MATHÉMATIQUES ET INFORMATIQUE
Sciences
Université de Paris

Master 2 Informatique
Rapport de projet

Apprentissage profond pour données textuelles

David EL RAIS

Année universitaire : 2020 – 2021

Encadrant : M. Lazhar LABIOD – M. Mohamed NADIF

RÉSUMÉ

L'apprentissage profond a permis des progrès importants et rapides dans plusieurs domaines de l'apprentissage automatique.

En raison de ses applications réussies dans les tâches de classement au cours de la dernière décennie, les chercheurs ont tenté d'appliquer le même paradigme à un domaine d'apprentissage non supervisé, en particulier pour les problèmes de clustering.

En effet, habituellement les data scientists ont tendances à utiliser séquentiellement une méthode de réduction de dimension pour ensuite appliquer un algorithme de clustering.

Cependant de récentes études ont montré que l'utilisation combinée d'une méthode de réduction de la dimension et d'un algorithme de clustering permettait d'obtenir des meilleures performances.

Ce projet se base sur l'algorithme DCN, l'algorithme peut être défini comme une procédure recherchant simultanément une nouvelle représentation des données contenant le maximum d'information en utilisant un réseau neuronal (**auto-encodeur**) et un algorithme de clustering (**K-Means**).

Nous allons dans un premier temps étudier les performances des méthodes séquentielles ce qui nous permettra de mieux comprendre l'apport d'une méthode DCN.

Dans un second temps, nous étudierons l'intérêt et l'impact de la version régularisée de DCN.

Ce rapport retrace les travaux et les choix effectués durant ce projet.

TABLE DES MATIÈRES

LISTE DES FIGURES

LISTE DES TABLES

INTRODUCTION

1. Introduction

1.1 Contexte et Motivations

Dans le but de comprendre l'intérêt d'utiliser l'algorithme DCN, c'est-à-dire une approche combinant simultanément un auto-encoder pour réduire la dimension et l'utilisation de K-Means pour déterminer les classes, nous devons définir d'un point de vue scientifique l'état de l'art en matière d'apprentissage non supervisé notamment concernant la symbiose entre les méthodes de réduction de dimension et les méthodes de clustering.

En Machine Learning, les méthodes de clustering consistent à partitionner un ensemble d'objets décrits par un ensemble de variables en classes homogènes. Cependant, les méthodes conventionnelles présentent parfois des difficultés compte tenu de la complexité des données.

En effet, certaines méthodes de clustering imposent une organisation des données selon un critère interne.

Par exemple, concernant les conditions internes de l'algorithme K-Means, nous supposons que les individus suivent une distribution normale et ont la même probabilité d'apparaître dans chaque cluster, ce qui signifie que chaque classe a la même proportion d'individus, donc K-Means retourne des classes qui ont le même nombre d'individus (mêmes proportions).

De plus, la matrice de variance-covariance est diagonale, elle est sous forme $\boldsymbol{\lambda} \cdot \boldsymbol{Id}$, donc K-Means est plus adapté aux clusters sphériques plutôt qu'aux clusters allongés.

L'une des méthodes qui a fait ses preuves et qui est l'une des plus répandues pour améliorer les performances de clustering consiste à utiliser une méthode de réduction de dimension des données afin d'obtenir une nouvelle représentation des données qui serait profitable pour le clustering.

1.2 Organisation du rapport

Le rapport est divisé en 3 sections.

La première section décrit les différents jeux de données et les prétraitements associés.

La seconde section est une étude comparative entre DCN et les autres méthodes de clustering sur des données textuelles telles que NMF et Spherical K-Means.

La troisième section concerne l'implémentation de la variante DCN-régularisé.

STATISTIQUE DESCRIPTIVE

2. Statistique Descriptive

2.1 Spam

Le jeu de données **spam** est un ensemble de messages qui ont été collectés pour la recherche de spam par SMS.

Il contient un ensemble de messages SMS en anglais de 5574 messages, étiquetés comme étant **ham** (légitime) ou **spam**.

Les fichiers contiennent un message par ligne, chaque ligne est composée de deux colonnes : v1 contient le libellé (ham ou spam) et v2 contient le texte brut.

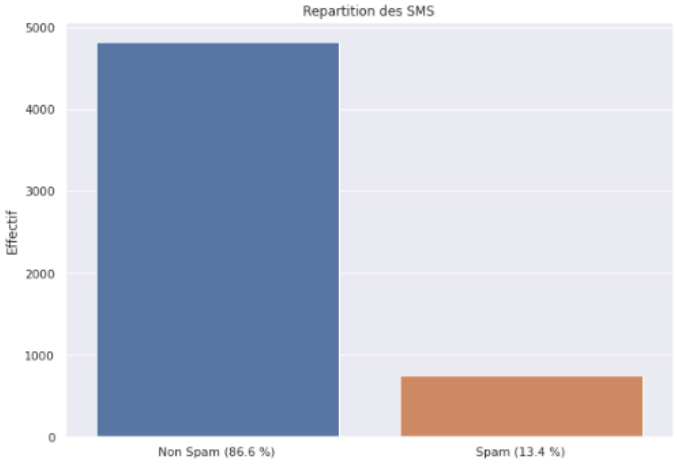

Figure 1. Répartitions des SMS

En étudiant le jeu de donnée, on s'aperçoit que les SMS ne sont pas équitablement distribués, en effet on a **86,6%** de sms légitime et **13,4%** de spam, cette distribution n'influence pas les résultats du clustering car on s'intéresse seulement au positionnement des individus dans un espace mais permettra de vérifier si les algorithmes de clustering préservent la proportion des classes.

Une des caractéristiques intéressantes à étudier est la taille des SMS, en effet elle peut être considérée comme une caractéristique discriminante si on suppose que les sms ont une certaine structure, ce qui est souvent le cas.

La plupart des spams sont envoyés à une multitude de (potentiels) victimes et doivent donc adopter une certaine forme pour faire le plus de victimes.

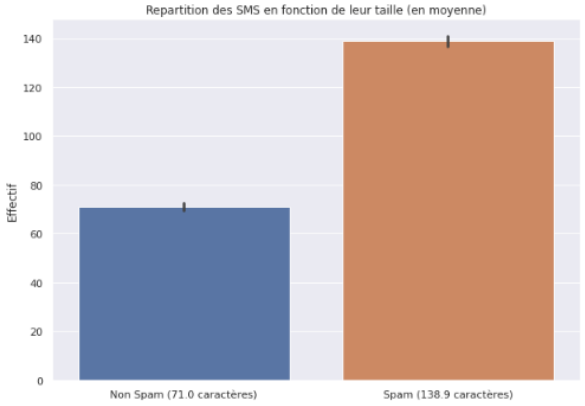

Figure 2. Répartition des SMS en fonction de leur taille (en moyenne)

On s'aperçoit que les spams ont en moyenne le double de la taille des sms non spams, on peut approfondir cette piste en visualisant la distribution des spams et des non spams.

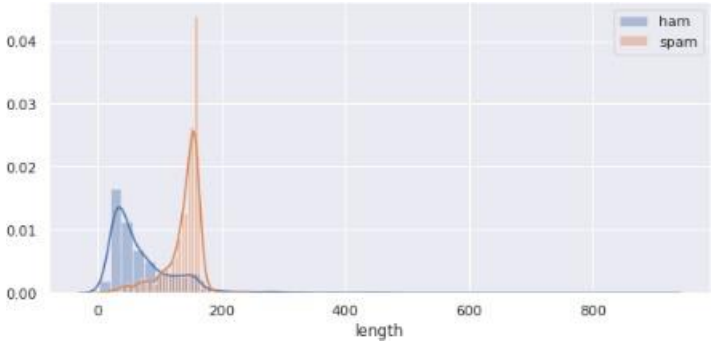

Figure 3. Distribution des données

En étudiant ce graphe, on remarque que les fonctions sont assez bien séparées, cependant on peut noter la présence d'un chevauchement, ce qui laisse supposer que les deux classes sont assez bien séparées avec quelque faux positifs et faux négatifs.

Il est donc nécessaire de trouver un nouvel espace qui permettra de séparer au mieux les deux classes afin d'avoir un espace profitable pour le clustering.

Pour le reste du rapport, sur les figures, les points jaunes correspondent à des spam et les points bleus correspondent à des non spams.

2.2 20NewsGroup

Le jeu de données **20NewsGroup** est un ensemble de groupe de discussion contenant environ 18 000 articles sur 20 sujets répartis en deux sous-ensembles : l'un pour la formation (ou le développement) et l'autre pour les tests (ou pour l'évaluation des performances).

Dans ce dataset, nous avons a priori l'information sur le nombre de clusters (catégorie) qui est utilisé pour évaluer nos méthodes de classification.

Les 20 catégories sont :

- alt.atheism
- comp.graphics
- comp.os.ms-windows.misc
- comp.sys.ibm.pc.hardware
- comp.sys.mac.hardware
- comp.windows.x
- misc.for sale
- rec.autos
- rec.motorcycles
- rec.sport.baseball
- rec.sport.hockey
- sci.crypt
- sci.electronics
- sci.med
- sci.space'
- soc.religion.christian
- talk.politics.guns
- talk.politics.mideast
- talk.politics.misc
- talk.religion.misc

On remarque que les articles traitent de sujets différents, parmi eux : la religion, le sport, l'informatique, l'industrie, la politique.

De plus, nous pouvons aller plus loin en faisant du topic modeling pour regrouper les articles par thème (sport, religion, politique, …) mais ceci n'est pas nécessaire pour réaliser notre objectif de clustering.

