difficultés à capturer des dépendances à long terme dans les séquences.

Le Transformer est principalement composé de deux blocs : l'encodeur et le décodeur. L'élément clé du Transformer est le mécanisme d'attention, et plus précisément l'**attention multi-tête**. Ce mécanisme permet au modèle de se concentrer sur différentes parties d'une séquence d'entrée simultanément, en attribuant des poids différents à différentes parties du texte selon leur importance relative.

L'attention dans un Transformer est définie comme suit :

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V$$

où :

Q représente la matrice des requêtes (queries).

Kest la matrice des clés (keys).

V est la matrice des valeurs (values).

d_k est la dimension des clés.

Cette formule montre comment, pour chaque position dans une séquence donnée, le modèle génère une attention pondérée sur toutes les autres positions de la séquence. Le résultat est une combinaison pondérée des valeurs, ce qui permet de capturer les dépendances contextuelles entre différents éléments de la séquence.

Les Transformers sont utilisés pour modéliser la probabilité conjointe d'une séquence de mots. Par exemple, dans un modèle génératif comme GPT, l'objectif est de maximiser la probabilité suivante :

$$P(w_1, w_2, \ldots, w_n) = \prod_{t=1}^{n} P(w_t | w_1, w_2, \ldots, w_{t-1})$$

Ici, chaque mot wt est prédit en fonction de tous les mots précédents dans la séquence. Le mécanisme d'attention permet au modèle de regarder l'intégralité de la séquence passée pour effectuer cette prédiction, capturant ainsi à la fois les dépendances à court et à long terme.

Les Transformers ont permis le développement des premiers LLM capables de générer du texte cohérent à une échelle sans précédent, posant les bases pour des modèles tels que BERT, GPT, qui dominent aujourd'hui le domaine du traitement automatique du langage naturel.

An.2 : Réseau de neurones : cellule LSTM

Les cellules LSTM (Long Short-Term Memory) sont une variante avancée des réseaux de neurones récurrents(RNN). Elles ont été conçues pour résoudre les problèmes liés à la capture des dépendances à long terme dans les séquences de données, une difficulté rencontrée dans les architectures RNN. Les LSTM sont particulièrement adaptées aux tâches où les données d'entrée sont séquentielles, comme dans le NLP.

Les LSTM, introduites par Hochreiter et Schmidhuber en 1997, offrent une solution à ces problèmes en intégrant une mémoire de long terme explicitement gérée par une série de portes. Contrairement aux RNN classiques, où chaque cellule a une simple fonction d'activation, les LSTM ont une structure interne plus complexe, avec **plusieurs portes qui régulent le flux d'informations**.

Une cellule LSTM est composée de trois portes principales :

- Porte d'entrée : Elle décide de la quantité d'informations provenant de l'entrée actuelle à ajouter à l'état de la mémoire.
- Porte d'oubli : Elle contrôle la quantité d'informations à conserver ou à oublier de l'état précédent.
- Porte de sortie : Elle détermine quelle partie de l'état de la mémoire doit être utilisée pour produire la sortie.

Les formules qui décrivent ces portes sont les suivantes :

Porte d'oubli :

$$f_t = \sigma(W_f \cdot [h_{t-1}, x_t] + b_f)$$

où :

- f_t est le vecteur de la porte d'oubli à l'instant t.
- W_f est la matrice de poids associée à la porte d'oubli.
- h_{t-1} est l'état caché à l'instant précédent.
- x_t est l'entrée à l'instant actuel.
- b_f est le biais de la porte d'oubli.
- σ est la fonction sigmoïde.

Porte d'entrée :

$$i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i)$$

$$\tilde{C}_t = \tanh(W_C \cdot [h_{t-1}, x_t] + b_C)$$

où :

- i_t est le vecteur de la porte d'entrée à l'instant t.
- W_i est la matrice de poids associée à la porte d'entrée.
- $\tilde{C}_t$ est le vecteur de l'état de la mémoire candidat à l'instant t.
- W_C est la matrice de poids associée à l'état de la mémoire candidat.

Mise à jour de l'état de la mémoire :

$$C_t = f_t \times C_{t-1} + i_t \times \tilde{C}_t$$

où C_t est le nouvel état de la mémoire à l'instant t.

Porte de sortie :

$$o_t = \sigma(W_o \cdot [h_{t-1}, x_t] + b_o)$$

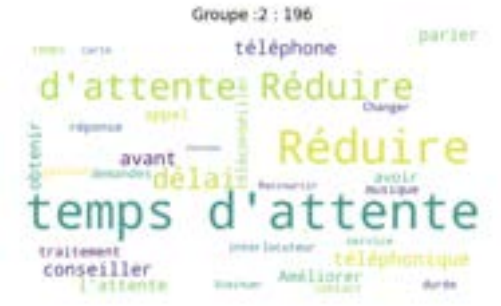
$$h_t = o_t \times \tanh(C_t)$$

où :

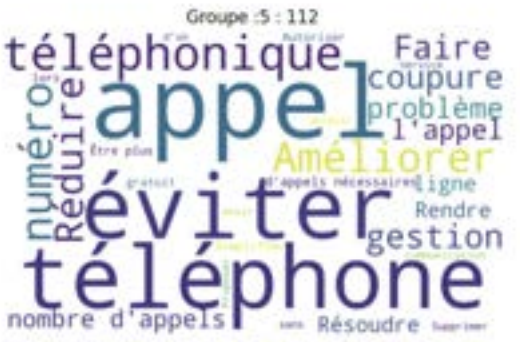
- o_t est le vecteur de la porte de sortie à l'instant t.
- h_t est l'état caché (ou la sortie) à l'instant t.

An.3 : WordCloud verbatim LBP - Améliorations

- Word-clouds des groupes forts :



- Word-clouds des groupes faibles :




Groupe :0 : 89
l'appel
Avoir
client
Rappeler
Éviter
interlocuteur
téléphonique
conversation
communication
téléphone
demande
personne
prendre
Pouvoir
après
service
cas
appel
lorsque
Assurer
Faire
avant


Groupe :1 : 30
éviter
virement
d'un
Envoyer
chèque
compte
bancaire
clients
Informer
Autoriser
livret A
relevés
vers
Permettre
opposition
service
cas
SMS
exemple
refus
Internet
chéquier


Groupe :2 : 61
bancaire
service
Banque
Postale
Changer
Améliorer
Banque
conseiller
Mettre
Meilleure
entre
ligne
clients
démarches
jour
compte
eCB
opérations
numéro
site
agence
virement
Faciliter
comme
système
sans
d'un
place
délais
Pouvoir