UNIVERSITÉ DU QUÉBEC EN OUTAOUAIS

CLASSIFICATION DE DOCUMENTS MULTIMÉDIAS EN COMBINANT LE TEXTE ET L'IMAGE : APPROCHE PAR LE MODÈLE LDA ET L'APPRENTISSAGE PROFOND

MÉMOIRE
PRÉSENTÉ
COMME EXIGENCE PARTIELLE
DE LA MAÎTRISE EN SCIENCES ET TECHNOLOGIES DE L'INFORMATION

PAR

SABRI KRICHEN

DÉCEMBRE 2019

UNIVERSITÉ DU QUÉBEC EN OUTAOUAIS

Département d'informatique et d'ingénierie

CLASSIFICATION DE DOCUMENTS MULTIMÉDIAS EN COMBINANT LE TEXTE ET L'IMAGE : APPROCHE PAR LE MODÈLE LDA ET L'APPRENTISSAGE PROFOND

Sabri Krichen

pour l'obtention du grade de maître ès science (M.Sc.)

Ce mémoire a été évalué par un jury composé des personnes suivantes :

Dr. Larbi Talbi ..... Président du jury

Dr. Alan Davoust ..... Membre du jury

Dr. Mohand Saïd Allili ..... Directeur de recherche

Mémoire accepté le : .....

Dédicaces

À ALLAH Le Très-Haut, le très miséricordieux qui ma donné la volonté et le courage pour la réalisation de ce travail.

À ma mère Ghoulem Rakia et mon père Krichen Moncef qui grâce à leurs sacrifices, leurs amours et leurs bénédictions j'ai pu réaliser ce travail.

À ma femme Nancy pour sa patience, sacrifices et amour.

À mes frères et soeurs pour leurs soutiens.

À mes deux filles Yasmine et Malek

Remerciements

Avant tout, je voudrais exprimer ma gratitude à mon directeur de recherche Dr Mohand Saïd Allili pour son soutien, sa patience et ses conseils lors de la finalisation de ce projet de mémoire. Je remercie également les membres de mon jury pour le temps et l'attention qu'ils l'ont accordé à mon travail.

Je tiens à remercier aussi ma famille et mes amis, grâce à leurs prières et leurs encouragements, j'ai pu surmonter tous les obstacles. Le travail dans ce projet de mémoire n'aurait pas pu être accompli sans leurs soutiens.

Table des matières

Liste des figures

Liste des tableaux

Liste des abréviations, sigles et acronymes

LDA Latent Dirichlet Allocation

API Application Programming Interface

LSA Latent Semantic Analysis

PLSA Probabilistic Latent Semantic Analysis

NLP Natural language processing

ANN Artificial neural network

SIFT Scale-invariant feature transform

HOG Histogram of oriented gradients

AI Artificial Intelligence

BOW Bag Of Words

HTML Hypertext Markup Language

XML Extensible Markup Language

CNN Convolutional Neural Network

KPPV Algorithme de K plus proches voisins

ML Maximum Likelihood

NB Naive Bayes

SVD Singular Value Decomposition

SVM Support Vector Machine

TF-IDF Term Frequency-Inverse Document Frequency

Résumé

La classification de documents multimédias (ex. pages Web) a été longtemps basée sur les algorithmes de classification de texte tels que les machines à vecteurs de support et la naïve bayésienne. Bien que ces méthodes aient eu beaucoup de succès, la majorité d'entre elles sont axées sur l'analyse des occurrences de mots sans tenir compte de l'ordre de leur apparition ni du rôle syntaxique qu'ils jouent, ce qui accentue l'écart sémantique. Outre le contenu textuel de documents multimédias, le contenu visuel (ex. images et vidéos) peut fournir également de nombreuses informations précieuses qui peuvent faciliter la classification thématique de documents. Peu sont les travaux qui se concentrent sur la classification automatique de documents en se basant sur les caractéristiques textuelles et visuelles conjointement.

Considérant ces limitations, nous proposons une solution basée sur l'apprentissage automatique pour la modélisation conjointe de ces deux modalités (texte et images) pour la classification de documents multimédias. Ainsi, nous exploitons les derniers développements de l'apprentissage profond pour extraire l'information visuelle de haut niveau des images contenu dans les documents. Nous proposons par la suite une approche probabiliste basée sur l'allocation de Dirichlet latente (LDA) pour combiner l'information textuelle et visuelle pour une classification thématique de documents multimédias. Les résultats obtenus par notre méthode ont montré une bonne performance en comparaison à des méthodes de l'état de l'art.

Abstract

The classification of multimedia documents has long been based on text classification algorithms such as support vector machines and naïve Bayes. Although these methods have been very successful, the majority of them are focused on the analysis of word occurrences without regard to the order of their appearance or the syntactical role they play, which accentuates the semantic gap. In addition to the textual content of web pages, visual content (e.g. images and videos) can also provide a lot of valuable information that can facilitate the thematic classification of the content of a web page. Few work focuses on the automatic classification of multimedia documents based on textual and visual characteristics together.

Considering these limitations, we propose a solution based on machine learning for the joint modelling of these two modalities (text and images) for the classification of multimedia documents. Thus, we exploit the latest developments in deep learning to extract high-level visual information from images contained in documents. We then propose a probabilistic approach based on latent Dirichlet allocation (LDA) to combine textual and visual information for a thematic classification of multimedia documents. The results obtained by our method showed a good performance in comparison with state of the art methods.

Chapitre 1

Introduction

1.1 Généralités

Les dernières années ont vu une croissance rapide des documents multimédias sur Internet (ex. pages Web, diaporamas, vidéos, courriels, etc.). Ces documents multimédias intègrent un ensemble d'informations et de ressources numériques de types différents, notamment des textes et des images. La croissance exponentielle de ces données nécessite des technologies évolutives, efficaces et robustes pour les gérer et les indexer. Parmi ces technologies, on trouve l'intelligence artificielle qui comporte un ensemble de théories et des techniques comme la fouille de texte et d'images. Elle désigne un ensemble de traitements informatiques consistant à extraire des connaissances selon un critère de nouveauté ou de similarité dans des textes produits par des humains. Dans la pratique, cela revient à mettre en algorithme un modèle simplifié des théories linguistiques dans des systèmes informatiques d'apprentissage et de statistiques, et des technologies de compréhension du langage naturel. Aussi, la vision par ordinateur a pour but de permettre à une machine d'analyser, traiter et comprendre une ou plusieurs images prises par un système d'acquisition (ex. caméras, etc.).

Pour mieux organiser et indexer le nombre grandissant de documents multimédias, il est devenu impératif, premièrement, d'exploiter plusieurs caractéristiques existantes dans ces dernières et, deuxièmement, d'y identifier et d'y détecter automatiquement le contenu sémantique. Ceci requiert à son tour un système de classification automatique de documents basé sur la combinaison des caractéristiques textuelles et visuelles.

Dans le passé, la plupart des moteurs de recherche utilisent des propriétés syntaxiques de bas niveau pour caractériser la sémantique des documents multimédias, autrement dit, ils considèrent les mots clés tels quels, au lieu de chercher leur signification sémantique. Cela peut emmener à une discordance entre les requêtes des utilisateurs et les réponses du moteur de recherche, problème communément appelé : vide sémantique. En d'autres termes, les jugements de similarité humaine n'obéissent pas aux exigences de la métrique de similarité objective utilisée dans les moteurs de recherche. En réalité, un certain mot peut être interprété de différentes manières dans différents contextes (polysémie), alors que le même concept est généralement associé à un ensemble de termes différents et que les personnes peuvent avoir des préférences différentes quant à celui à utiliser.

Pour une classification automatique efficace des documents multimédias, il serait alors désavantageux de se limiter uniquement aux propriétés de bas niveau telles que les mots clés. La solution est alors d'aller vers un niveau d'abstraction plus élevé permettant une meilleure organisation des documents multimédias en créant des index qui reflètent leurs contenus sémantiques. Une des approches qui peuvent répondre à ce besoin est **la classification thématique des documents multimédias** représentée par le modèle probabiliste, l'allocation de Dirichlet latente (LDA) [1]. C'est un modèle probabiliste génératif qui permet de décrire des collections de documents de texte ou d'autres types de données discrètes. Il fait partie d'une catégorie de modèles appelés **modèles thématiques** qui cherchent à découvrir des structures thématiques cachées dans des collections de documents. Nous voulons montrer dans ce travail l'efficacité de cette approche pour le problème du traitement et de l'organisation de documents multimédias.

1.2 Problématique

La classification des documents multimédias a été longtemps basée sur les algorithmes de classification de texte tels que les machines à vecteurs de support [2] et la technique bayésienne naïve [3]. Bien que ces méthodes aient eu beaucoup de succès, la majorité d'entre elles sont axées sur l'analyse des occurrences de mots sans tenir compte de l'ordre de leur apparition ni du rôle syntaxique qu'ils jouent. Cela donne lieu à ce qu'on appelle un écart sémantique. Outre le contenu textuel des documents multimédias, le contenu visuel (ex. images et vidéos) peut fournir également de nombreuses informations précieuses qui peuvent améliorer la performance de la classification des documents

multimédias. Peu sont les travaux qui se concentrent sur la classification automatique de documents multimédia en se basant sur les caractéristiques textuelles et visuelles conjointement.

Considérant ces limitations, nous proposons une solution basée sur l'apprentissage automatique pour la modélisation conjointe de ces deux modalités (texte et image) pour la classification de documents multimédias (ex. pages Web). Ainsi, nous exploitons les derniers développements de l'apprentissage profond pour extraire l'information visuelle de haut niveau des images contenu dans les documents. Nous proposons par la suite une approche probabiliste basée sur le modèle LDA pour combiner l'information textuelle et visuelle pour une classification thématique de documents multimédias.

Pour valider et évaluer notre approche, nous considérons la classification des articles dans les pages Web d'actualités. La classification de ce genre de documents pose plusieurs problèmes de recherche en raison du grand nombre de caractéristiques multimodales présentes dans le jeu de documents et de leurs dépendances. Nous démontrons, entre autres, que l'intégration de l'information textuelle et visuelle dans le modèle LDA permet une classification plus précise de documents que l'exploitation de contenu purement textuel ou visuel séparément.

1.3 Contribution

Depuis l'invention du Web, plusieurs approches ont été proposées pour classer automatiquement les pages Web. Souvent, ces méthodes sont effectuées en s'appuyant sur le contenu textuel d'une page Web, mettant ainsi en œuvre diverses méthodes d'analyse de texte comme la représentation de sac à mots basés sur la fréquence de mots et des algorithmes tels que les machines à vecteurs de support. Dans la plupart des cas, les informations structurées contenues dans le balisage hypertexte des pages Web sont utilisées aussi comme entrée supplémentaire pour les processus de classification. Notre principale contribution dans ce mémoire est l'exploitation du contenu visuel et textuel conjointement de documents multimédias afin de permettre une meilleure classification de ces derniers. En effet, l'humain analyse souvent le texte et l'image pour comprendre le sens des documents multimédias (ex. articles, courriel, etc.). Il est donc naturel et important que tout système de classification de documents multimédias doit faire usage

de ces modalités.
Nous explorons, dans ce travail, cette combinaison dans le cadre de l'analyse sémantique latente et l'apprentissage profond. Par ailleurs, la plupart des bases de données des documents multimédias qui existent à l'heure actuelle ne contiennent que les informations qui concernent les caractéristiques textuelles. Donc, il était nécessaire de construire une base de données qui comporte les caractéristiques visuelles (ex. lien URL de l'image et sa description sous forme d'annotations) ainsi que les caractéristiques textuelles (ex. lien URL article, titre article, contenue textuelle, etc.). Plus de 6500 articles ont été collectés à partir de différents pages Web d'actualités numériques.

1.4 Structure du mémoire

Le chapitre 2 met en évidence les techniques et méthodologies importantes utilisées dans la classification des documents axés principalement sur les techniques d'apprentissage automatique. Le chapitre 3 présente la représentation thématique de documents textuels en présentant les modèles thématiques tels que le modèle d'allocation de Dirichlet latente. Dans le chapitre 4, on présente notre méthodologie suivie par les expériences et résultats. Enfin, des conclusions sont tirées au chapitre 5.