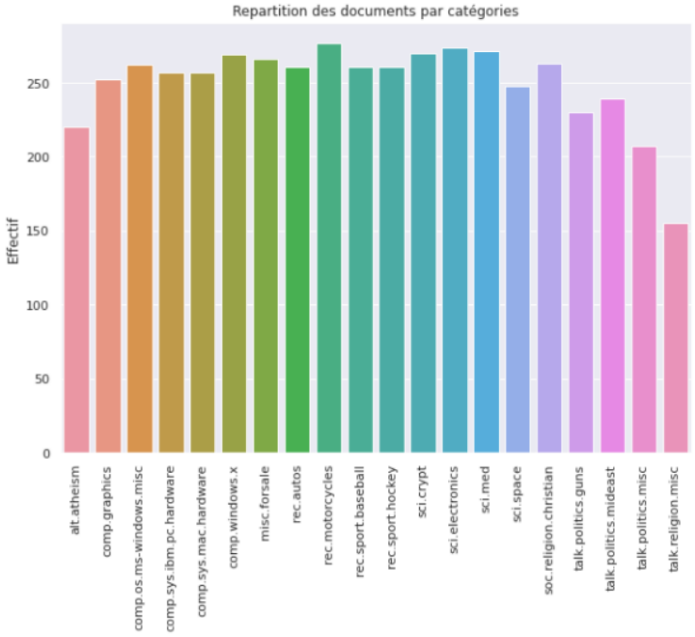

Figure 4. Répartition des documents par catégories

Contrairement au dataset précèdent, on remarque que le nombre d'articles dans chaque catégorie est presque équilibré, on a en moyenne 250 articles par catégorie.

On peut donc supposer que les clusters devront être homogènes.

Comme précédemment, nous allons étudier le nombre de caractères par catégories, ce qui nous donnera une idée des répartitions par catégories.

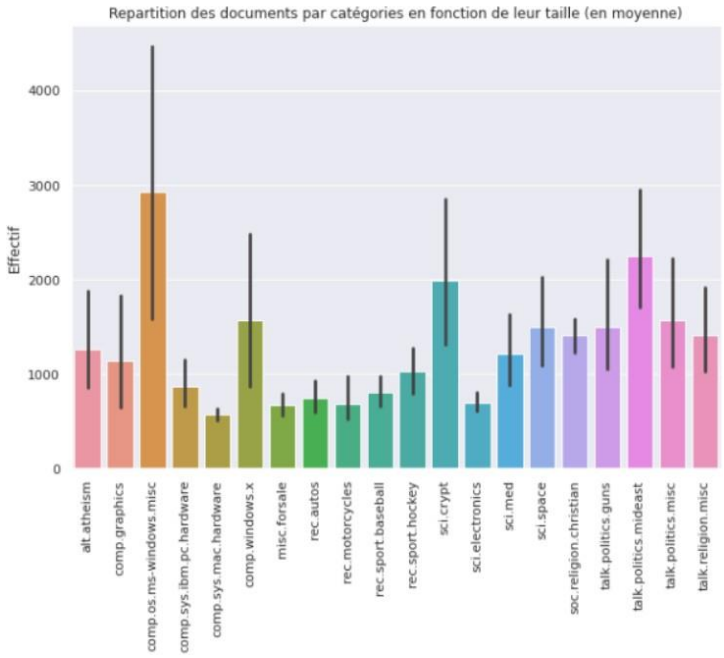

Figure 5. Répartition des documents par catégories en fonction de leur taille (en moyenne)

On remarque que le nombre de caractères par catégorie est inégal entre documents, on peut classer les documents en trois catégories : en dessous de 1200, entre 1200 et 1700 et au-dessus de 1700, on peut émettre l'hypothèse qu'on a un problème complexe où la visualisation des classes sera difficile à observer.

Pour appuyer cette hypothèse, nous pouvons étudier les distributions de chaque catégorie.

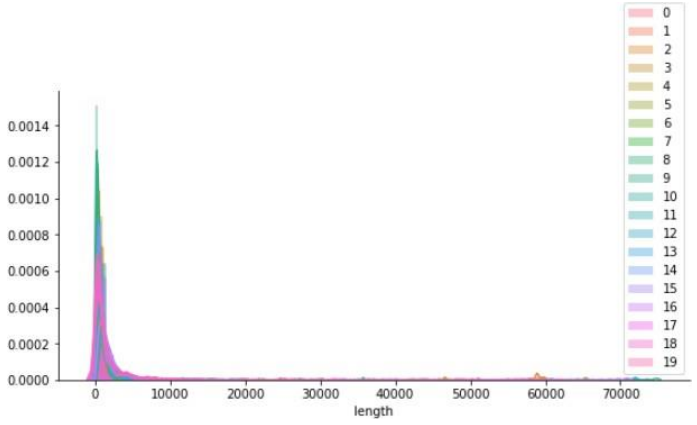

Figure 6. Distribution des données

En étudiant ce graphe, on remarque que les fonctions ne sont pas bien séparées, on peut donc s'attendre à la présence de plusieurs faux positifs et faux négatifs.

Il est donc nécessaire de trouver un nouvel espace qui permettra de séparer au mieux les 20 classes afin d'avoir un espace profitable pour le clustering.

MATRICE DOCUMENT - TERME

3. Pré-Traitement et Matrice Document-Terme

Les deux jeux de données sont des données textuelles, le prétraitement ainsi que la création des matrices document-terme et TF-IDF sont similaires nous allons donc faire une synthèse des deux jeux de données.

Donc pour chaque dataset nous allons avoir deux matrices document-terme (**Count** et **TF-IDF**), ce qui nous permettra de comparer les performances des méthodes de clustering sur ces deux matrices.

3.1 Pré-Traitement

Avant la création des matrices, nous devons obligatoirement passer par une étape de pré-traitement qui se décompose en différentes étapes :

- Suppression des caractères numériques
- Mise en minuscule
- Suppression des stops words
- Lemmatisation des termes
- Suppression des mots de taille inférieur à 2 caractères.

3.2 Matrice Document-Terme

La matrice document-terme est une matrice qui contient le nombre de fois où un mot apparaît dans un article, cependant cette représentation des données n'est pas la plus profitable.

En effet, il existe des moyens d'ajuster la fréquence, en fonction de la longueur d'un document ou de la fréquence brute du mot le plus fréquent dans un article.

Au niveau du code, une fois le prétraitement effectué, on utilise la fonction **CountVectorizer**, du package **Sklearn** qui permet de convertir une collection de document en matrice sparse '**token counts**', avec un max_features égale à **1500**.

3.3 Matrice TF-IDF

TF-IDF est une mesure statistique qui évalue la pertinence et l'importance d'un mot pour un article en fonction des autres documents. Cette mesure se résume en multipliant deux matrices : **Term Frequency** et **Inverse Document Frequency.**

La matrice Terme Fréquence correspond à la matrice Document-Terme décrite ci-dessus.

Concernant la Fréquence inverse des documents, cette matrice contient l'information sur la fréquence ou la rareté d'un mot dans l'ensemble des articles.

La normalisation TF-IDF permet de répondre à la question : " Est-ce que le mot apparaît dans tous les articles ? "

S'il est proche de 0 c'est-à-dire que le mot est courant (il apparaît dans plusieurs articles).

On peut ainsi calculer cette mesure par la division du nombre total d'articles par le nombre d'articles contenant un mot tout en utilisant le logarithme (si le mot est présent dans tous les articles alors la division du nombre total d'articles par le nombre total d'articles contenant le mot sera égale à 1, en utilisant le logarithme on a log (1) = 0, ce qui permet d'affecter un score faible aux mots les plus courant).

La multiplication des deux matrices implique un score TF-IDF d'un mot dans un article sur l'ensemble des articles.

Donc, si le mot est très courant et qu'il apparaît dans de nombreux articles, ce score se rapprochera de 0 sinon il se rapprochera de 1.

Au niveau code, pour obtenir une matrice TF-IDF nous utilisons la fonction **TfidfVectorizer**, du package **Sklearn** avec un max_features égale à **1500**.

RÉDUCTION DE LA DIMENSION

4. Réduction de la dimension

La réduction de la dimension consiste à prendre des données dans un espace de grande dimension et à les remplacer par des données dans un espace de plus petite dimension tout en préservant au maximum la structure des données.

On a utilisé deux méthodes de réduction de la dimension, la première étant **l'analyse des correspondances** qui est une méthode linéaire adaptée aux données textuelles. La seconde méthode est **t-SNE**, c'est une méthode non linéaire qui permettra de comparer l'état de l'art entre les méthodes linéaire et non linéaire.

4.1 AC

On utilise l'analyse des correspondances (AC), car c'est une méthode qui permet de décrire et de visualiser des tableaux de contingence et donc d'étudier les relations entre les modalités de 2 variables qualitatives.

La méthode est classiquement utilisée en analyse textuelle.

De plus, le tableau de données est considéré comme une table de contingence car la somme des lignes a un sens (nombre des termes dans un document) et la somme des colonnes à un sens (nombre de fois qu'un terme apparaît dans tous les documents) de plus les données sont positives, donc on tient compte de la métrique du **Khi-Deux**.

C'est très important dans ce type de matrice, car lorsqu'on dit qu'un document appartient à la même classe qu'un autre document (un résumé de 10 lignes et un document de 20 pages qui traite du même sujet), lors d'une classification, on veut que ces 2 documents soient dans la même classe, la métrique du Khi-Deux va jouer ce rôle car elle va normaliser les documents.

4.1.1 Spam

Afin d'avoir un aperçu de la réduction de dimension sur le dataset spam, nous allons prendre un échantillon de seulement 50 documents ce qui nous permettra d'alléger nos graphes et d'améliorer l'interprétation de nos résultats. De plus, nous allons présenter une version avec les 1500 features afin d'avoir un aperçu des données.