Chapitre 2

Apprentissage automatique pour la classification de documents

2.1 Introduction

Avec la génération croissante des documents multimédias, la classification automatique de ces derniers est devenue une tâche importante pour pouvoir mieux les indexer afin de garantir aux utilisateurs du Web une recherche efficace et rapide. L'analyse et le classement de documents multimédias tels que les pages Web d'actualités, les courriels, etc., qui intègrent un ensemble d'informations et des ressources numériques de types différents, notamment des textes et des images, nécessitent des techniques d'analyse avancées. Ce chapitre met en évidence les concepts fondamentaux de l'apprentissage automatique pour la classification des documents multimédias.

2.2 Travaux connexes

Dans nos jours, le Web est l'une des sources d'information les plus riches et il est en constante expansion et évolution. Il permet d'accéder à de nombreuses pages qui intègrent un ensemble d'informations et de ressources numériques de types différents, notamment du texte et des images, des vidéos, etc. La fouille du Web est l'application de méthodes d'apprentissage automatique au contenu Web pour découvrir des motifs et représentations utiles de ce dernier. Elle rassemble plusieurs domaines de recherche, tels que la recherche d'informations, le traitement automatique du langage naturel et la

fouille de données. La classification des pages Web est le processus d'attribution d'étiquettes de catégorie prédéfinies aux pages Web. Le volume d'informations disponibles sur le Web est très important. Il est donc impossible de classer les pages Web manuellement, d'où la nécessité d'un système de classification automatique.

Plusieurs recherches ont été faites dans le domaine de la classification des pages Web. Différentes des textes purs, les pages Web contiennent plusieurs données intéressantes qui peuvent être utilisées pour améliorer les performances de classification. Sun et al. [4] ont proposé d'utiliser des machines à vecteurs de support SVM avec des caractéristiques textuelles et contextuelles pour la classification des pages Web. Afin d'examiner l'efficacité des caractéristiques, le titre et des liens hypertextes dans les pages Web ont été utilisés comme caractéristiques de contexte. Les résultats expérimentaux ont indiqué que SVM fonctionne bien dans la classification des pages Web, en particulier avec les caractéristiques contextuelles.

Chen et Hsieh [5] ont proposé une méthode de classification des pages Web en utilisant un SVM basé sur un schéma de vote pondéré. Une analyse sémantique latente LSA a été utilisée pour trouver les relations sémantiques entre les mots clés et les pages disponibles. En parallèle, les caractéristiques textuelles de la page sont également extraites. Ensuite, les informations latentes de la page ainsi que les caractéristiques extraites du texte ont été introduites dans SVM pour l'entrainement et les tests respectivement. Sur la base de la sortie du SVM, un schéma de vote a été utilisé pour déterminer la catégorie de la page Web.

Shen et al. [6] ont proposé un algorithme de classification de pages Web basé sur des résumés de pages Web, générés par des experts humains. Les résultats expérimentaux ont montré que l'algorithme a obtenu une amélioration d'environ 8,8% par rapport aux algorithmes de classification basés sur du texte pur.

les auteurs dans [7] ont proposé une autre approche en combinant le classificateur K le plus proche voisin $KppV$ et la recherche des règles d'association. Il a utilisé un nouveau système de pondération des caractéristiques basé sur des règles d'association ainsi qu'un système de vote pondéré par la distance, qui a permis au modèle de fonctionner pour n'importe quelle valeur de K (impair ou pair). Les expérimentations effectuées sur la

base de données *WebKB* avaient montré que la précision de classification de la méthode proposée était nettement meilleure que celles des travaux existants.

Le contenu visuel est peu utilisé pour classer les pages Web car, traditionnellement, seules les informations textuelles sont utilisées, ce qui permet d'obtenir une précision raisonnable. Il a cependant été remarqué dans [8] que le contenu visuel peut aider à lever l'ambiguïté de la classification basée uniquement sur ce contenu textuel. De plus, un autre facteur en faveur de l'utilisation du contenu visuel est le fait que les variables subjectives comme la conception des pages Web et la valeur esthétique ne peuvent pas être étudiées en utilisant le contenu textuel contenu dans le code HTML. Ces variables gagnent en importance en raison des stratégies de marketing sur le Web.

Boer et al. [8] ont réussi à classer les pages Web en utilisant uniquement des caractéristiques visuelles. Ils ont classé les pages en deux variables binaires : valeur esthétique et valeur de conception. Ils ont obtenu une bonne précision de classification. Les auteurs ont également appliqué le même algorithme de classification et les mêmes méthodes à une classification en classes multiples du sujet des sites Web et bien que les résultats obtenus soient raisonnables, il a été conclu que cette classification est plus difficile à effectuer.

Asirvatham et al. [9] ont proposé une approche de classification basée sur le contenu visuel, où un certain nombre de caractéristiques visuelles comme la structure des pages Web, ainsi que des caractéristiques textuelles, ont été utilisées. La combinaison des caractéristiques visuelles et textuelles a augmenté la précision de la classification.

2.3 Rappels sur les techniques d'apprentissage

L'apprentissage automatique est une branche de l'intelligence artificielle qui utilise les données afin d'apprendre des modèles/algorithmes pour la prédiction. Les données peuvent être étiquetées ou non, dépendamment du type d'apprentissage souhaité [10] . Il existe principalement deux types de techniques d'apprentissage automatique, l'apprentissage supervisé et l'apprentissage non supervisé.

En apprentissage supervisé, le système apprend à classer les documents selon un modèle de classification. Il reçoit des exemples d'entrées qui sont étiquetés avec les sorties

souhaitées. L'algorithme compare les étiquettes prédites avec les étiquettes réelles pour trouver des erreurs et modifie le modèle en conséquence. Ce dernier est une fonction de prédiction utilisée par la suite pour prédire les valeurs d'étiquettes sur des données supplémentaires non étiquetées. Les techniques d'apprentissage supervisé peuvent être subdivisées en classification et en régression selon le type de variable de réponse (catégoriel ou numérique). Les techniques de classification supposent une variable de réponse catégorielle et le but est de classer les instances en fonction des variables prédictives. Les techniques de régression supposent une variable de réponse de type réel. Le but est de trouver une fonction qui correspond aux données avec le moins d'erreurs possible. Dans certains cas, il est pratique de transformer une variable de réponse numérique en une variable catégorielle et vice versa.

En apprentissage non supervisé, l'algorithme trouve tout seul des points similaires et des structures sous-jacentes à partir de données non étiquetées. En plus de découvrir des structures cachées dans un ensemble de données, l'objectif de l'apprentissage non supervisé permet aussi à l'ordinateur de découvrir automatiquement les représentations nécessaires pour classer les données brutes. Cette approche peut examiner des données complexes, plus abondantes et en apparence sans points communs, afin de les organiser de manière potentiellement significative. Ceci rend l'apprentissage non supervisé beaucoup plus complexe que l'apprentissage supervisé. Outre ces deux techniques, il existe également d'autres familles d'algorithmes d'apprentissage automatique, tels que l'apprentissage par renforcement [11] et l'apprentissage semi-supervisé [12].

2.4 Techniques d'apprentissage supervisé

Les algorithmes d'apprentissage supervisé utilisent des données dont les réponses sont connues pour entrainer et valider le modèle. Ils peuvent ajuster les paramètres à l'aide des étiquettes pour en déduire un modèle optimal en fonction des données disponibles. Un plus grand ensemble de données d'apprentissage est préférable, car il peut contenir suffisamment d'étiquettes pour que le modèle puisse faire face à toutes les situations possibles (c.-à-d une meilleure capacité de généralisation). Dans le cas contraire, on peut avoir un problème de surapprentissage consistant en un modèle d'apprentissage qui classe bien les données d'entrainement, mais moins bien les nouvelles données.

2.4.1 Algorithme de K plus proches voisins (KPPV)

Le KPPV est l'une des méthodes d'apprentissage automatique les plus simples et est souvent qualifié d'apprenant paresseux, car l'apprentissage n'est pas implémenté tant que la classification ou la prédiction n'est pas requise. Il prend la classe la plus fréquente, telle que mesurée par la distance euclidienne pondérée (ou une autre mesure de distance), parmi les k-exemples d'apprentissage les plus proches dans l'espace de représentation des données.

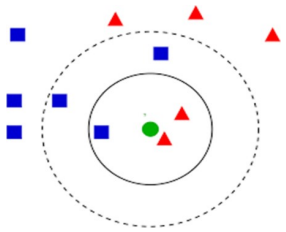


FIGURE 2.1 – Illustration de la classification par l'algorithme KPPV

La figure 2.1 illustre un exemple de classification avec $KPPV$. L'échantillon à tester, représenté par un cercle, doit être classé soit dans la première classe, représentée par des carrés, soit dans la deuxième classe, représentée par des triangles. Si $K = 3$ (cercle en trait plein), il est attribué à la deuxième classe, car il y a deux triangles et un seul carré à l'intérieur du cercle intérieur. Si $K = 5$ (cercle en trait pointillé), il est attribué à la première classe (trois carrés contre deux triangles à l'intérieur du cercle extérieur) [13].

Dans des problèmes spécifiques tels que la classification textuelle, il a été démontré que $KPPV$ fonctionne aussi bien que des modèles plus complexes [14]. L'inconvénient de l'utilisation de ce modèle est la lenteur de la classification, car les données doivent être mémorisées tout le temps. Cependant, nous pouvons augmenter la vitesse en utilisant des algorithmes de réduction de dimensionnalité ; par exemple, réduire le nombre d'attributs. Étant donné que l'apprentissage n'est pas implémenté avant la phase de classification, il s'agit d'un algorithme inapproprié à utiliser lorsque des décisions sont requises en temps réel.

2.4.2 Algorithme du Bayes naïf

Le classificateur du Bayes naïf (NB) est un classificateur probabiliste simple et couramment utilisé pour le texte. Ce modèle calcule la probabilité a posteriori d'une classe en utilisant la distribution des mots dans un document donné. C'est l'une des méthodes de base qui utilise la représentation de sac à mots en ignorant l'ordre des mots. L'idée générale de NB est illustrée dans la figure 2.2 [15].

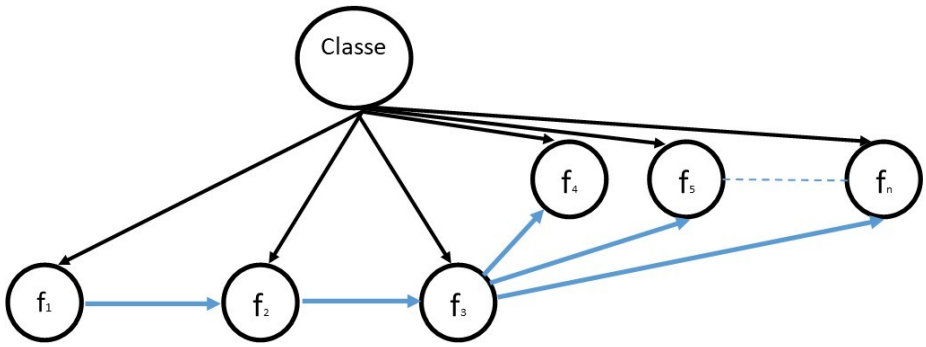


FIGURE 2.2 – Idée générale de l'algorithme Bayes naïf

Le théorème de Bayes est utilisé afin de prédire la probabilité qu'une caractéristique donnée appartienne à une étiquette de classe particulière C, notée à l'aide de l'équation suivante :

$$P(C|f_1, ..., f_n) = (P(C) \times P(f_1, ..., f_n|C))/(P(f_1, ..., f_n)) \tag{2.1}$$

Avec :

- $f_1, f_2, ..., f_n$ sont les caractéristiques du document.
- $P(C)$: la probabilité a priori d'une classe C,
- $P(f_1, ..., f_n|C)$: la probabilité a posteriori qu'un ensemble de caractéristiques donné soit classé comme une classe.
- $P(f_1, ..., f_n)$: la probabilité d'une caractéristique donnée. En utilisant la propriété naïve qui suppose que toutes les caractéristiques sont indépendantes.

L'équation indiquée ci-dessus peut être récrite comme suit :

$$P(C|f_1, ..., f_n) = (P(C) \times P(f_1|C) \times P(f_2|C) \times ... \times P(f_n|C))/P(f_1, ..., f_n) \tag{2.2}$$

2.4.3 Machine à vecteurs de support (SVM)

Les machines à vecteurs de support ou séparateurs à vaste marge (en anglais support vector machine, *SVM*) sont un ensemble de techniques d'apprentissage supervisé destinées à résoudre des problèmes de classification et de régression. Ils sont depuis longtemps reconnus comme capables de traiter efficacement des données de grandes dimensions. Conçu à l'origine comme un classificateur à deux classes, le *SVM* peut fonctionner avec plus de classes en créant plusieurs classifications binaires.

L'algorithme fonctionne en classant les instances sur la base d'une fonction linéaire des caractéristiques. Le classificateur est alimenté avec des instances pré-étiquetées et en sélectionnant des points comme vecteurs de support, le *SVM* recherche un hyperplan qui maximise la marge [2]. La figure 2.3 illustre la construction géométrique d'un hyperplan optimal pour un espace d'entrée bidimensionnel.

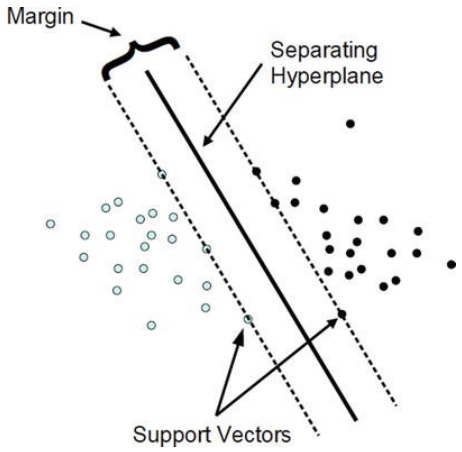


FIGURE 2.3 – Fonctionnement d'un classificateur SVM

2.4.4 Réseaux de neurones artificiels

Les réseaux de neurones artificiels sont des techniques populaires d'apprentissage automatique qui simulent le mécanisme d'apprentissage dans des organismes biologiques. Ce mécanisme contient des unités de calcul appelées neurones [16]. Les unités de calcul sont reliées les unes aux autres par des poids, qui jouent le même rôle que les points forts des connexions synaptiques chez les organismes biologiques. Chaque entrée dans

un neurone est mise à l'échelle avec un poids, ce qui affecte la fonction calculée à cette unité. Cette architecture est illustrée dans la figure 2.4.

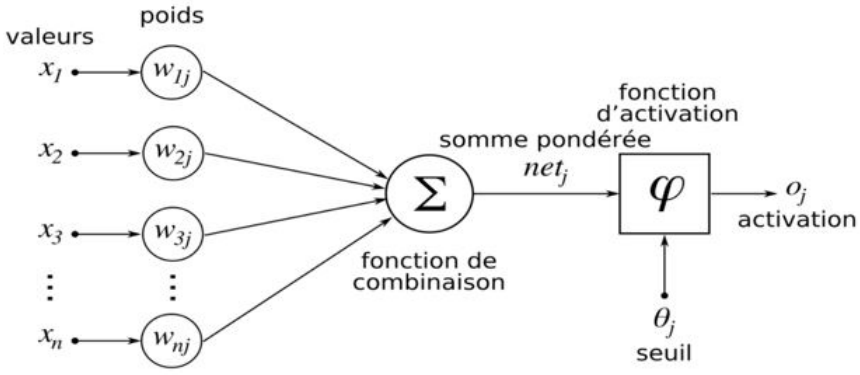


FIGURE 2.4 – Structure d'un neurone artificiel

Un réseau de neurones artificiel calcule une fonction des entrées en propageant les valeurs calculées des neurones d'entrée vers le ou les neurones de sortie et en utilisant les poids comme paramètres intermédiaires. L'apprentissage se produit en modifiant les poids reliant les neurones. Tout comme des stimuli externes sont nécessaires à l'apprentissage dans des organismes biologiques, les données d'apprentissage contenant les exemples de paires entrée-sortie de la fonction à apprendre constituent le stimulus externe dans les réseaux de neurones artificiels.

Par exemple, les données d'apprentissage peuvent contenir des représentations en pixels d'images en entrée et de leurs étiquettes annotées (par exemple, orange, pomme) en sortie. Ces paires de données d'apprentissage sont introduites dans le réseau de neurones en utilisant les représentations en entrée pour effectuer des prédictions sur les étiquettes en sortie. Les données d'apprentissage servent pour estimer les poids du réseau de neurones, en fonction de la correspondance entre la sortie prédite (par exemple, la probabilité d'une orange) pour une entrée particulière et l'étiquette de sortie annotée dans les données d'apprentissage.

2.4.5 Arbres de décision

Les arbres de décision constituent une méthode efficace de fouille de données, en vue de la prédiction d'une variable à l'aide d'autres variables qualitatives ou quantitatives.

Cette flexibilité constitue un avantage par rapport à certains outils de classification, prévus pour des prédicteurs d'un seul et même type. Il s'agit d'une méthode itérative, dite de partitionnement récursif des données. En effet, la méthode construit des classes d'individus, les plus homogènes possible, en posant une succession de questions binaires (de type oui/non) sur les attributs de chaque individu.

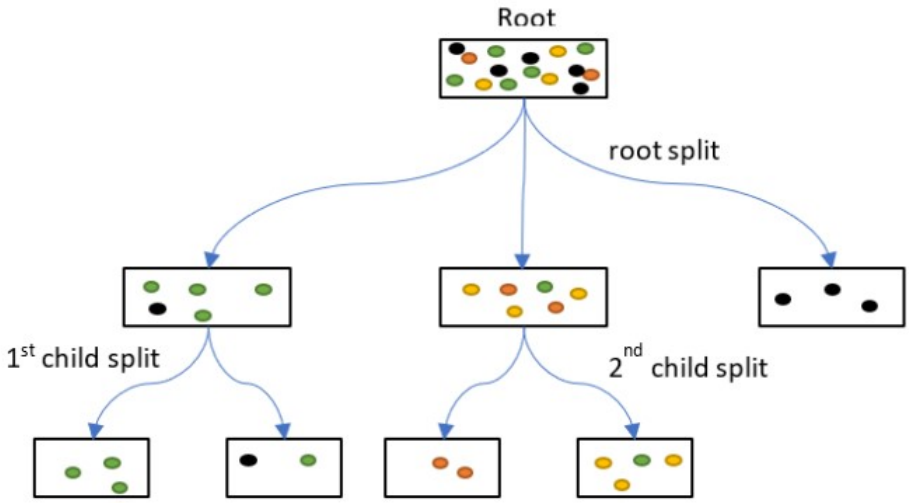


FIGURE 2.5 – Schéma d'un arbre de décision [1]

Contrairement à beaucoup de techniques de classification comme la régression logistique, le *SVM*, etc., les arbres de décision sont extrêmement intuitifs et fournissent une représentation graphique parlante et facile à interpréter. Les arbres de décision sont formés en transmettant les données d'un nœuds racine aux feuilles. Les données sont fractionnées à plusieurs reprises en fonction de variables choisies afin que les nœuds enfants soient plus homogènes en termes de classes. Ce processus est illustré dans la figure 2.5. Les arbres de décision permettent donc d'identifier très rapidement les variables les plus discriminantes d'un ensemble de données, en fonction de leur présence parfois répétée le long des nœuds.

1. https ://www.displayr.com/how-is-splitting-decided-for-decision-trees/

2.4.6 Forêts aléatoires

Les forêts aléatoires (en anglais *Random Forest*) ont été formellement proposées en 2001 comme une extension des arbres de décision [17]. Au plus haut niveau, une forêt aléatoire est une collection d'arbres de décision générés de manière légèrement différente à partir du même ensemble de données, qui votent collectivement pour offrir un meilleur modèle qu'un arbre individuel.

Les arbres d'une forêt aléatoire sont générés en sélectionnant aléatoirement un sous-ensemble d'enregistrements avec remplacement (technique appelée *bagging*) et en sélectionnant aléatoirement un sous-ensemble des attributs, afin que la forêt soit composée d'arbres de décision légèrement différents. Cette méthode introduit de petites variations dans les arbres qui sont créés dans la forêt aléatoire. L'ajout de cette dose de variance contrôlée permet d'améliorer la valeur prédictive de l'algorithme. Considérons une tâche de classification de m individus $x_i (i = 1, m)$ et n attributs. Un arbre de décision (noté DT) dans la forêt de k arbres (noté $RF = DT_i, i = 1, k$) est créé de la façon suivante :

- tirage avec remise depuis l'ensemble d'apprentissages d'un échantillon *bagging* (noté $B_i, i = 1, k$) qui est utilisé pour la construction de l'arbre ;
- recherche d'une meilleure coupe pour chaque nœuds de décision à partir d'un sous-ensemble aléatoire d'n' attributs ($n' < n$, p. ex. $n' = \sqrt{n}$) ;
- construction de l'arbre le plus profond possible.

2.5 L'apprentissage profond (Deep Learning)

L'apprentissage profond fait partie d'une famille plus large de méthodes d'apprentissage automatique basées sur l'apprentissage de représentations de données. L'apprentissage peut être supervisé, semi-supervisé ou non supervisé. Les algorithmes d'apprentissage profond constituent un axe de recherche prometteur pour l'extraction automatisée de représentations de données complexes à des niveaux d'abstraction élevés. Ces algorithmes développent une architecture hiérarchisée basée sur les réseaux de neurones. Les couches du réseau forment une représentation des données, dans laquelle les caractéristiques plus abstraites (couche supérieure) sont définies en fonction de caractéristiques moins abstraites (couches inférieures).

Le concept principal des algorithmes d'apprentissage profond est l'automatisation de l'extraction des caractéristiques [18, 19, 20]. Ces algorithmes utilisent une énorme quantité de données étiquetées pour extraire automatiquement une représentation complexe de données. Ces algorithmes sont largement motivés par le domaine de l'intelligence artificielle, qui a pour objectif général de reproduire la capacité du cerveau humain à observer, analyser, apprendre et prendre des décisions, en particulier pour des problèmes extrêmement complexes.

Les algorithmes d'apprentissage profond sont très utiles lorsqu'il s'agit d'apprendre des classes d'une grande quantité de données [21]. Des études empiriques ont montré que les représentations de données obtenues en empilant des extracteurs de caractéristiques non linéaires, comme dans l'apprentissage profond, donnent souvent de meilleurs résultats d'apprentissage automatique. Les solutions d'apprentissage profond ont donné des résultats remarquables dans différentes applications d'apprentissage automatique, y compris la reconnaissance vocale [22, 23], la vision par ordinateur [22, 24, 21, 25] et le traitement du langage naturel [26, 27].

2.6 Classification de documents multimédias

La classification de documents est le processus d'attribution automatique d'étiquettes (ou de classes) pour des documents à partir d'un ensemble d'étiquettes. Ces derniers peuvent être textuels, visuels ou les deux à la fois. La figure 2.6 montre plusieurs documents (A à H), qui sont passés par un système de classification de document, représenté ici par une boîte noire. On peut voir que chaque document est affecté à une classe ou une catégorie spécifique que nous avions définie précédemment. L'apprentissage de ces classes est généralement réalisé avec un modèle de classification. Ce modèle est utilisé pour prédire les étiquettes de classes d'un ou de plusieurs exemples de documents avec des étiquettes inconnues. La plupart des techniques de classification comportent généralement deux phases :

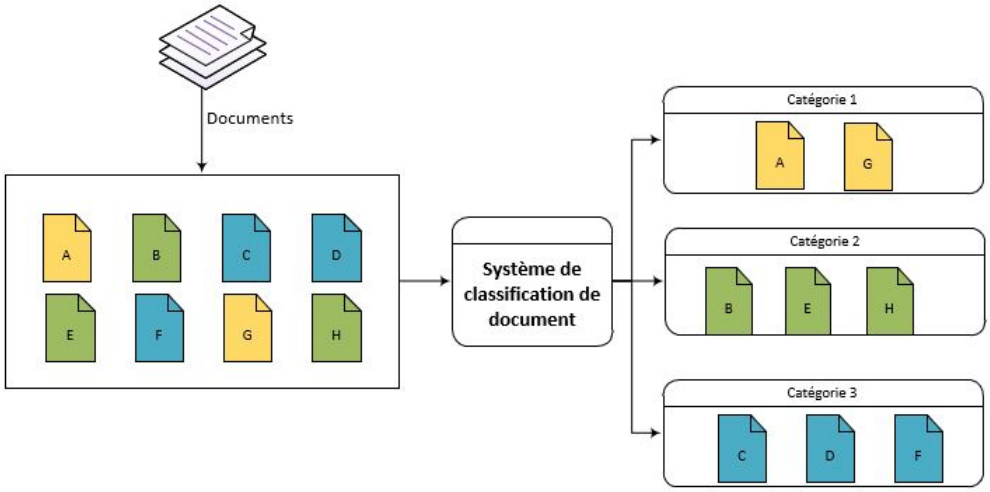


FIGURE 2.6 – Vue d'ensemble conceptuelle de la classification de documents

1. **Phase d'entrainement** : Dans cette phase, un modèle d'entrainement est construit à partir des instances d'entrainement. Intuitivement, cela peut être compris comme un modèle mathématique récapitulatif des classes étiquetées dans l'ensemble de données d'entrainement.
2. **Phase de test** : dans cette phase, le modèle d'entrainement est utilisé pour déterminer l'étiquette de classe d'une ou de plusieurs instances de test non identifiées.