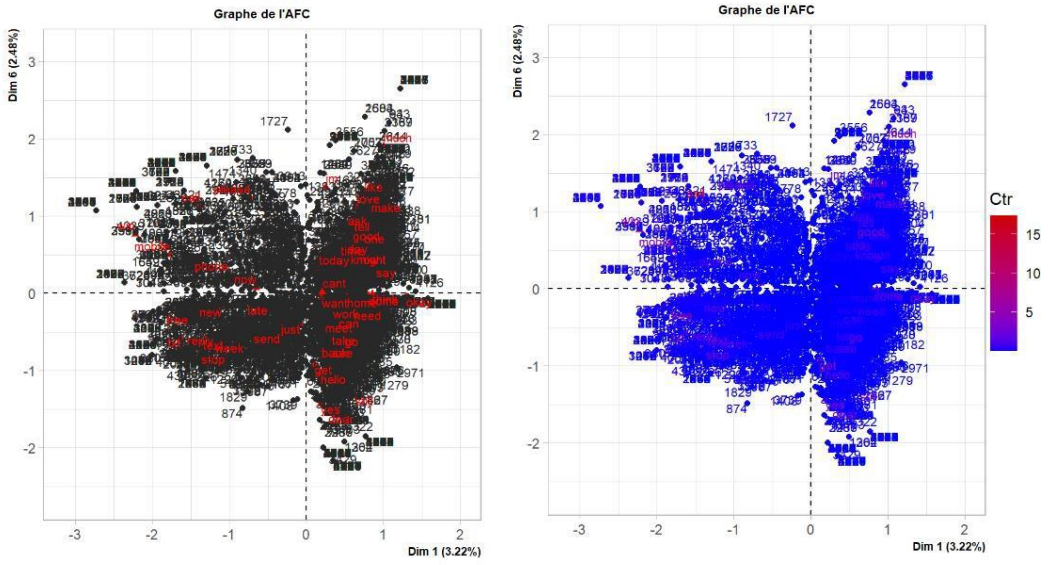

Figure 7. Visualisation sur les axes factoriels 1 et 6 sans contribution (à gauche) et avec contribution (à droite)

Nous avons choisi les axes factoriels 1 et 6 afin d'avoir une visualisation en classe, d'après ce graphique on remarque la présence de 4 clusters.

Nous avons affiché les deux graphes avec et sans la contribution, afin d'avoir un aperçu des termes sur le plan factoriel.

La première classe (en haut à gauche) est caractérisée par le champ lexical de la téléphonie, on remarque la présence des termes 'win' et 'please**'** ce qui laisse penser que ces types de documents font références à des spams qui demandent d'appeler pour gagner quelque chose (catfish, scam téléphonique, …).

La seconde classe (en haut à droite) est caractérisée par des mots communs par exemple : 'ask', 'today', 'day', 'think', etc.

La troisième classe (en bas à gauche) est caractérisée par un champ lexical qui rappelle celui du spam, en effet on remarque la présence des termes 'free', 'stop', 'reply', 'send', etc.

On peut donc supposer que ces documents font références à des spams qui demandent un retour (reply, send) pour obtenir quelque chose (free).

La quatrième classe (en bas à droite) est caractérisée par des mots communs par exemple : yes, go, home, meet, can, etc.

On peut approfondir l'étude sur les clusters, en étudiant les différents axes factoriels, on remarque que le premier axe factoriel est caractérisé par des termes faisant référence aux termes utiliser dans les spams, tandis que le second axe factoriel est caractérisé par des termes communs, des termes utilisés par l'ensemble des SMS.

Ceci peut aussi être vu en analysant la contribution, en effet, les termes qui contribuent le plus au premier axe factoriel sont 'win', 'mobile' et 'okay' tandis que les termes qui contribuent le plus au second axe factoriel sont 'dear'.

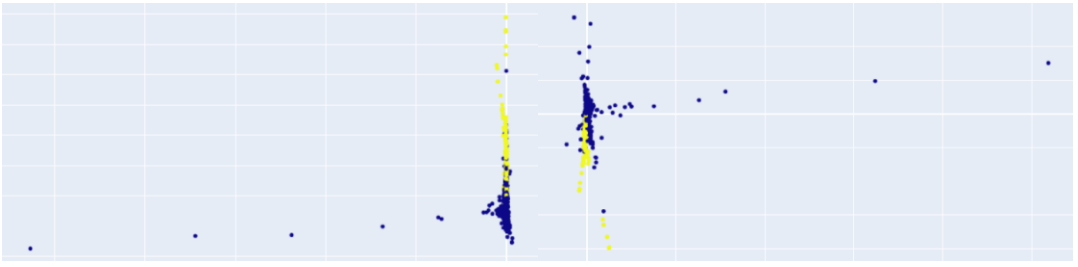

Figure 8. Matrice document-terme (à gauche) et matrice TF-IDF (à droite)

Si on applique le même raisonnement sur l'ensemble des 1500 termes on s'aperçoit qu'on a quasiment les mêmes représentations, avec des classes qui sont assez bien séparés mais on note la présence de quelques chevauchements, ce qui fait sens à ce qui a était dit dans la partie 2.1 Statistique Descriptive – Spam.

4.1.2 20NewsGroup

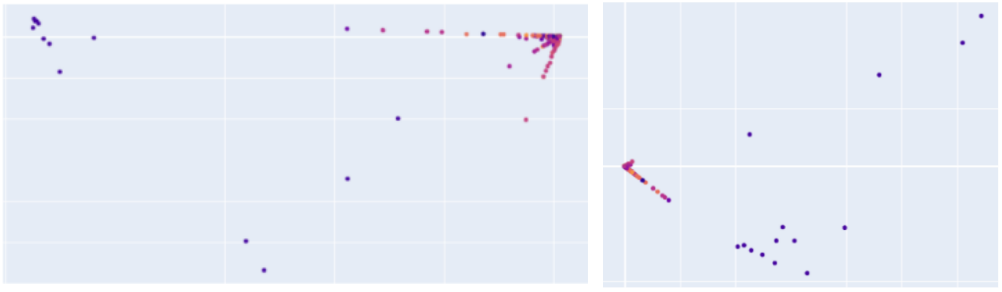

Figure 9. Matrice document-terme (à gauche) et matrice TF-IDF (à droite)

On remarque qu'on a quasiment les mêmes résultats entre les deux types de matrices ce qui suppose qu'on est dans un problème compliqué et que les clusters ne sont pas linéairement séparables.

On peut donc supposer que t-SNE permettra de mettre plus en valeur les clusters et améliora les performances de clustering.

4.2 T-SNE

Le t désigne une distribution utilisée dans l'algorithme (loi t de Student), c'est une méthode de réduction de la dimension des données **non-linéaires** afin de **repérer des structures locales**.

Le concept général de cette méthode est de considérer chaque point de données séparément, et d'assigner une probabilité conditionnelle (**un poids**) gaussien à chacun des autres points en fonction de leur distance par rapport à ce point.

Une fois que ces probabilités sont calculées on peut réduire la dimension des données en respectant la distribution des données d'origine.

4.2.1 Spam

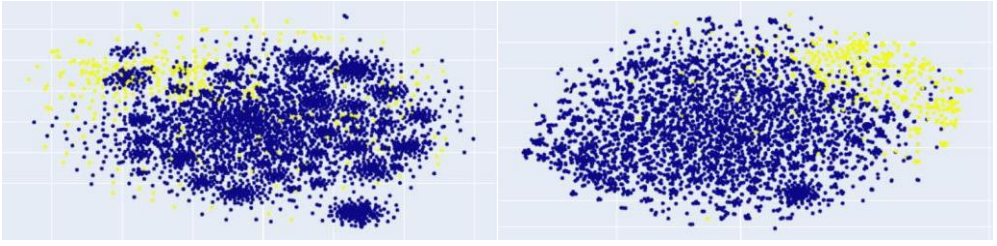

Figure 10. Matrice document-terme (à gauche) et matrice TF-IDF (à droite)

Contrairement à l'analyse des correspondances, on voit bien l'effet de la matrice TF-IDF dans les résultats, ceci peut être expliqué par le fait que t-SNE est une méthode non linéaire on obtient donc une meilleure séparation des données.

4.2.2 20NewsGroup

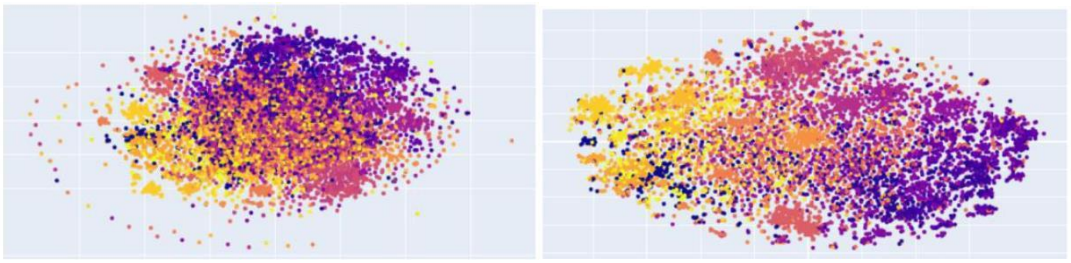

Figure 11. Matrice document-terme (à gauche) et matrice TF-IDF (à droite)

En observant les figures, on remarque la présence de plusieurs clusters sur la matrice TF-IDF, ce qui vérifie l'hypothèse émise dans la partie 4.1.2 Reduction de la dimension – 20NewsGroup.

ALGORITHME DE CLUSTERING

5. Algorithme de Clustering

Le but de cette partie est d'étudier la symbiose entre les méthodes de réduction de la dimension et les algorithmes de clustering à travers deux algorithmes de clustering (**Spherical Kmeans** et **NMF)**, deux méthodes de réduction de dimension (**AC** et **t-SNE**), deux datasets (**Spam** et **20NewsGroup**) et deux représentations des matrice documents termes (**Count Vectorizer** et **Tf-Idf Vectorizer**).

5.1 Spherical K-Means

L'intuition et l'idée générale est la même que l'algorithme K-Means, la différence est la métrique à utiliser pour calculer la distance pour séparer et classifier les points au cluster le plus proches.

K-Means utilise la distance euclidien et Spherical K-Means opte pour la distance cosinus pour séparer les points.