Les étapes de la classification automatique de documents sont décrites dans la figure 2.7. Le processus de classification est précédé par une phase de prétraitement qui consiste à réduire les bruits causés par les conditions d'acquisition des données et d'éliminer les données inutiles. Ensuite, on trouve l'étape d'extraction et de sélection des caractéristiques qui permet d'identifier les caractéristiques, à partir desquelles on peut discriminer les différentes classes. Enfin, le classificateur est formé en utilisant les données d'apprentissage et il est évalué en utilisant les résultats de prédiction de classes pour les données de test.

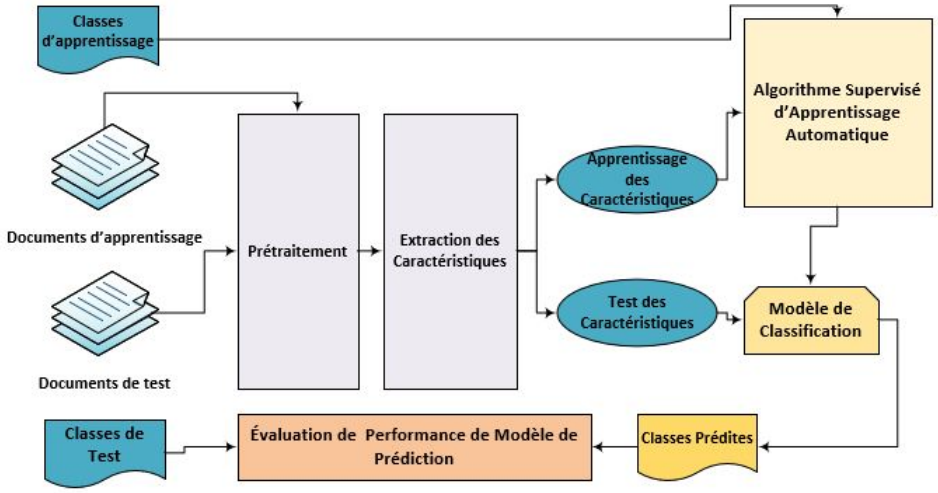


FIGURE 2.7 – Étapes de la classification automatique de documents

Le processus peut être itératif dans la mesure où l'évaluation peut conduire à un ajustement des données, des caractéristiques, des paramètres du modèle ou de la formation afin d'obtenir de meilleures performances.

2.7 Classification de documents textuels

La classification de documents textuels est un problème important dans le domaine de la fouille de texte. Son but est de trouver des règles de classification à partir d'un corpus d'entrainement connu pour créer un modèle d'apprentissage automatique dont l'objectif est de classer de nouveaux documents textuels avec une bonne précision. Les données textuelles sont souvent non structurées, mais appartiennent généralement à un langage spécifique suivant une syntaxe et une sémantique spécifique. Toute donnée textuelle, un mot simple, une phrase ou un document se rapporte à un langage naturel. Pour comprendre l'analyse du texte et le traitement du langage naturel, nous devons comprendre ce qui rend un langage naturel.

En termes simples, un langage naturel est celui qui est développé et évolué par les humains à travers l'utilisation naturelle et la communication plutôt que construit et créé artificiellement comme un langage de programmation informatique. Les langues humaines comme le français, l'anglais et l'arabe sont des langues naturelles. Les langues naturelles

peuvent être communiquées sous différentes formes, y compris la parole, l'écriture, ou même des signes. L'objectif général du traitement automatique du langage naturel est de parvenir à une meilleure compréhension de ce dernier par l'utilisation d'ordinateurs [28]. D'autres incluent aussi l'emploi des techniques simples pour un traitement rapide des données textuelles [29]. En outre, les techniques d'analyse linguistique sont utilisées entre autres pour le traitement du texte.

2.7.1 Prétraitement du texte

Dans la grande majorité des tâches de la fouille de texte, les mots individuels (souvent appelés termes ou jetons) sont les principales caractéristiques utilisées en entrée des algorithmes de classification. La création d'entités à partir de texte nécessite l'application d'actions de prétraitement de texte standard. Les étapes les plus courantes pour le prétraitement du texte sont :

Nettoyage de texte : Souvent, les données textuelles que nous voulons utiliser ou analyser contiennent beaucoup de caractères et des symboles inutiles qui doivent être supprimés avant d'effectuer d'autres opérations comme la segmentation ou d'autres techniques de prétraitement. Cela comprend l'extraction de textes significatifs à partir de sources de données comme les données *HTML*, qui consistent en des balises *HTML* inutiles, ou même des données provenant de flux *XML*.

Conversion de casse : Cette opération consiste à transformer entièrement, dans un document, tous les caractères qui sont majuscules en minuscules. Cette étape est nécessaire afin d'éviter toute confusion entre les mots similaires qui diffèrent en minuscules ou en majuscules.

Segmentation de texte : Les techniques de segmentation les plus populaires sont la segmentation de phrases et de mots, qui sont utilisés pour décomposer un corpus de texte en phrases, et chaque phrase en mots. Ainsi, la segmentation peut être définie comme le processus de décomposition ou de fractionnement des données textuelles en composantes significatives plus petites appelées jetons (tokens). Habituellement, la segmentation de texte est faite avant ou après avoir supprimé des caractères et des symboles inutiles. Ce choix dépend de la nature du problème à résoudre et les données dont on a besoin pour appliquer les techniques de fouille de textes.

Suppression des caractères spéciaux : Une tâche importante dans la normalisation du texte consiste à éliminer les caractères spéciaux et inutiles. Ceux-ci peuvent être des symboles spéciaux ou même des ponctuations qui se produisent dans les phrases. Cette étape est souvent effectuée avant ou après la segmentation du texte. La raison principale de cela est que souvent la ponctuation et les caractères spéciaux n'ont pas beaucoup de signification lorsque nous analysons le texte pour extraire des caractéristiques ou des informations basées sur le traitement de langage naturel et l'apprentissage automatique.

Suppressions des mots vides (Stopwords) : Les mots qui ont peu ou pas de signification. Ils sont habituellement retirés du texte pendant le prétraitement afin de conserver les mots ayant un maximum de signification et de contexte. Ils sont généralement des mots qui ont une fréquence très élevée par rapport aux autres mots après une segmentation par mots. Des mots comme : *et, le, la, du,* etc. sont des mots vides. Il n'existe pas une liste universelle de mots vides. Chaque domaine de recherche peut avoir son propre ensemble de mots vides.

Lemmatisation : La lemmatisation est le processus de réduction des formes fléchies ou parfois les formes dérivées d'un mot à sa forme de base afin qu'ils puissent être analysés comme un seul terme. L'utilisation de cette étape dépend du contexte de l'application. La lemmatisation cherche une forme canonique d'un mot. Le lemme est un mot de base, comme un verbe à l'infinitif, un adjectif ou un mot au singulier masculin. Selon Plisson [30], la lemmatisation joue un rôle important au cours de l'étape de prétraitement du document dans de nombreuses applications de la fouille de textes. Outre son utilisation dans le domaine du traitement du langage naturel et de la linguistique, il est également utilisé pour générer des mots clés génériques pour les moteurs de recherche ou des étiquettes pour les cartes conceptuelles.

2.7.2 Représentation du texte

Pendant de nombreuses années, la classification des données textuelles a été considérée comme une tâche pratique et nécessaire de la fouille de textes. Afin d'améliorer l'exécution d'une tâche aussi importante, nous avons toujours besoin d'une méthode informative et expressive pour représenter les textes [31, 32]. À cet égard, si nous considérons les mots comme les plus petites unités d'information d'un texte, il existe une

variété de mesures d'information quantitatives bien connues pouvant être utilisées pour représenter un texte.

Modèle de sac à mots (Bag of Words : BOW)

Le modèle de sac de mots [33] est une représentation par laquelle un texte est décrit par l'ensemble de ses mots, dit vocabulaire, sans tenir compte de l'ordre des mots ou de la grammaire. Il est utilisé dans la classification de documents textuels où l'occurrence de chaque mot est utilisée comme caractéristique pour former un classificateur. Après avoir développé les vecteurs pour chaque document textuel, les termes sont pondérés. La méthode de pondération des termes la plus courante est *TF-IDF* (en anglais *Term Frequency-Inverse Document Frequency*). C'est l'une des premières méthodes proposées dans le domaine de la recherche d'information.

Cette méthode consiste à transformer un document en un vecteur composé des fréquences des mots contenus dans un sac de mots. Pour chacun des mots contenus dans ce vocabulaire, cette méthode détermine son nombre d'occurrences au sein du document appelé fréquence du mot (*Term-Frequency TF*). Une fois normalisée, cette valeur est comparée à la fréquence inverse du document (*Inverse Document Frequency IDF*). La rareté d'un mot au sein d'une collection des documents (le corpus) est mesurée par *IDF*. En effet, un mot peut avoir un *IDF* nul s'il est apparu dans tous les documents d'un corpus. Inversement, un mot est rarement présent dans les documents, aura un *IDF* élevé. Pour un terme t_i contenu dans un document d_j du corpus C, le *TF-IDF* de t_i est donné par :

$$tf(t_i, d_j) = 1 + \frac{f(t_i, d_j)}{|d_j|}, \tag{2.3}$$

$$idf(t_i) = log\frac{|C|}{n(t_i)}, \tag{2.4}$$

$$n(t_i) = |d_j \in C : t_i \in d_j| \tag{2.5}$$

Avec $tf(t_i, d_j)$ la fréquence du terme t_i au sein du document d_j, $|C|$ le nombre de documents contenus dans le corpus C, et $n(t_i)$ le nombre de documents où le terme t_i apparaît.

Après avoir calculé les deux indicateurs du pouvoir discriminant TF et IDF de chacun des mots contenus dans chaque document composant le corpus, une matrice A terme par document est obtenue par la multiplication de $TF \times IDF$. Les colonnes de celle-ci représentent le *TF-IDF* de chaque mot du document. Ainsi, l'espace de représentation du document est de taille fixe et connue, et cette taille est le nombre de mots contenus dans le vocabulaire V [34]. Soit $(a_{i,j})$ les composants de la matrice A :

$$a_{i,j} = tf(t_i, d_j) \times idf(t_i)$$

Salton et al. [35] ont étudié le pouvoir discriminant des mots au sein d'un corpus, et ils ont proposé une méthode pour le quantifier. Pour chaque document d'un corpus représenté par un vecteur, ayant comme dimension la taille de sac de mots, il est possible de mesurer dans un espace la distance entre deux documents en calculant leur similarité. Deux documents seront considérés comme proches dans l'espace vectoriel terme-document si la mesure de similarité entre ces deux documents est élevée et, dans le cas contraire, si la distance qui les sépare est élevée, ces documents partageant peu d'information. Alors, un terme est caractérisé par un fort pouvoir discriminant si son apparition dans un document permet de diminuer la similarité de ce dernier avec les autres documents du corpus. Au contraire d'un terme qui est caractérisé par un pouvoir discriminant faible, la similarité du document, dans lequel il apparaît, avec les autres documents du corpus augmente.

Word2vec

Word2vec [36], est un groupe de modèles utilisés pour produire des *Word embeddings* qui est un ensemble de techniques de représentation du langage et d'apprentissage de fonctions dans le traitement du langage naturel (NLP). Dans ce modèle, des mots ou des phrases du vocabulaire sont mappés à des vecteurs de nombres réels qui sont plus représentatifs du sens sémantique.
Les modèles *Word2vec* sont des réseaux de neurones à deux couches peu profondes formés pour reconstruire les contextes linguistiques des mots. Ils prennent en entrée un grand corpus de texte et produisent un espace vectoriel, généralement de plusieurs centaines de dimensions, avec chaque mot unique dans le corpus étant assigné un vecteur correspondant dans l'espace. Les vecteurs de mots sont positionnés dans l'espace vectoriel de

telle sorte que les mots partageant des contextes communs dans le corpus soient situés à proximité les uns des autres dans l'espace.

2.7.3 Sélection des caractéristiques

La sélection de caractéristiques est un domaine de recherche actif depuis les années 1970 [37]. Il a un impact important sur la fouille de données en général et la fouille de texte en particulier [38]. Plusieurs techniques de sélection des caractéristiques ont été développées en vue de réduire la dimension de l'espace vectoriel. Chacune de ces techniques utilise des critères lui permettant de rejeter les termes non pertinents à la tâche de classification. On obtient alors un vocabulaire réduit, des textes représentés par des vecteurs de moindre dimension, un temps de calcul plus abordable et une précision de classification accrue [39, 40]. La sélection de caractéristiques à deux principaux approche [41] :

Méthode de Filtrage : opèrent directement sur le jeu de données et fournissent une pondération, un classement ou un ensemble de variables en sortie. Ces méthodes ont l'avantage d'être rapides et indépendantes du modèle de classification, mais au prix de résultats inférieurs.

FIGURE 2.8 – Modèle de la méthode de Filtrage

La figure 2.8 décrit l'approche par filtrage. On peut voir que le processus de sélection des sous-ensembles de caractéristiques est effectué d'abord ; ensuite, le sous-ensemble sélectionné d'entités est utilisé pour évaluer le modèle de classification. Dans ce cas, la sélection de sous-ensembles de caractéristiques est effectuée séparément et il n'a pas d'interaction avec les modèles de classification.

Les méthodes de filtrage suppriment les variables les moins intéressantes. Les autres variables feront partie d'une classification ou d'un modèle de régression utilisé pour classer ou prédire des données. Ces méthodes sont particulièrement efficaces en temps de

calcul et robustes au surapprentissage [42]. Cependant, les méthodes de filtrage ont tendance à sélectionner des variables redondantes, car elles ne prennent pas en compte les relations entre les variables. Par conséquent, ils sont principalement utilisés comme une méthode de prétraitement.

Les méthodes enveloppes (wrapper methods) [43] : effectuent une recherche dans l'espace des sous-ensembles de variables, guidées par le résultat du modèle, par exemple les performances en validation croisée sur les données d'apprentissage. Elles ont souvent de meilleurs résultats que les méthodes de filtrage, mais au prix d'un temps de calcul plus important et un risque plus élevé de surapprentissage.

La figure 2.9 décrit la méthode enveloppe. L'algorithme de sélection de sous-ensembles de caractéristiques fonctionne comme une enveloppe autour de l'algorithme d'induction [44, 45]. Il utilise une technique de recherche heuristique pour générer un sous-ensemble différent de fonctionnalités ; et ensuite, un algorithme d'induction est utilisé pour évaluer chaque sous-ensemble de caractéristiques séparément [46, 44]. Le sous-ensemble de caractéristiques ayant la plus forte évolution est sélectionné en tant que meilleur sous-ensemble de caractéristiques et est ensuite utilisé pour construire le modèle de classification. Ensuite, un ensemble de données séparé qui n'était pas impliqué dans la sélection du meilleur sous-ensemble des données est utilisé pour évaluer le modèle de classification [45].

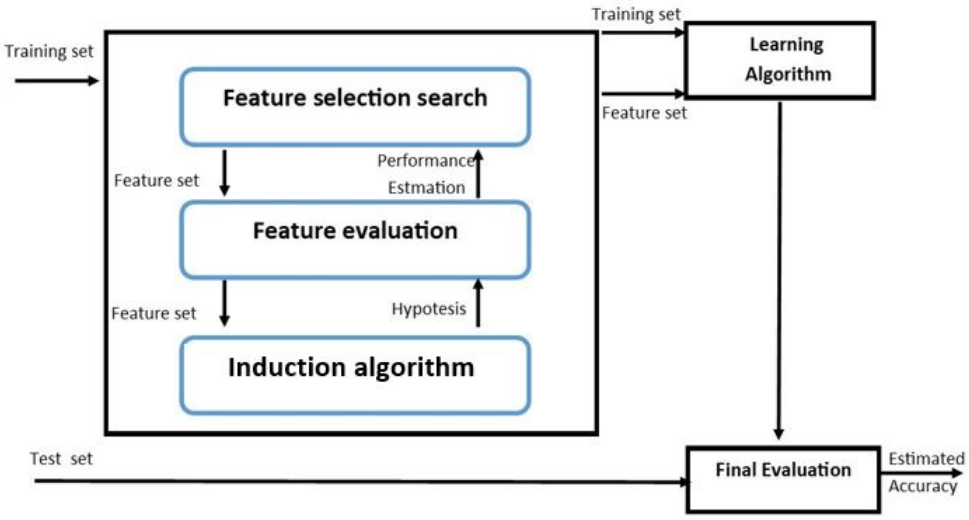


FIGURE 2.9 – Méthode enveloppe

2.8 Classification de documents visuels

La classification des images est l'une des technologies clés de la vision par ordinateur. Il existe une large gamme d'applications telles que la classification d'images basée sur le contenu, la reconnaissance d'objets et de scènes, l'exploration de données vidéo et la surveillance vidéo intelligente [47, 48, 49]. Au cours des dernières années, un grand nombre d'approches de classification classiques ont été proposées pour la représentation et la classification d'images [50, 51, 52]. La plupart de ces méthodes sont basées sur les caractéristiques telles que *Scale invariant feature transform SIFT* [53], *histograms of gradients HOG* [54], *GIST* [55], *local binary patterns LBP* [56], etc., pour générer des mots de type sac de mots visuels (Bag of Visual Word BoVW) pour la reconnaissance d'objets [57, 58].

Cependant, il existe des milliers de points-clés *SIFT* dans une image, ce qui prend beaucoup de temps pour entrainer le modèle de classification. En outre, la précision de la classification est affectée par les bruits. Dans [57], les auteurs ont proposé une méthode pour améliorer la qualité des mots visuels. L'idée est de combiner les points-clés le plus proches et de supprimer la catégorie qui a une fréquence de document élevée et une association statistique faible avec toutes les autres catégories de corpus. Lu et Wang [59] ont développé une factorisation matricielle sémantique basée sur la régulari-

sation Laplacienne afin d'améliorer l'efficacité au cours de la phase d'entrainement des mots visuels. Bien que ces méthodes aient grandement contribué à la classification des images, leurs performances sont encore loin d'être satisfaisantes. La raison peut en être que les caractéristiques visuelles extraites par les techniques classiques ne peuvent pas exprimer efficacement la sémantique de l'image.

Cependant, en raison du fossé sémantique bien connu et des caractéristiques hétérogènes des images, telles que l'échelle et la perspective universelle, la déformation des objets, le fond encombré, l'éclairage, etc., il est très difficile de représenter un concept de haut niveau par des caractéristiques faibles extraites de pixels d'image. Ces techniques d'extraction ont limité les performances de reconnaissance d'image, ce qui a fait de la classification des images l'un des problèmes les plus difficiles en matière de vision par ordinateur. Récemment, des progrès importants ont été réalisés dans les tâches de reconnaissance visuelle et la classification des images en raison du développement de réseau de neurones convolutionels *CNN* [60, 48].

2.8.1 Caractéristiques visuelles pour la représentation d'images

La caractéristique visuelle de l'image est l'objectif inhérent de l'image elle-même, qui peut refléter le contenu de l'image. Au cours des dernières décennies, les caractéristiques d'images les plus utilisées sont la caractéristique de couleurs, de textures, de formes, *SIFT* et *HOG*.

SIFT (Scale-invariant feature transform)

Lowe [53] a développé l'algorithme *SIFT* pour extraire des caractéristiques invariantes à l'échelle, à la rotation et à l'éclairage changeant de l'image.

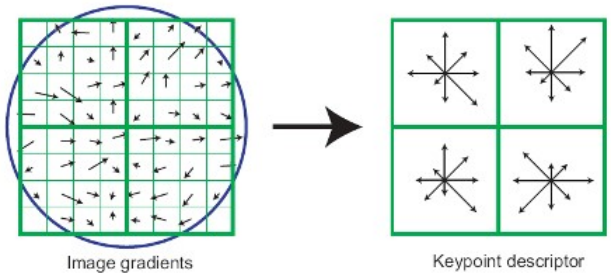


FIGURE 2.10 – Illustration visuelle de SIFT

La transformation *SIFT* regroupe un détecteur et un descripteur de caractéristiques. Le détecteur extrait d'une image un certain nombre de points d'intérêt (régions attribuées) d'une manière compatible avec certaines variations de l'éclairage, du point de vue et d'autres conditions de visualisation. Le descripteur associe aux points une signature qui identifie leur apparence de manière compacte et robuste.

Les points-clés donnés par le détecteur *SIFT* à des échelles et orientations particulières sont introduits en tant qu'entrée dans le code descripteur. Ce dernier calcule les vecteurs descripteurs pour ces points-clés de manière à ce qu'ils soient hautement distinctifs et partiellement invariants des variations, comme l'emplacement de l'image, l'échelle et la rotation, l'éclairage, le point de vue 3D, etc.

Un descripteur de points-clés est créé en calculant d'abord la magnitude et l'orientation du gradient à chaque point d'échantillon d'image dans une région située autour de l'emplacement du point-clé, comme indiqué à gauche. Celles-ci sont pondérées par une fenêtre gaussienne, indiquée par le cercle superposé. Ces échantillons sont ensuite accumulés dans des histogrammes d'orientation résumant le contenu de sous-régions 4×4, comme indiqué à droite, la longueur de chaque flèche correspondant à la somme des magnitudes de gradient proches de cette direction dans la région. La figure 2.10, montre un tableau de descripteurs 2×2 calculé à partir d'un ensemble d'échantillons 8×8.

Histogrammes des Orientations de gradients (HOG)

Dalal et Triggs [54] ont à l'origine développé le descripteur *HOG* pour la détection des piétons dans les images en niveaux de gris. Les caractéristiques *HOG* ont depuis été

utilisées pour la reconnaissance d'autres catégories d'images, telles que les voitures [61] les bicyclettes [62] et les expressions faciales [63]. L'extraction de caractéristiques *HOG* comprend quatre étapes principales. Tout d'abord, une image d'entrée est normalisée par la loi de puissance et les gradients d'image sont calculés dans les directions horizontales et verticales. Ensuite, l'image est divisée en cellules, une cellule peut être une région rectangulaire ou circulaire.

Dans la troisième étape, les histogrammes pour plusieurs orientations sont calculés pour chaque cellule, chaque pixel de la cellule contribuant à un score pondéré à un histogramme. Enfin, les histogrammes de cellules sont normalisés et regroupés en blocs pour former les caractéristiques *HOG*. Un exemple de dispositifs de *HOG* est représenté sur la figure ci-dessous, qui illustre la solidité des caractéristiques moyennes du *HOG* à chaque emplacement de pixel.

La figure 2.11, présente une démonstration de la méthodologie d'extraction de caractéristiques HOG. Pour capturer les informations de dégradé local, l'image d'entrée (voir figure 2.11 (a)) est ensuite divisée en plusieurs sous-blocs superposés dans les directions verticales et horizontales. Sur la base des images de dégradé vertical et horizontal, les directions des caractéristiques de l'image à chaque pixel sont ensuite calculées, comme illustrer à la figure 2.11 (b). Dans chaque sous-bloc, une caractéristique d'histogramme est accumulée (voir figure 2.11 (c)), en fonction de l'intensité du gradient et de la direction de chaque pixel du sous-bloc illustré à la figure 2.11.