5.1.1 Spam

5.1.1.1 Sans méthode de réduction de la dimension

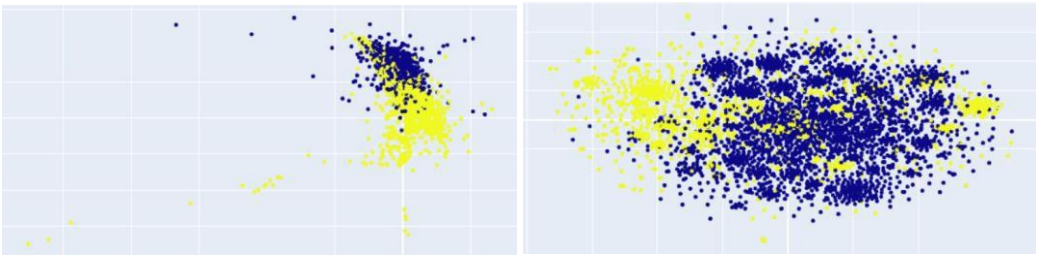

Figure 12. Résultat de Spherical Kmeans sur la matrice document-termes avec projection des résultats sur AC (à gauche) et t-SNE (à droite).

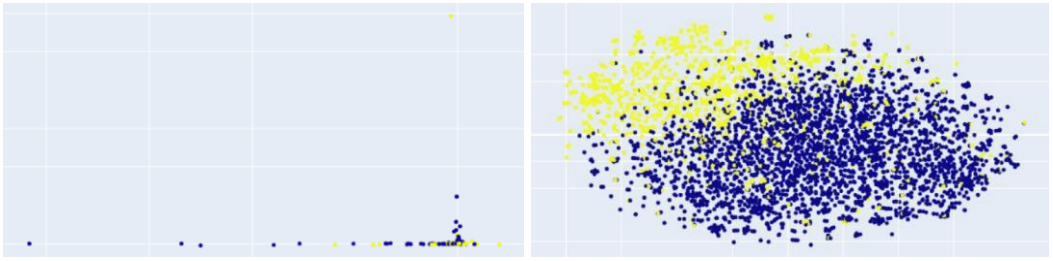

Figure 13. Résultat de Spherical Kmeans sur la matrice TF-IDF avec projection des résultats sur AC (à gauche) et t-SNE (à droite).

On remarque que l'application de Spherical K-Means sur le jeu de donnée spam sans réduction de la dimension n'est pas similaire entre les deux types de matrice document-terme et les deux types de la réduction de dimension.

En effet, l'apport de TF-IDF permet une meilleure séparation entre les classes et permet d'éviter les faux positifs et faux négatifs, pour rappel seulement 13.8% des sms sont classés spam au niveau de la version matrice document-terme on peut voir environ 40% de spam.

De plus, la version t-SNE permet de très bien séparer les classes car c'est une méthode non linéaire.

5.1.1.2 Avec méthode de réduction de la dimension

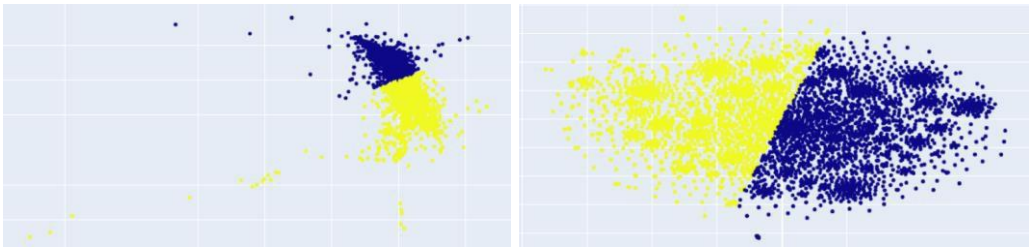

Figure 14. Résultat de Spherical Kmeans sur la matrice document-termes avec réduction de la dimension et projection des résultats sur AC (à gauche) et réduction de la dimension et projection des résultats sur t-SNE (à droite).

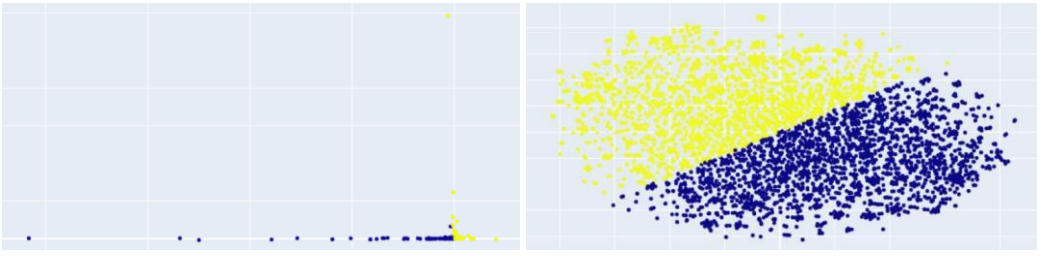

Figure 15. Résultat de Spherical Kmeans sur la matrice TF-IDF avec réduction de la dimension et projection des résultats sur AC (à gauche) et réduction de la dimension et projection des résultats sur t-SNE (à droite).

La réduction de la dimension permet une meilleure représentation des classes sur les deux types de matrices.

Cependant les résultats de la matrice TF-IDF avec l'analyse factorielle permet de préserver l'effectif de chaque cluster, on préserve les 13.8% de spam.

5.1.2 20NewsGroup

5.1.2.1 Sans méthode de réduction de la dimension

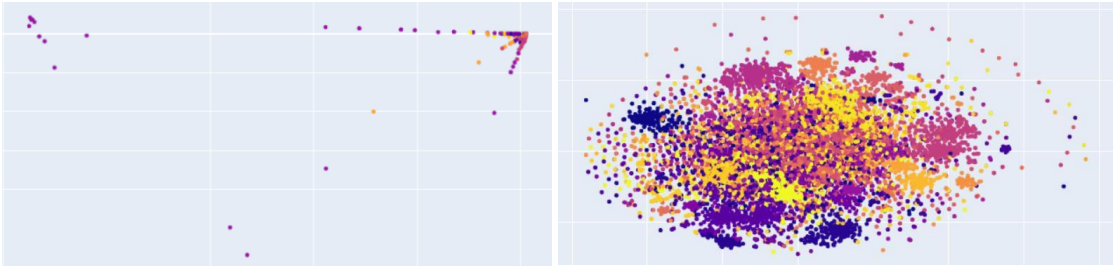

Figure 16. Résultat de Spherical Kmeans sur la matrice document-termes avec projection des résultats sur AC (à gauche) et t-SNE (à droite).

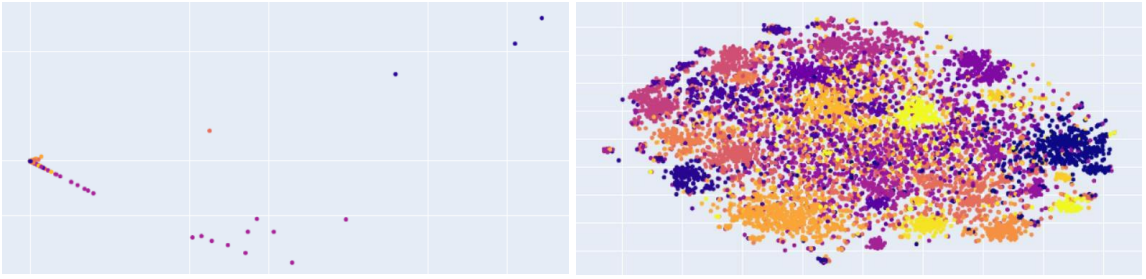

Figure 17. Résultat de Spherical Kmeans sur la matrice TF-IDF avec projection des résultats sur AC (à gauche) et t-SNE (à droite).

On ne remarque pas de réelle différence entre les deux types de matrice ceci fait écho à ce qui a était dit dans la partie (2.2 Statistique Descriptive – 20NewsGroup), en effet on est dans un problème compliqué ou les classes se chevauche, il est donc nécessaire de passer par une méthode de réduction de dimension non linéaire afin de capter la structure des classes et préserver le maximum d'information utile au clustering.

5.1.2.2 Avec méthode de réduction de la dimension

Figure 18. Résultat de Spherical K-Means sur la matrice document-termes avec réduction de la dimension et projection des résultats sur AC (à gauche) et réduction de la dimension et projection des résultats sur t-SNE (à droite).

Figure 19. Résultat de Spherical K-Means sur la matrice TF-IDF avec réduction de la dimension et projection des résultats sur AC (à gauche) et réduction de la dimension et projection des résultats sur t-SNE (à droite).

On peut vérifier l'hypothèse émise précédemment, t-SNE (méthode non linéaire), a permis une meilleure représentation des classes, en effet on remarque la présence des 20 classes comparée à l'analyse factorielle qui est une méthode linéaire.

Concernant la matrice TF-IDF, on constate une meilleure homogénéité des classes, en effet l'ensemble des documents d'une classe peut être assimilés à un triangle isocèle.

5.2 NMF

La NMF est une autre méthode de réduction de dimension adaptée aux données positives, par exemple des occurrences de mot (données textuelles).

Le principe de la factorisation X=UV d'une matrice est utilisé dans différentes méthodes comme l'ACP qui utilise la SVD pour construire les facteurs orthogonaux.

Mais avec la NMF il n'y aura pas de contrainte d'orthogonalité, par conséquent les matrices des facteurs sont positives (pour un but unique simplifier l'interprétation).

On ne peut pas utiliser NMF avec une méthode de réduction de la dimension car les valeurs des composantes principales sont positives et négatives.