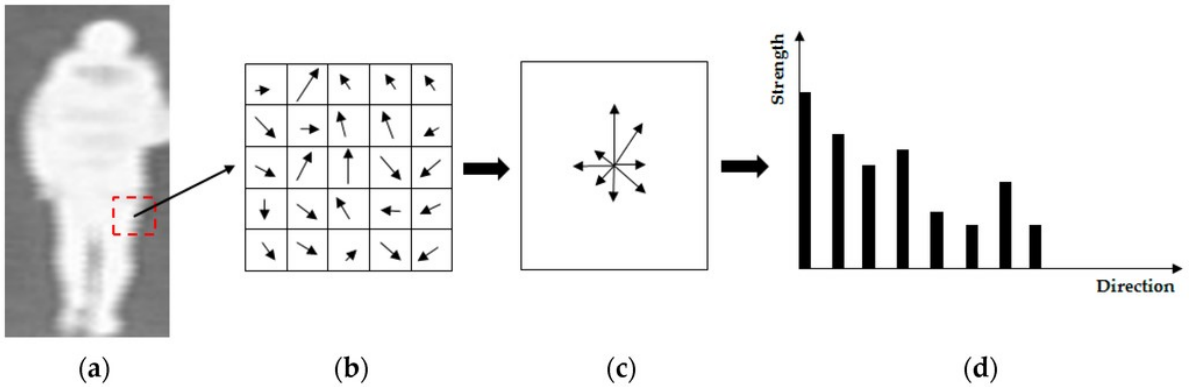


FIGURE 2.11 – Démonstration de la méthode d'extraction des caractéristiques HOG[2]

Réseaux de neurones convolutionels (CNN)

La classification des images, qui vise à attribuer une catégorie sémantique aux images, a fait l'objet de nombreuses études au cours des dernières années. Plus récemment, les réseaux de neurones convolutionels (*CNN*) [58] ont atteint des résultats très prometteurs. Comparé aux techniques classiques d'extraction de caractéristiques telles que *SIFT* [53], HOG [54], *GIST* [55], *LBP* [56] et autres, le *CNN* peut extraire automatiquement les caractéristiques d'une image, sans nécessiter de conception manuelle. En tant que modèle d'apprentissage profond représentatif, il peut fournir de bonnes caractéristiques pour représenter les objets dans l'image. Ainsi, il a fait un grand progrès dans la classification d'image et la détection d'objet.

Le *CNN* est un réseau *feed-forward* c'est-à-dire que le flux d'information se fait dans une seule direction, de son entrée à sa sortie. Tout comme le réseau de neurones artificiel (*ANN*) est biologiquement inspiré, les *CNN* le sont aussi. Le cortex visuel dans le cerveau, constitué de couches alternées de cellules simples et complexes [20], motive leur architecture. Les architectures *CNN* existent en plusieurs variantes ; cependant, en général, ils sont constitués de couches de convolution et de regroupement (ou de sous-échantillonnage), qui sont regroupées en modules. Une ou plusieurs couches entièrement connectées, comme dans un standard réseau de neurones *feed-forward*, suivent ces modules. Les modules sont souvent empilés les uns sur les autres pour former un modèle profond.

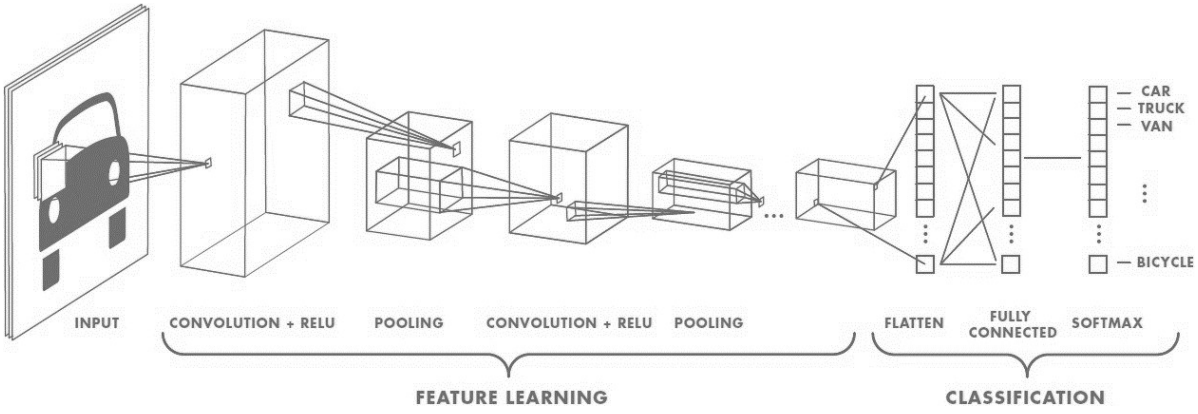


FIGURE 2.12 – Exemple d'un réseau de neurones convolutionels

2. https ://www.mdpi.com/1424-8220/16/7/1134/htm

La figure 2.12 illustre l'architecture de *CNN* pour une tâche de classification d'objets. Une image est entrée directement sur le réseau, suivie de plusieurs étapes de convolutions et d'échantillonnages. Par la suite, les représentations de ces opérations alimentent une ou plusieurs couches entièrement connectées. Enfin, la dernière couche entièrement connectée sort l'étiquette de classe. Bien que ce soit l'architecture de base la plus populaire dans la littérature, plusieurs changements d'architectures ont été proposés ces dernières années dans le but d'améliorer la précision de la classification des images ou de réduire les coûts de calcul.

Lecun [64] a proposé la première structure *CNN* (LeNet), pour la reconnaissance de caractères manuscrits. Il utilise également un système basé sur *CNN* pour identifier une voiture ou un avion [65] dans un environnement bruyant. Quelques travaux récents [66, 48, 67] ont démontré que les modèles *CNN* peuvent extraire la relation et l'information spatiale entre les couches de l'image et exprimer les caractéristiques pertinentes à l'intérieur de l'image. Avec le développement de la technologie informatique, l'expansion de *CNN* a été introduite et étudiée pour améliorer les performances.

Dans son travail, Krizhcvsky et al. [68] a réussi de réduire le taux d'erreur de classification des images. Il est passé de 26,2% à 15,3% en utilisant un réseau de neurones à convolution à 7 couches (Alexnet). L'équipe *Google* a utilisé un réseau de neurones à 22 couches (*GoogleNet*) dans le cadre du défi de la reconnaissance d'objets *ILSVRC-2014* [69] pour atteindre un taux d'erreur de 6,67% [70]. Au même moment, *Visual Geometry Group* VGG [71] a été conçu et a atteint un taux d'erreur de 7,3%. En outre, le MSRA-VCG (*Microsoft Research Institute of Visual Computing Group*) a mis au point un apprentissage résiduel approfondi [72] de *ImageNet*, qui conduit à un taux d'erreur réduit à 3,57%.

2.9 Conclusion

Nous avons présenté dans ce chapitre les techniques d'apprentissages automatiques utilisées pour la classification du texte et de l'image. L'apprentissage de caractéristiques dans la classification de documents dépend fortement de la représentation de celles-ci. Une bonne représentation de documents augmente l'efficacité et la performance des modèles d'apprentissage. Pour venir en aide à la représentation *BOW*, qui est basée

uniquement sur la fréquence de termes, on présente dans le chapitre suivant les principaux modèles thématiques qui permettent de trouver automatiquement des relations sémantiques entre les mots et les documents qui forment le corpus.

Chapitre 3

Représentation thématique de documents textuels

3.1 Introduction

Le modèle sac à mots est couramment utilisé pour représenter les documents textuels et les images. Elle tient compte du nombre d'occurrences de chaque mot dans le document. Les vecteurs BOW ont une très grande dimensionnalité. Cependant, pour analyser les concepts présents dans les documents, un espace sémantique de dimension inférieure est idéal. La réduction de dimension peut être appliquée pour trouver l'espace sémantique et sa relation avec la représentation BOW. Le modèle thématique se démarque par sa capacité de réduire la dimension de la représentation d'un document. Il est conçu pour extraire automatiquement les thèmes d'un corpus de documents textuels [73, 74, 75, 76]. Un thème est déterminé par un ensemble de termes qui apparaissent fréquemment dans les documents du corpus. En raison de la nature de l'utilisation de la langue, les termes qui constituent un thème sont souvent sémantiquement liés [1].

Les modèles thématiques ont été développés à l'origine comme moyen d'indexation, de recherche, de regroupement et de structuration de grands corpus de documents non structurés et non annotés. À l'aide de ces modèles, les documents peuvent être représentés par les thèmes qui les composent et ainsi l'ensemble du corpus peut être indexé et organisé en fonction de cette structure sémantique. En représentant les documents au moyen de thème de dimension inférieure, par opposition aux termes, les modèles

thématiques mettent en évidence des relations sémantiques latentes et permettent une analyse plus rapide du texte [76]. Il existe trois principaux techniques de modélisation thématique. Premièrement, l'analyse sémantique latente, qui utilise une technique de factorisation basée sur décomposition de valeur singulière (*SVD*). Deuxièmement, l'analyse sémantique probabiliste latente et troisièmement l'allocation de Dirichlet latente, en revanche, fournissent un cadre probabiliste pour la tâche de réduction des dimensions permettant de déterminer des thèmes abstraits dans un document.

3.2 Analyse sémantique latente (LSA)

Parmi les méthodes qui permettent la réduction de l'espace de représentation, on trouve l'analyse sémantique latente (*Latent semantic analysis LSA*). C'est une méthode d'analyse, d'indexation et d'extraction d'information permettant d'identifier des modèles et de trouver automatiquement des relations sémantiques entre les mots et les documents qui forment le corpus. La *LSA* suppose que les mots dans les documents ont une structure sémantique latente. Ainsi les mots qui ont une signification proche apparaîtront dans des documents similaires [77, 78]. La fréquence de chaque mot apparaissant dans le document est calculée. Ce nombre est normalisé en utilisant la pondération *TF-IDF* dont le but de construire une matrice des occurrences. Pour relier les documents entre eux, *LSA* transforme cette matrice en une relation entre les termes et les thèmes, et en une relation entre ces thèmes et les documents (voir section 2.7.2).

La méthode *LSA* utilise une technique mathématique connue sous le nom de décomposition de valeur singulière (*SVD*) pour réduire la dimensionnalité des données tout en préservant la structure de similarité et les informations clés présentées dans la matrice des occurrences [79]. Cette réduction de dimension permet d'estimer de façon plus fiable les similitudes entre les documents ou entre les documents et les mots. La figure 3.1 illustre le fonctionnement de *SVD*. Formellement, la décomposition en valeurs singulières d'une matrice à n lignes et m colonnes, contenant des valeurs réelles ou complexes, est une factorisation de la forme $U\Sigma V^T$, où U est une matrice réelle ou complexe , $\sum$ est une matrice diagonale rectangulaire avec des nombres réels non négatifs sur la diagonale, et V est une matrice unitaire de taille $n \times m$ réelle ou complexe.

La valeur du cosinus entre deux vecteurs colonnes dans la matrice reflète la simila-

rité entre deux documents. En effet, les documents qui ont plusieurs mots en commun sont considérés comme sémantiquement proches et vice versa. *SVD* utilise des valeurs dérivées statistiquement au lieu de mots individuels. Cette technique est capable de réaliser une compression significative dans de grandes collections de documents, tout en représentant la majeure partie de la variance dans la collection [80].

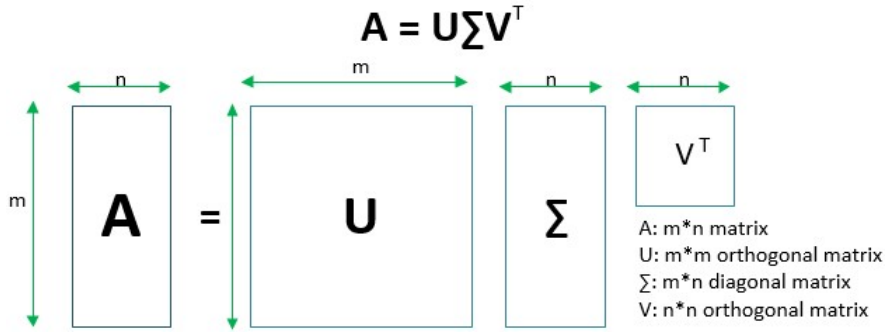


FIGURE 3.1 – Fonctionnement de la décomposition de valeur singulière SVD

Le modèle *LSA* offre certains avantages et surmonte de nombreuses limites de la méthode pondération *TF-IDF*. Ce modèle capture la synonymie, filtre certaines informations et réduit le bruit [79, 80]. Cependant, dans [81], les auteurs montrent que le modèle *LSA* ne reproduit pas convenablement la structure statistique du corpus d'apprentissage, il représente le texte comme un ensemble non ordonné de mots et ne prend pas en compte la polysémie, car un mot ne correspond qu'un seul point de l'espace sémantique. De plus, *SVD* est coûteuse en calcul et en mémoire.

3.3 Analyse sémantique latente probabiliste (PLSA)

Hofmann et al. [78] ont proposé une approche probabiliste de *LSA*, appelée modèle d'analyse sémantique latente probabiliste (*PLSA*), pour répondre aux faiblesses du modèle *LSA*. Contrairement à l'analyse sémantique latente, *PLSA* a une base statistique solide et définit un modèle génératif approprié en utilisant les concepts et les bases de la probabilité et des statistiques. L'idée principale est de construire un espace sémantique où la dimensionnalité des données n'est pas élevée.
Pour résoudre le problème de la haute dimensionnalité et pour obtenir une représentation plus réaliste du document, le modèle *PLSA* représente le document comme une matrice

de terme-document, qui est la fréquence de chaque mot distinct dans chaque document. Autre que les mots et les documents, un autre ensemble de variables est pris en compte dans ce modèle, qui est celui des thèmes. Un thème est une variable aléatoire, latente ou cachée. Le principe de *PLSA* est d'utiliser la représentation de chaque document pour extraire les thèmes et de représenter les documents comme un mélange d'entre eux [82]. En prenant en compte des observations sous forme de co-occurrences (w, d) des mots et des documents, *PLSA* modélise la probabilité de chaque co-occurrence sous la forme d'un mélange de distributions multinomiales conditionnellement indépendantes :

$$P(w|d) = \sum_z P(z)P(d|z)P(w|z) = P(d)\sum_z P(z|d)P(w|z) \tag{3.1}$$

avec z étant le thème des mots. Notez que le nombre de thèmes est un hyperparamètre qui doit être choisi à l'avance et n'est pas estimé à partir des données. La première formulation est la formulation symétrique , où w et d sont tous deux générés à partir de la classe latente z (en utilisant les probabilités conditionnelles $P(d|z)$ et $P(w|z)$ tandis que la deuxième formulation est la formulation asymétrique , où pour chaque document d, une classe latente est choisie conditionnellement au document selon $P(z|d)$ et un mot est alors généré à partir de cette classe selon $P(w|z)$.

PLSA s'appuie sur deux hypothèses. L'hypothèse du sac à mots où le couple (d, w) est généré de manière indépendante, car le document est considéré comme un sac de mots. La deuxième hypothèse est l'indépendance conditionnelle ce que signifie que les mots et les documents sont conditionnellement indépendants compte tenu du thème. Autrement dit, chacun des mots w est généré indépendamment du document d sachant la variable latente (non observée) z. Le modèle *PLSA* est illustré dans la figure 3.2 suivante :

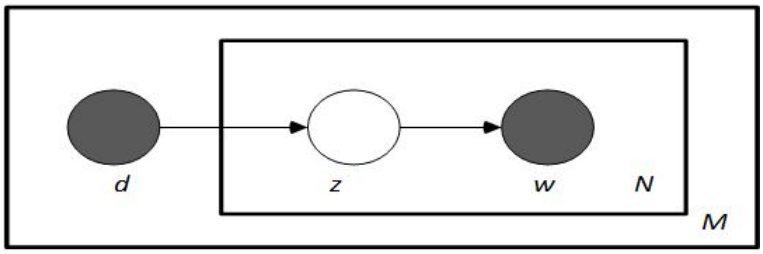


FIGURE 3.2 – Modèle PLSA

Où M est le nombre de document, N le nombre de mots dans un document, d est la variable d'index du document, z est le thème d'un mot tiré de la répartition du thème du document $P(z|d)$, et w est un mot tiré de la distribution des mots du thème de ce mot $P(w|z)$. d et w sont des variables observables, le thème z est une variable latente.

Le modèle *PLSA* résout le problème de polysémie du modèle *LSA*, cependant, il n'est pas considéré comme un modèle entièrement génératif de documents et il est connu pour son surapprentissage [1] car le nombre de paramètres augmente linéairement avec le nombre de documents.

3.4 Modèle d'allocation de Dirichlet latente

L'allocation de Dirichlet latente (*LDA*) est un puissant modèle de thème probabiliste. Il est le plus représentatif de modèles thématiques. Il a été proposé comme un modèle graphique de découverte de thèmes par Blei en 2003 [1]. Il offre une solution pour contourner les défauts du modèle *PLSA*. Ce dernier fournit une bonne base pour l'analyse de texte, mais il a deux problèmes. Premièrement, il contient un grand nombre de paramètres qui augmente linéairement avec le nombre de documents, de sorte qu'il a tendance au surapprentissage. Deuxièmement, il n'existe aucun moyen naturel de calculer la probabilité qu'un document ne figure pas dans les données d'apprentissage. *LDA* inclut un processus pour générer les thèmes dans chaque document, réduisant ainsi considérablement le nombre de paramètres à apprendre et fournissant une probabilité clairement définie pour des documents arbitraires.

Le principe du modèle *LDA* repose sur le fait que chaque document est composé de différents thèmes, ayant une distribution d'occurrence dans un corpus. Les thèmes représentent une distribution de mots spécifique où l'ordre d'apparition dans le document n'est pas pris en compte. La figure 3.3 décrit le modèle *LDA*. À gauche, on peut voir la structure de chaque thème, donnant une probabilité à chaque mot d'un vocabulaire fixe. Pour un document donné, l'histogramme à droite décrit la distribution des thèmes dans ce document.

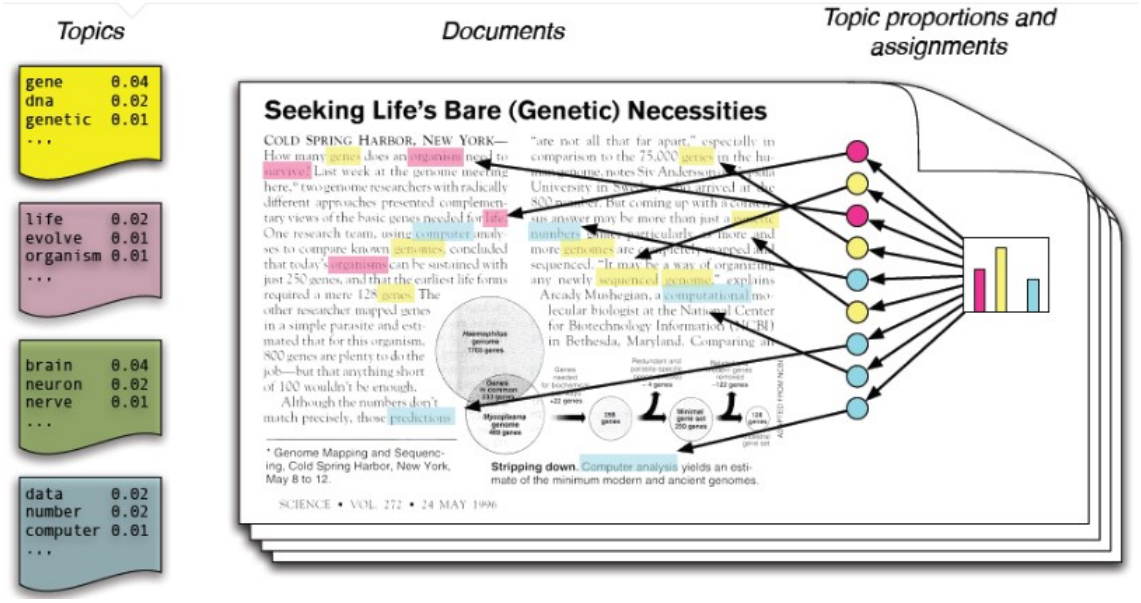


FIGURE 3.3 – Schéma décrivant LDA

3.4.1 Distribution de Dirichlet

Les documents et les thèmes composant le modèle *LDA* suivent une loi de Dirichlet. La distribution de Dirichlet [83] est une généralisation multinomiale de la loi beta [84]. Cette distribution possède une densité de probabilité, qui varie en fonction de la valeur du vecteur α, définie comme suit :

$$p(\theta|\alpha) = \frac{\Gamma(\sum_{i=1}^{k} \alpha_i)}{(\sum_{i=1}^{k} \Gamma(\alpha_i)} \theta_1^{(\alpha_i - 1)} ... \theta_k^{(\alpha_k - 1)} \tag{3.2}$$

$$\Gamma(z) = \int_0^{\infty} t^{(z-1)} e^{-t} dt \tag{3.3}$$

Cette distribution permet donc d'obtenir une distribution multinomiale de paramètre θ, correspondant pour *LDA* au mélange de thèmes d'un document d.

3.5 Processus génératif d'un document

Le modèle graphique de *LDA* est illustré dans la figure 3.4. La plaque externe représente les documents, tandis que la plaque interne représente les positions répétées des

mots dans un document donné, chacune de ces positions étant associée à un choix de thème et de mot. M désigne le nombre de documents, N le nombre de mots dans un document et K le nombre de thèmes. Les noms de variables sont définis comme suit :

α :	le paramètre préalable de Dirichlet sur les distributions des thèmes par document
β :	le paramètre préalable de Dirichlet sur les distributions de mots par thème
θ_i :	la distribution de thème pour le document i
φ_k :	la distribution des mots pour le thème k
z_i :	le thème du j ème mot du document i
w_i :	le mot spécifique

TABLE 3.1 – Définition des variables dans le modèle LDA

Le fait que W soit grisé signifie que les mots w_ij sont les seules variables observables et les autres variables sont des variables latentes. Afin d'inférer réellement les thèmes d'un corpus, les documents sont représentés sous forme de mélanges aléatoires sur des thèmes latents, chaque thème étant caractérisé par une distribution sur tous les mots. *LDA* assume le processus de génération suivant pour un corpus D consistant en M documents chacun de longueur N :

1. Échantillonnage de thèmes par $\varphi_k \sim P_{Dir}(\beta)$, $k \in [1, K]$.
2. Dans le document d , $d \in [1, M]$ effectuer un échantillonnage de la distribution de probabilité de thème de $\theta_i \sim P_{Dir}(\alpha)$.
3. Sélectionnez un thème latent $z_{ij} \sim Multinomial(\theta)$ pour le j ème mot du document d , où $j \in [1, N]$.
4. Générer un mot $w_{ij} \sim Multinomial(\varphi_{z_{ij}})$

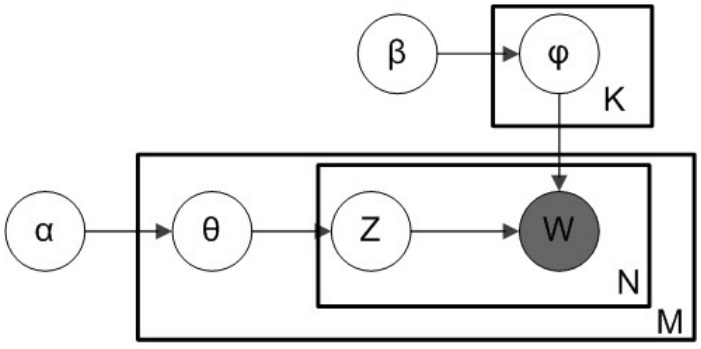


FIGURE 3.4 – Modèle graphique du LDA

3.5.1 Estimation des paramètres de LDA

Les variables et paramètres du modèle *LDA* ne sont pas connus initialement, donc il faut essayer de les apprendre à partir des données observables, c'est à dire les mots des documents. Dans la figure 3.4 on remarque que les seules variables observées sont les mots w_ij, alors que toutes les autres variables sont cachées. Étant donnés les paramètres α et β, le rôle de l'inférence est de déterminer les variables cachées θ et z_j d'un document d, étant donnée la liste des mots w_j du document. Les principales méthodes d'inférence (approchée) pour *LDA* sont :

Algorithme d'inférence bayésienne variationnelle : c'est une méthode dont les hyperparamètres z, θ et φ sont des variables latentes où les distributions a posteriori sont estimées en utilisant la distribution variationnelle bayésienne.

L'échantillonnage de Gibbs : c'est un algorithme bâti sur les méthodes de Monte-Carlo par chaînes de Markov. Il utilise un algorithme d'échantillonnage permettant d'estimer les paramètres d'un espace discret de grande dimension [85]. Facile à mettre en œuvre et capable d'extraire très efficacement des thèmes de corpus à grande échelle.