5.2.1 . Spam

5.2.1.1 Sans méthode de réduction de la dimension

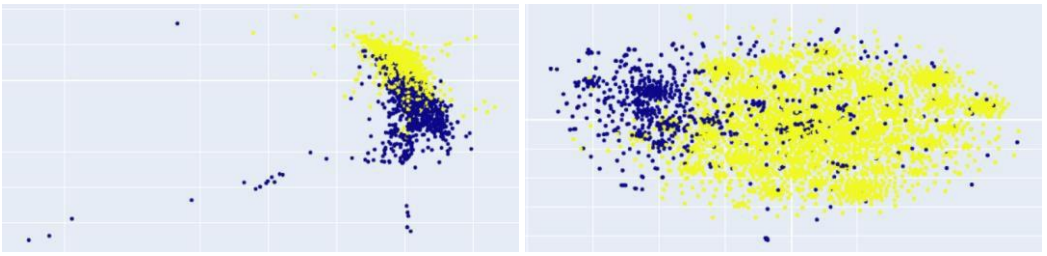

Figure 20. Résultat de NMF sur la matrice documents-termes avec projection des résultats sur AC (à gauche) et t-SNE (à droite).

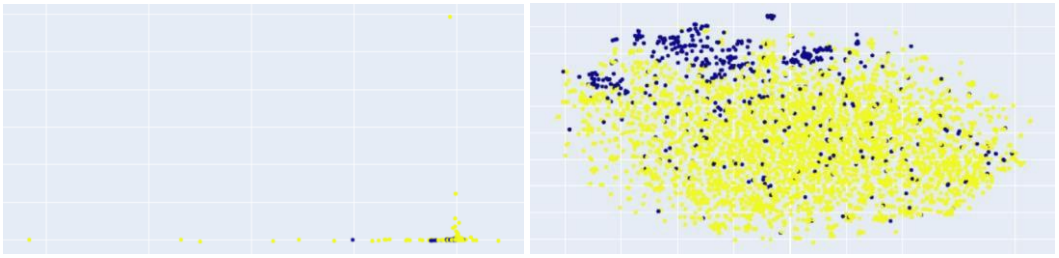

Figure 21. Résultat de NMF sur la matrice TF-IDF avec projection des résultats sur AC (à gauche) et t-SNE (à droite).

Les résultats sont similaires à l'algorithme Spherical K-Means (sans la réduction de dimension), on constate une meilleure représentation des données en utilisant la matrice TF-IDF et t-SNE permet de très bien séparer les classes.

5.2.2 20NewsGroup

5.2.2.1 Sans méthode de réduction de la dimension

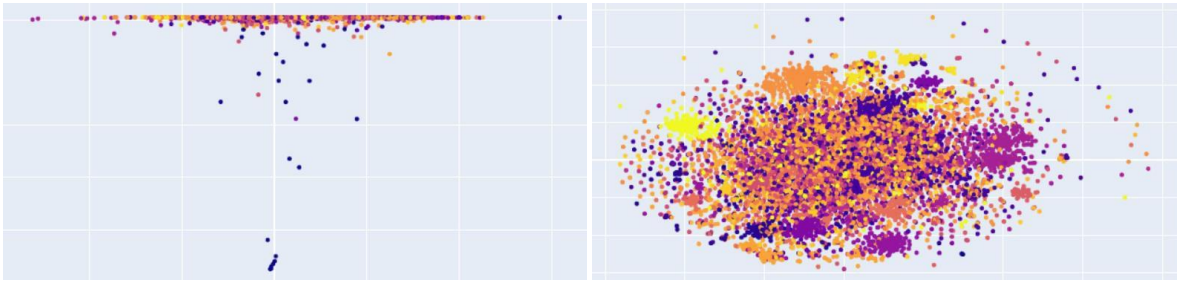

Figure 22. Résultat de NMF sur la matrice documents-termes avec projection des résultats sur AC (à gauche) et t-SNE (à droite)

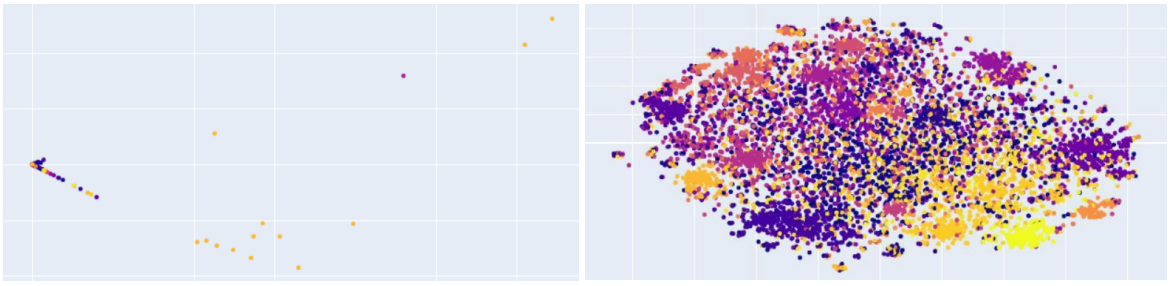

Figure 23. Résultat de NMF sur la matrice TF-IDF avec projection des résultats sur AC (à gauche) et t-SNE (à droite)

Comme précédemment, les résultats sont similaires à l'algorithme Spherical K-Means (sans la réduction de dimension), on ne remarque pas de réelle différence entre les deux types de matrice document-terme au niveau de la séparation des classes, il faudra passer par une méthode de réduction de dimension non linéaire afin de capturer la structure des classes et préserver le maximum d'information profitable au clustering.

5.3 Conclusion

Pour conclure sur l'état de l'art, on constate que les méthodes de clustering couplé aux méthodes de réduction de dimension permettent d'obtenir les meilleurs résultats en termes de performance et de visualisation.

De plus, la matrice TF-IDF permet une représentation plus fidèle des données car elle permet de préserver la structure et les proportions des classes.

AUTO-ENCODER + K-MEANS

6. Auto-encodeur + K-Means

6.1 Définition d'un auto-encodeur

Un auto-encodeur est une généralisation de l'ACP (non linéaire) il est décomposé en deux parties : encodeur et décodeur. Le décodeur compresse le data initial, et le décodeur reconstruit approximativement le data initial à partir de la version compressée obtenue de l'encodeur.

L'auto-encodeur est un modèle de réseaux de neurones, souvent utilisé dans l'apprentissage non supervisé. Il nous donne une autre représentation approximative des données initiales.

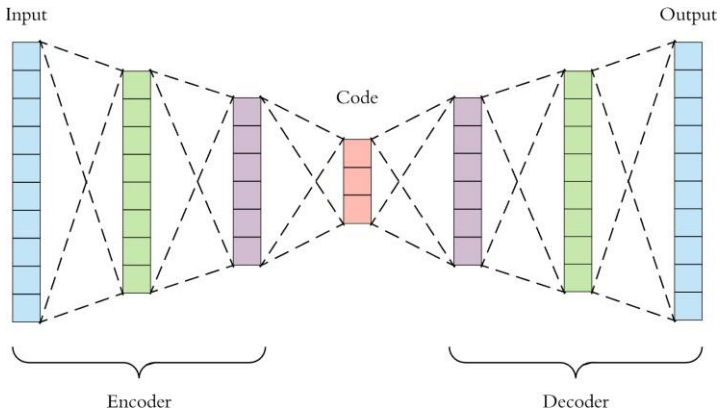

Figure 24. Fonctionnement d'un auto-encoder.

L'encodeur est constitué d'un ensemble de couches de neurones, qui traitent les données pour construire de nouvelles représentations dites encodées (espace de faible dimension).

Le décodeur prend la sortie du codeur (la couche Bottleneck) et tente de recréer une approximation de l'entrée.

Les différences entre les données reconstruites et les données d'origine sont une mesure de l'erreur commise par l'auto-encodeur.

L'entraînement consiste à modifier les paramètres de l'auto-encodeur afin de réduire l'erreur de reconstruction mesurée sur les différents exemples du jeu de données.

La plupart du temps, on ne s'intéresse pas à la couche finale du décodeur, qui contient uniquement la reconstruction des données d'origine, mais plutôt la nouvelle représentation créée par l'encodeur.

Une des difficultés d'un modèle deep learning est de définir le nombre de couche, ect.

6.2 Définition de K-Means

K-Means est une méthode de partitionnement de données et un problème d'optimisation combinatoire.

Étant donné les points et un entier k, le problème est de diviser les points en k groupes, souvent appelés clusters, pour minimiser une certaine fonction. On considère la distance d'un point à la moyenne des points de son centre, la fonction à minimiser est la somme des carrés de ces distances.

Concernant les conditions internes de la méthode, lors de l'utilisation de l'algorithme K-Means, nous supposons que les individus suivent une distribution normale et ont la même probabilité d'apparaître dans chaque cluster, ce qui signifie que chaque classe a la même proportion d'individus, donc K-Means nous donne des classes qui ont le même nombre d'individus (mêmes proportions).

De plus, la matrice de variance-covariance est diagonale, et elle est sous forme $\boldsymbol{\lambda} \cdot \boldsymbol{Id}$, donc K-Means est adapté aux clusters sphériques plutôt qu'aux clusters allongés.

Parmi les faiblesses, nous pouvons citer les labels des individus, en effet nous ne connaissons jamais le cluster réel, nous pouvons aussi citer la faiblesse liée à la moyenne arithmétique, K-Means n'est pas robuste aux valeurs aberrantes, les données très éloignées du centroïde et peuvent donc influencer sur la position du centroïde est donc sur les performances du clustering.

En termes de points forts, nous pouvons parler de la vitesse d'exécution, de la mise à l'échelle de grands ensembles de données, des garanties de convergence.

6.3 Performance d'un auto-encodeur + K-Means

6.3.1 . Spam

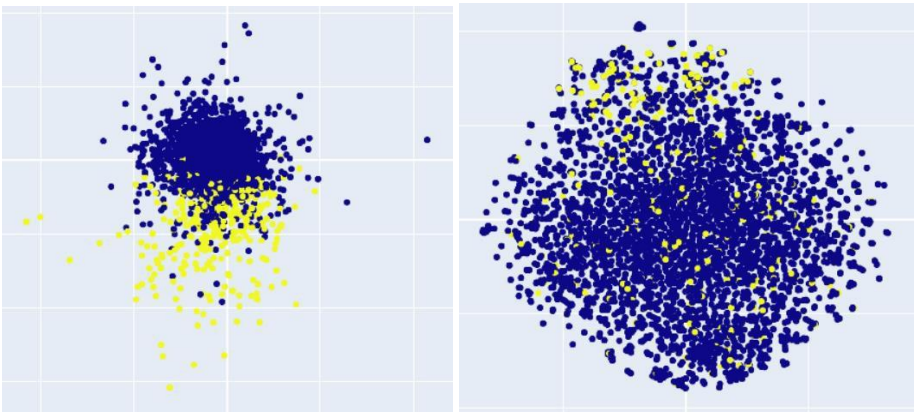

Figure 25. Résultat de l'auto-encodeur + Kmeans sur la matrice documents-termes avec projection des résultats sur AC (à gauche) et t-SNE (à droite).

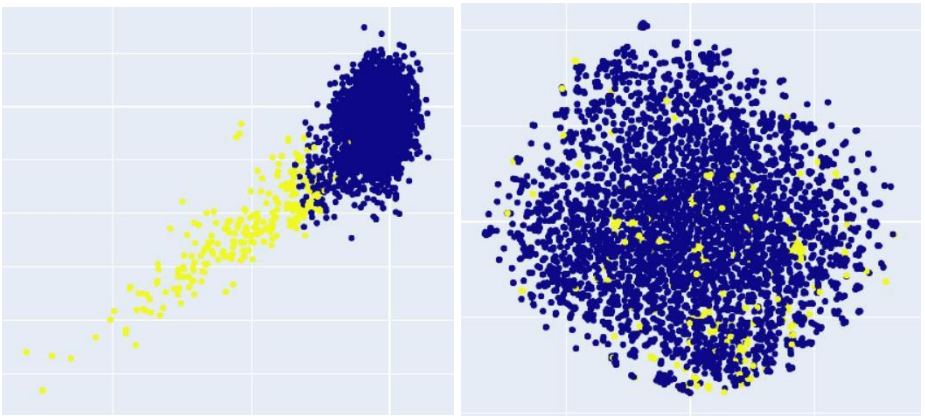

Figure 26. Résultat de l'auto-encodeur + Kmeans sur la matrice TF-IDF avec projection des résultats sur AC (à gauche) et t-SNE (à droite).

On remarque que l'apport de TF-IDF permet une meilleure séparation entre les classes et permet d'éviter les faux positifs et faux négatifs, cependant il est important de souligner que ce n'est pas le cas pour la version t-SNE.

6.3.2 20NewsGroup

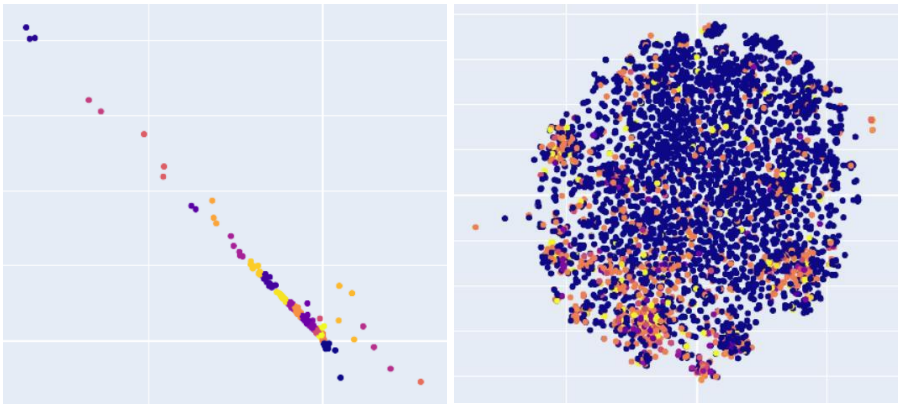

Figure 27. Résultat de l'auto-encodeur + Kmeans sur la matrice documents-termes avec projection des résultats sur AC (à gauche) et t-SNE (à droite).

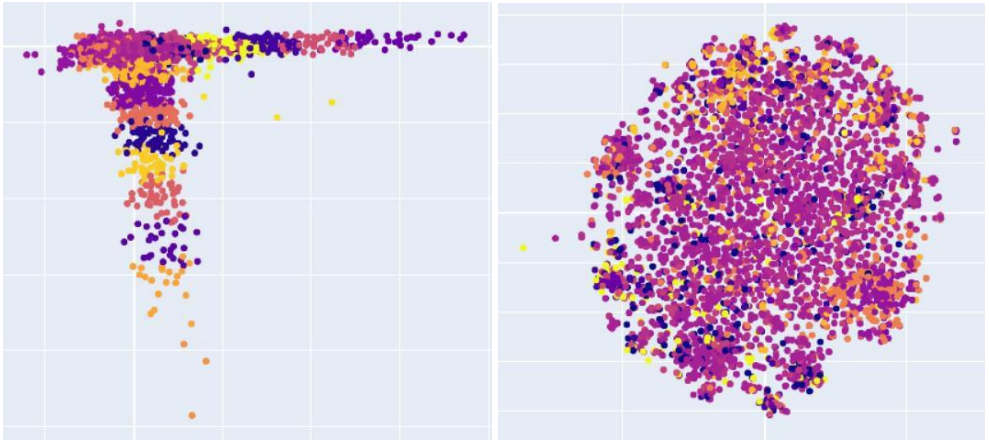

Figure 28. Résultat de l'auto-encodeur + Kmeans sur la matrice TF-IDF avec projection des résultats sur AC (à gauche) et t-SNE (à droite).

Les résultats sont similaires au jeu de données spam, TF-IDF permet une meilleure séparation entre les classes mais t-SNE ne permet pas de bien distinguer les classes.

Il serait intéressant d'étudier l'algorithme DCN afin d'avoir une **vision globale** (utilisation simultanée d'une méthode de réduction de la dimension et d'un algorithme de clustering) plutôt qu'une **vision locale** (utilisation séquentielle d'une méthode de réduction de la dimension et d'un algorithme de clustering).

DCN

7. DCN

Un DCN est un algorithme de clustering combiné à un encodeur automatique afin d'apprendre la réduction de dimension et le clustering simultanément, en effet l'encodeur, le décodeur et l'algorithme de clustering sont optimisés en même temps.

7.1 Description de l'algorithme

La fonction objective de DCN est la suivante :

$$\min_{\boldsymbol{\theta_1},\boldsymbol{\theta_2},\boldsymbol{S},\boldsymbol{M}} ||\boldsymbol{X} - \boldsymbol{g_{\theta_2}}(\boldsymbol{f_{\theta_1}}(\mathbf{x}))||^2 + \boldsymbol{\lambda}||\boldsymbol{f_{\theta_1}}(\mathbf{x}) - \mathbf{SM^T}||^2$$

La fonction suivante peut être décomposée en deux termes distincts :

- Le premier terme permet de calculer l'erreur de reconstruction de l'input avec l'auto-encodeur en faisant la différence entre l'input initial X et la reconstruction de la matrice $\hat{X}$
- Le deuxième terme permet de calculer l'erreur de classification (fonction objective de K-Means), en effet le second terme revient à mesurer la différence entre la matrice approximative de X et le produit des matrices S (label) et M (centre) généré par K-Means.

Concernant les paramètres on a :

- X : Matrice des données initiales
- f_{θ_1}(x) : Approximation des données initial obtenue à partir de l'auto-encodeur
- g_{θ_2}(x) : Décodeur pour reconstituer l'input à partir de l'auto-encodeur
- S : Matrice des clusterings (labels).
- M : Matrice de centroïde.
- λ : Paramètre permettant de réguler la fonction objective.

7.2 Performance de DCN

DCN nous retourne deux matrices avec leurs labels associés, la première correspond au données initiale (image à gauche) et la seconde matrice correspond au résultat final de DCN sur la nouvelle représentation des données à partir de l'auto-encodeur (image à droite).

7.2.1 . Spam

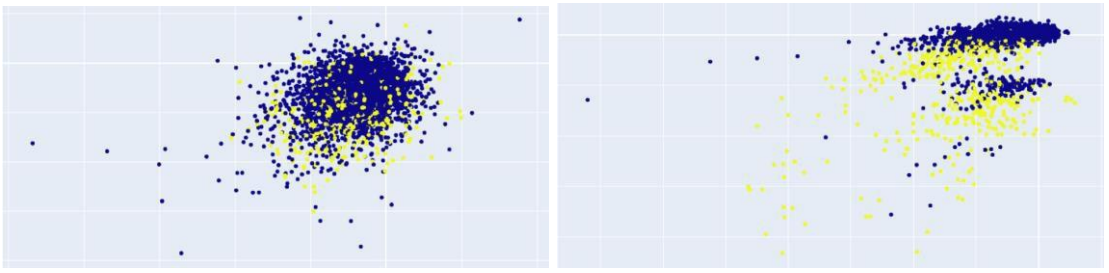

Figure 29. Résultat de DCN sur la matrice documents-termes

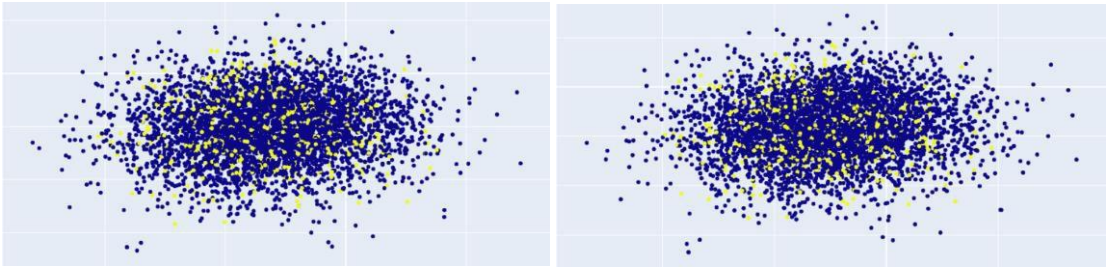

Figure 30. Résultat de DCN sur la matrice TF-IDF

On remarque que l'apport de DCN sur la matrice document-terme permet une meilleure séparation entre les classes spam et non spam et une meilleure visualisation contrairement aux algorithmes précédant.

L'apport de TF-IDF n'est pas important dans la séparation et la visualisation des classes.

7.2.2 20NewsGroup

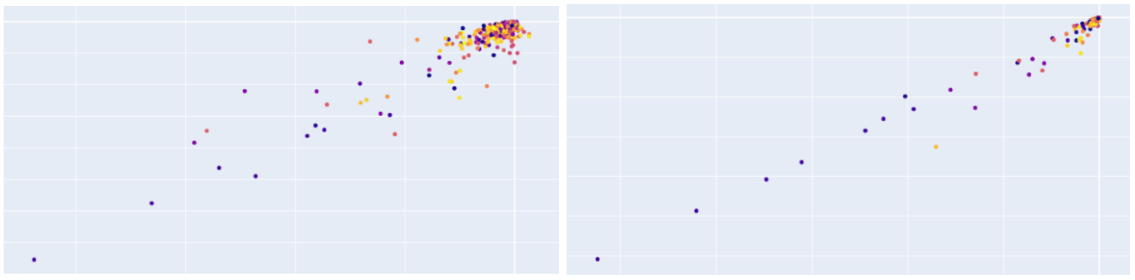

Figure 31. Résultat de DCN sur la matrice documents-termes

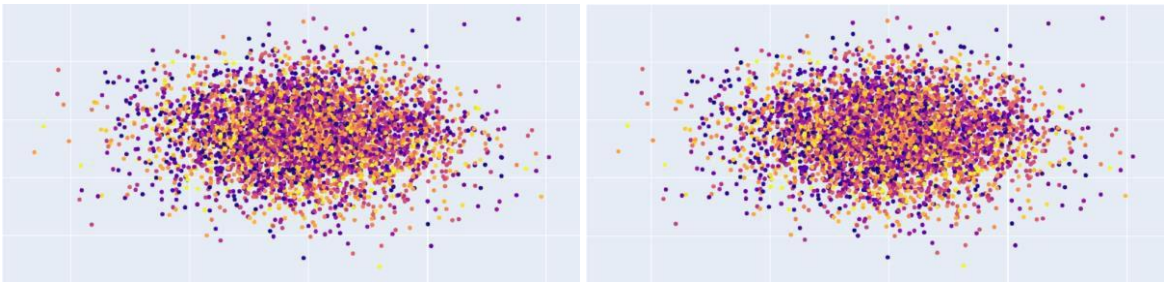

Figure 32. Résultat de DCN sur la matrice TF-IDF

Concernant le jeu de données 20NewsGroup, on remarque une similarité entre les deux jeux de données la version document-terme et TF-IDF ne permettent pas de classifier les documents et d'avoir une bonne visualisation.

DCN RÉGULARISÉ

8. DCN Régularisé

Dans le but d'optimiser DCN sur des données textuelles et par conséquent améliorer ses performances, il est d'usage en clustering d'intégrer une information additionnelle sous forme d'un graphe de similarité W (document x document).

Dans ce cas, DCN régularisé aura comme input le dataset initial et une information supplémentaire W qui mettrons à jours les paramètres (θ_1, θ_2, matrices S, M et B) de la fonction objective afin de minimiser l'erreur.

De plus, le défi revient à ajouter les mises à jour de B, car B fait le relai entre les deux termes : l'information fournie par W mais aussi par le clustering à chaque itération et l'approximation des données initiales obtenue à partir de l'auto-encodeur.

8.1 Description de l'algorithme

Pour introduire l'information supplémentaire W dans la formule objective de DCN régularisé, nous devons optimiser la fonction objective suivante :

$$\min_{\boldsymbol{\theta_1},\boldsymbol{\theta_2},\boldsymbol{S},\boldsymbol{M},\boldsymbol{B}} ||\boldsymbol{X} - \boldsymbol{g_{\theta_2}}(\boldsymbol{f_{\theta_1}}(\mathbf{x}))||^2 + \boldsymbol{\lambda_1}||\mathbf{W} - \mathbf{SMB^T}||^2 + \boldsymbol{\lambda_2}||\boldsymbol{f_{\theta_1}}(\mathbf{x}) - \mathbf{B}||^2$$

La fonction suivante peut être décomposée en trois termes distincts :

- Le premier terme permet calculer l'erreur de reconstruction de l'input avec l'auto-encoder en faisant la différence entre l'input initial X et la reconstruction de la matrice $\hat{X}$
- Le deuxième terme permet de calculer l'erreur de classification (fonction objective de K-Means) et permet d'intégrer un graphe de similarité document * document, en effet le second terme revient à ajouter une information additionnelle, au départ on a que notre X ensuite on va traiter

un jeu de donnée qui concerne toujours un document mais dans lequel on a une information supplémentaire qui permet de voir les similarités entre les documents deux à deux pour aider le modèle à mieux constituer les groupes.

- Le troisième terme nous permet de faire le lien entre le premier et le deuxième terme. On veut imposer la contrainte : B soit le plus proche possible de l'encodeur, c'est donc une manière de relié l'auto-encodeur au deuxième terme qui va faire le clustering et l'intégration de la similarité document * document.

Concernant les paramètres on a :

- X : Matrice des données initiales
- f_{θ_1}(x) : Approximation des données initial obtenue à partir de l'auto-encodeur
- g_{θ_2}(x) : Décodeur pour reconstituer l'input à partir de l'auto-encodeur
- W : Matrice document * document permettant d'ajouter une information supplémentaire afin d'améliorer les performances.
- S : Matrice des clusterings (labels).
- M : Matrice de centroïde.
- B : Matrice des embeddings, c'est-à-dire une représentation des documents dans un espace réduit.
- λ_1, λ_2 : Paramètres permettant le contrôle et la calibration de la fonction objective.

8.2 Implémentation du papier de recherche

Pour résumer, la différence (entre DCN et DCN régularisé) consiste à l'ajout d'une information supplémentaire qu'on va intégrer dans le code source de DCN.

Les modifications apportées à DCN sont :

- Dans la fonction objective, on ajoute W matrice de dimension (document * document), c'est une information supplémentaire, en effet chaque W_{ij} va croiser les documents deux à deux et nous donner la similarité entre document.

  Si les documents sont similaires alors leur score de similarité sera proche de 1, si les documents sont dissimilaires alors leur score de similarité sera proche de 0.

- Dans la fonction objective, on ajoute B, matrice de dimension (document * nombre de classe), le deuxième et le troisième terme dépendent de B.

  On va optimiser les paramètres de l'auto-encodeur, S et M à partir d'un B donner, donc pour la première itération on va initialiser B de façon aléatoire et ensuite le mettre à jour en fonction de S, M et $f(\mathrm{x})$.

La matrice B va donc être mise à jour à partir d'une svd qui regroupe tous les paramètres du deuxième et troisième terme.

Cependant il est intéressant de noter que le deuxième terme $|| \boldsymbol{W} - \boldsymbol{SMB^t} ||^2$ peut être réécrit comme $|| \boldsymbol{WB} - \boldsymbol{SM} ||^2$, si on remplace WB par f(x) on remarque la similitude par rapport à la fonction objective de K-Means.

On va donc appliquer la formule suivante : $\boldsymbol{U}, \boldsymbol{\Sigma}, \boldsymbol{V} = \boldsymbol{SVD}(\boldsymbol{WSM} + \boldsymbol{\lambda_2}\ \boldsymbol{f}\ (\mathbf{x}))$ à chaque itération ce qui nous permettra d'obtenir une matrice qui dépend des paramètres qui viennent d'être mise à jour (S, M et $f(\mathrm{x})$) à travers la formule suivante : $\boldsymbol{B} = \boldsymbol{U}\mathbf{V^t}$.

Donc à chaque itération, on va avoir deux blocs, le premier bloc permettra de calculer les paramètres de l'auto-encodeur et S et M étant donnée B et un deuxième bloc qui va mettre à jour B.

8.3 Performance de DCN Régularisé

DCN régularisé nous retourne deux matrices avec leurs labels associés, la première correspond au données initiale (image à gauche) et la seconde matrice correspond au résultat final de DCN régularisé sur la nouvelle représentation des données à partir de l'auto-encodeur (image à droite).

8.3.1 . Spam

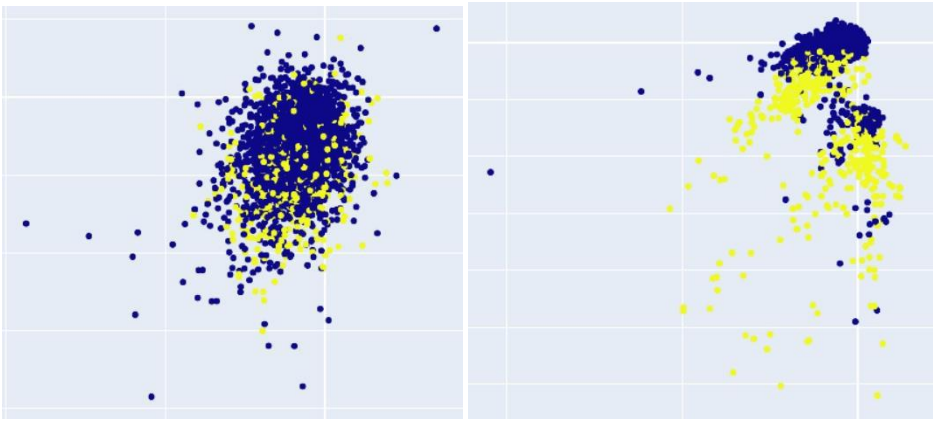

Figure 33. Résultat de DCN régularisé sur la matrice documents-termes

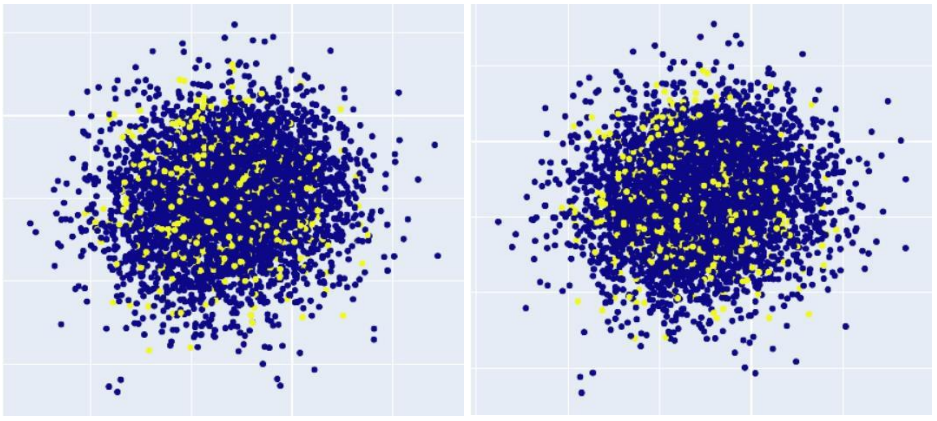

Figure 34. Résultat de DCN régularisé sur la matrice TF-IDF

On remarque que l'apport de l'auto-encodeur sur la matrice document-terme permet une meilleure séparation entre les classes spam et non spam et une meilleure visualisation contrairement aux algorithmes précédents.

L'apport de TF-IDF n'est pas important dans la séparation et la visualisation des classes.

8.3.2 . 20NewsGroup

Figure 35. Résultat de DCN régularisé sur la matrice documents-termes

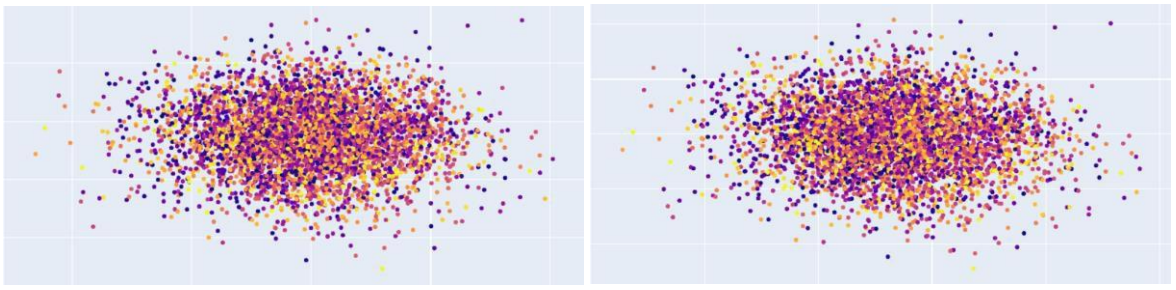

Figure 36. Résultat de DCN régularisé sur la matrice TF-IDF

Concernant le jeu de données 20NewsGroup, on remarque une similarité entre les deux jeux de données la version document-terme et TF-IDF ne permettent pas de classifier les documents et d’avoir une bonne visualisation.

CONCLUSION GÉNÉRALE

9. Conclusion Générale

9.1 Conclusion

Dataset Spam	**Doc-Terme**	**TF - IDF**
Spherical K means	ARI : 0.25 NMI : 0.25 **ACC : 0.77**	ARI : 0.01 NMI : 0.11 ACC : 0.43
Spherical K means + AFC	**ARI : 0.48** NMI : 0.40 ACC : 0.13	ARI : 0.17 NMI : 0.08 **ACC : 0.76**
Spherical K means + t-SNE	ARI : 0.05 NMI : 0.09 ACC : 0.39	ARI : 0.09 NMI : 0.18 **ACC : 0.65**
NMF	**ARI : 0.46** NMI : 0.31 ACC : 0.13	ARI : 0.32 NMI : 0.14 ACC : 0.14
NMF + AFC	ARI : NA NMI : NA ACC : NA	ARI : NA NMI : NA ACC : NA
NMF + t-SNE	ARI : NA NMI : NA ACC : NA	ARI : NA NMI : NA ACC : NA

Auto-Encodeur + K-Means	**ARI : 0.48** NMI : 0.33 **ACC : 0.91**	ARI : 0.25 NMI : 0.12 **ACC : 0.87**
DCN	ARI : 0.26 **NMI : 0.46** **ACC : 0.89**	ARI : 0.00 NMI : 0.00 ACC : 0.51
DCN régularisé	ARI : 0.27 **NMI : 0.47** **ACC : 0.90**	ARI : 0.00 NMI : 0.00 ACC : 0.51

Tableau 1. Performance des différents algorithmes sur le dataset Spam

Pour le dataset **spam** si on compare les résultats de NMI, ARI, et Accuracy pour les deux versions (document-terme et TF-IDF) on remarque clairement que nous obtenons des meilleures performances sur la version document-terme surtout au niveau des algorithmes **auto-encoder + K-Means** et **DCN régularisé**, en effet nous avons une accuracy égale à 91% et 90% et un NMI égal à 46 % et 47 %.

Cependant, les performances obtenues à partir de la matrice TF-IDF sont loin de nos attentes.

De plus, il est important de rappeler que Spherical K-Means est conçu pour des données textuelles, c'est pour cela qu'il obtient les meilleures valeurs sur la version document-terme avec une ARI égale à 0.48 et sur la version TF-IDF, une accuracy égale à 0.76.

Dataset 20NewsGroup	Doc-Terme	TF-IDF
Spherical K means	**ARI : 0.15** **NMI : 0.31** Acc : 0.04	**ARI : 0.25** **NMI : 0.40** Acc : 0.07
Spherical K means + AFC	ARI : 0.04 NMI : 0.11 Acc : 0.04	ARI : 0.02 NMI : 0.08 Acc : 0.04
Spherical K means + t-SNE	ARI : 0.04 NMI : 0.10 Acc : 0.05	ARI : 0.14 NMI : 0.28 Acc : 0.04
NMF	ARI : 0.07 **NMI : 0.18** **Acc : 0.07**	ARI : 0.16 NMI : 0.31 Acc : 0.05
NMF + AFC	ARI : NA NMI : NA Acc : NA	ARI : NA NMI : NA Acc : NA

NMF + t-SNE	ARI : NA NMI : NA Acc : NA	ARI : NA NMI : NA Acc : NA
Auto-Encodeur + K-Means	ARI : 0.01 NMI : 0.05 Acc : 0.04	ARI : 0.03 NMI : 0.15 Acc : 0.05
DCN	ARI : 0.03 NMI : 0.00 **Acc : 0.08**	ARI : 0.02 NMI : 0.00 **Acc : 0.09**
DCN régularisé	ARI : 0.03 NMI : 0.01 **Acc : 0.10**	ARI : 0.02 NMI : 0.00 **Acc : 0.10**

Tableau 2. Performance des différents algorithmes sur le dataset 20NewsGroup

Pour le dataset **20NewsGroup** si on étudie les résultats de NMI, ARI, et Accuracy pour les deux versions (Document-Terme et TF-IDF) on remarque que la version TF-IDF nous permet d'avoir des bonnes performances comparer à la version Document-Terme (contrairement au jeu de donnée spam) surtout avec l'algorithme Spherical K-Means avec une ARI égale à 0.25 et un NMI égal à 0.40.

Pour le reste des algorithmes on enregistre des performances assez faibles.

En conclusion des deux jeux de données, on s'aperçoit que le jeu donné sur lequel on applique l'algorithme influence les résultats et les performances de ces derniers.

De plus, même des algorithmes de clustering spécifique aux données textuelles ne nous donnent pas de bonnes performances.

Pour appuyer nos propos, on peut citer les algorithmes DCN et DCN régularisé, leur performance varie en fonction du dataset, par exemple on obtient de meilleures performances sur le dataset Spam plutôt que sur dataset 20NewsGroups malgré la nature commune des deux datasets.

L'utilisation de ce genre d'application dans des données images (MNIST dataset) permet d'avoir de meilleure performance car on n'est pas confronté au problème de sparsité et de dimensionnalité.

9.2 Perspective et Evolution

Concernant les évolutions, nous pouvons remplacer la fonction objective de K-Means (utilisée au niveau de DCN) par celle de Spherical K-Means car elle est beaucoup plus adaptée aux données textuelles.

Concernant les perspectives, il existe d'autres méthodes permettant d'améliorer les performances des méthodes de clustering sur des données textuelles, nous allons définir une autre approche connue appelée **Ensemble Clustering**.

Le principe de la méthode Ensemble Clustering est assez simple, il vise à fusionner plusieurs partitions en une partition de consensus, où chaque partition est obtenue en appliquant une méthode de clustering sur le même jeu de données.

Par exemple, on peut par exemple utiliser plusieurs DCN afin d'obtenir une partition unique.

Nous pouvons aussi étudier une autre approche nommée **Deep Ensemble Auto-Encoder,** qui consiste à utiliser plusieurs auto-encodeurs (plutôt qu'un seul) avec des architectures différentes **(**nombre de couche, fonction d'activation, …) et ensuite appliquer un clustering sur un ensemble de plusieurs auto-encodeur, ceci permettra de combler les difficultés du DCN car pour un dataset donnée on ne connaît pas l'architecture la plus adaptée.

RÉFÉRENCES

10. Références

[1] Yang, B., Fu, X., Sidiropoulos, N. D., & Hong, M. (2017, July). Towards k-means-friendly spaces: Simultaneous deep learning and clustering. In international conference on machine learning (pp. 3861-3870). PMLR.