L'algorithme d'échantillonnage de Gibbs est devenu l'algorithme d'extraction le plus populaire du modèle *LDA* [86]. Son idée de base est de structurer la chaîne de Markov qui converge vers une distribution de probabilité cible et d'extraire les échantillons considérés comme proches des valeurs de la distribution de probabilité de la chaîne par échantillonnage à partir de la distribution de probabilité postérieure. La clé de l'utilisation de l'échantillonnage de Gibbs pour déduire les paramètres du modèle *LDA* est que les paramètres du modèle ne sont pas obtenus directement, mais en échantillonnant les thèmes de chaque mot indirectement [87]. En d'autres termes, les valeurs des paramètres du modèle θ et φ peuvent être obtenues indirectement en calculant leur probabilité a posteriori.

3.5.2 Nombre optimal de thèmes K

La détermination du nombre optimale de thème K dans la collection de documents est nécessaire pour la modélisation thématique. La métrique utilisée pour cet objectif est la **perplexité**. C'est une métrique d'évaluation de modèles la plus utilisée. Celle-ci

diminue lorsque le log-vraisemblance du modèle augmente. Autrement dit, un modèle qui a un pouvoir de généralisation élevé lorsqu'il possède une perplexité faible [88].

$$perplexité(B) = \exp\left(\frac{1}{N_B}\sum_{d=1}^{W} logP(w)\right) \tag{3.4}$$

Avec

$$N_B = \sum_{d=1}^{W} N_d \tag{3.5}$$

Où N_B est la longueur des N documents, W l'ensemble de termes, N_d est le nombre de mots contenus dans le document d ; $P(w)$ est la vraisemblance que le modèle génératif assigne un terme w à un document d du corpus d'évaluation. La quantité contenue dans l'exponentielle est appelée entropie des données d'évaluation sachant le modèle. En effet, afin de mettre en évidence le nombre optimal de thèmes, il faut analyser les valeurs du log-vraisemblance pour chaque valeur K testée. La vraisemblance la plus élevée indique le nombre de thèmes retenus.

3.5.3 Avantages du modèle LDA

Le modèle *LDA* est un puissant modèle bayésien de thème probabiliste basé sur un réseau bayésien à trois couches. il a une structure interne claire, et peut utiliser des algorithmes efficaces d'inférence de probabilité pour estimer ces paramètres [1, 89]. La taille de l'espace des paramètres du modèle *LDA* n'a rien à voir avec le nombre de documents d'entrainement. Par conséquent, il est plus approprié pour manipuler des corpus à grande échelle [90]. Le modèle *LDA* introduit l'hyper-paramètre au niveau document-thème, qui est meilleur que le *PLSA* [91]. Les informations a priori sont ajoutées, ce qui signifie que les paramètres peuvent être considérés comme des variables aléatoires. En outre, elle permet à *LDA* d'être un modèle hiérarchique avec une structure plus stable, évitant le surapprentissage [89].

3.6 Conclusion

Les modèles thématiques probabilistes fournissent une base probabiliste intuitive pour la réduction des dimensions. Ils nous permettent d'identifier les thèmes présents dans un document et d'exposer la probabilité de voir chaque mot dans un thème donné.

Cela facilite beaucoup l'interprétation de la signification des thèmes. Cela facilite également l'extension des modèles de manière intéressante. De nombreuses extensions de *PLSA* et de *LDA* ont été développées, à la fois pour pouvoir être appliquées à des données à grande échelle et pour incorporer une structure spéciale pour une application particulière [92].

Chapitre 4

Méthode proposée et expérimentation

4.1 Introduction

Le nombre d'articles des journaux numériques publiés sur divers sites Web a considérablement augmenté au cours des dernières années. Ces articles contiennent des informations multimodales, notamment des caractéristiques textuelles et visuelles. De nos jours, plusieurs intervenants dans le domaine de journaux numériques se sont heurtés au problème de la maîtrise du grand nombre d'articles afin d'identifier des sujets et des évènements importants dans le monde entier. Par conséquent, il existe des besoins importants de regroupement et de classification rapides et précis des articles dans un ensemble de catégories, en fonction de leur contenu sémantique, afin de faciliter l'exploration ciblée des évènements, le développement et l'expansion des répertoires Web de journaux numériques, l'analyse de la structure de leurs contenus, etc.

Bien que le texte puisse contenir une bonne partie de l'information sémantique qui peut aider à comprendre les concepts de haut niveau comme les évènements décrits dans les journaux (ex. sport, politique, faits divers, etc.), les images aussi présentent des informations complémentaires précieuses, qui peuvent aider à mieux cerner ces concepts.

There was more evidence on display Saturday as Toronto (5-4-1) dropped a 2-1 decision to the surging Philadelphia Union (7-3-2).

Spanish playmaker Alejandro Pozuelo scored on a glorious free kick but Toronto paid for defensive miscues as a Chris Mavinga own goal and Jaimiro Monteiro's 68th-minute strike gave Philadelphia the win before an announced BMO Field crowd of 26,219.

To make matter worse, the offence has been creaky of late. Including a midweek loss in Atlanta, Toronto has managed just four shots on target in its last two games.

FIGURE 4.1 – Article d'une page Web

Par exemple, la figure 4.1 présente une publication avec une image accompagnée de texte. En utilisant uniquement le texte, il est facile pour nous de dire que c'est un article de sport. Cependant, l'image fournit une information complémentaire en précisant que c'est vraiment un article du sport, et plus spécifiquement c'est du football. Cet exemple montre que lorsque l'information est disponible selon différentes modalités, ce n'est qu'en les exploitant toutes que nous pourrons obtenir une compréhension complète et claire de la scène.

4.2 Méthode de classification proposée

La méthode proposée pour la classification automatique de pages Web de journaux numériques est basée sur le modèle *LDA*. Cette approche tient compte de caractéristiques textuelles et visuelles dans le but d'exploiter de manière plus complète ces modalités présentes dans les documents. Les caractéristiques textuelles sont extraites de la partie textuelle de l'article, tandis que les caractéristiques visuelles sont générées à partir de toutes les images de l'article à l'aide de l'*API* de Google *Vision* . Cet *API* utilise *CNN* pour annoter le contenu de chaque image. La modélisation conjointe des deux modalités (texte et images) est utilisée pour former notre modèle basé sur *LDA*.

La figure 4.2 présente la structure de notre approche. Tout d'abord, toutes les données nécessaires sont collectées sous forme de texte à partir des pages Web d'actualités, ainsi que les images associées. Dans l'étape suivante, les techniques décrites dans la sec-

tion 4.5 s'appliquent aux données brutes. Les caractéristiques de chaque modalité sont traitées indépendamment. De cette manière, les caractéristiques visuelles et textuelles sont générées. Par conséquent, deux vecteurs de caractéristiques différents (un pour chaque modalité) sont formulés à l'aide de sac à mots. Ensuite, le vocabulaire extrait de chaque modalité (texte et image) est utilisé pour construire les modèles *LDA*. Dans la phase d'apprentissage, les modèles *LDA* de chaque modalité sont utilisés en tant que données d'entrées pour la construction d'un classificateur basé sur les forêts aléatoires. Le classificateur entraîné est par la suite utilisé pour prédire les catégories de documents de test.

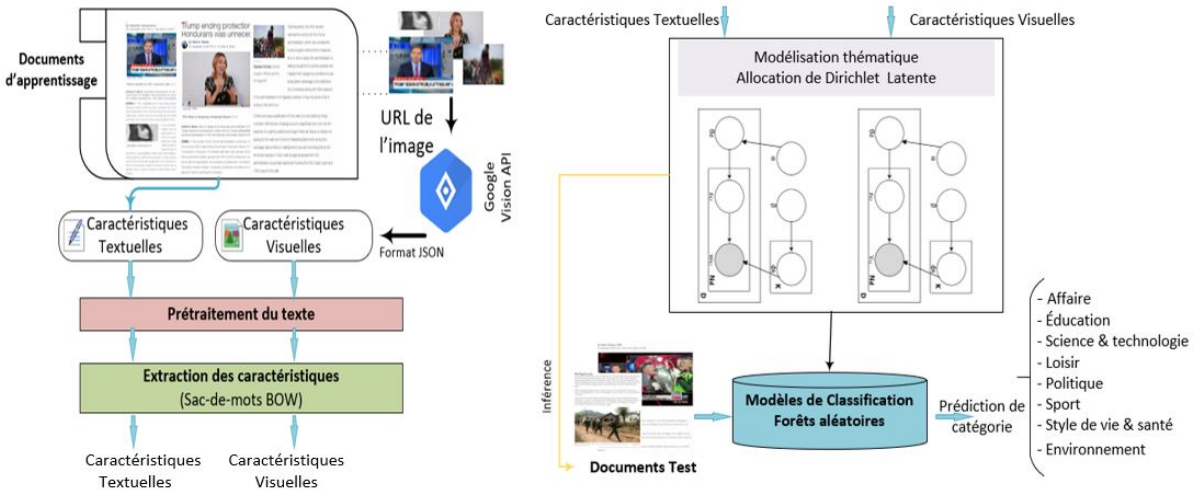


FIGURE 4.2 – Méthode proposée pour la classification automatique des pages Web d'actualités

4.3 Implémentation

Dans le but de montrer l'importance d'exploiter les caractéristiques visuelles conjointement avec les caractéristiques textuelles dans la classification de documents multimédias, nous avons construit trois modèles de classification, le premier basé sur les caractéristiques textuelles, le deuxième sur les caractéristiques visuelles et le troisième basé sur la fusion de deux modalités textuelles et visuelles. Les articles extraits, sont déjà classés dans huit catégories, *Affaire, Éducation, Loisir, Environnement, Style de vie et Santé, Politique, Science et Technologie, Sports*. Les principales étapes de notre approche sont

décrites dans les sections qui suivent. Notons que la plateforme *Matlab* a été utilisée comme langage de programmation pour le codage et l'implémentation du modèle de classification proposé en intégrant des classes programmées en *Python*.

4.4 Collection de données

Les expériences ont été réalisées sur un ensemble de données contenant les documents multimédias des cinq sites Web des journaux numériques bien connus, à savoir *CBC, BBC, CNN, The Guardian et NYTimes*. Au total, 6618 articles ont été rassemblés et sauvegardés dans une base de données *SQL*. Le tableau 4.1 indique le nombre total d'articles rassemblés par journal réparti sur huit catégories.

	Affaire	Éduc.	Loisir	Envi.	Style de vie. Santé	Politique	Sci.techno.	Sports	Total
BBC	358	139	269	0	211	254	289	1	1521
CBC	220	0	212	0	179	213	0	207	1031
CNN	20	0	30	0	0	28	0	16	94
NYTimes	104	17	139	15	96	54	52	71	548
The Guardian	249	312	620	260	459	391	476	657	3424
Total	951	468	1270	275	945	940	817	952	6618

TABLE 4.1 – Nombre d'articles par journal selon leurs catégories

De nombreux projets d'analyse de données et d'apprentissage automatique nécessitent de recueillir les données à partir des sites Web. Le langage de programmation Python est largement utilisé dans la communauté des sciences des données, et dispose donc des modules et d'outils que nous pouvons utiliser dans nos propres projets. Dans ce travail, nous avons utilisé le module *NewsScraper* [93] qui est un script basé sur le module *BeautifulSoup* [94] de *Python*. C'est une bibliothèque qui facilite le grattage Web et permet généralement aux programmeurs d'économiser des heures ou même des jours de travail.

Afin d'exploiter les images dans les articles, un module a été ajouté dans le script *NewsScraper* qui permet d'extraire l'URL (*Uniform Ressource Locator*) de la première image de chaque article et par la suite le passer dans l'*API* de Google Cloud Vision [95]. Ce dernier va annoter l'image avec la fonctionnalité de détection des étiquettes sous format *JSON*.

L'API *Cloud Vision* permet aux développeurs de comprendre le contenu d'une image en

intégrant des puissants modèles d'apprentissage automatique dans une *API REST* facile à utiliser. *Cloud Vision* classe les images rapidement dans des milliers de catégories. Elle détecte également les objets et les visages et reconnaît les mots en caractères d'imprimerie contenus dans les images. La figure 4.3 illustre cette approche pour un petit ensemble de catégories prédéterminées.

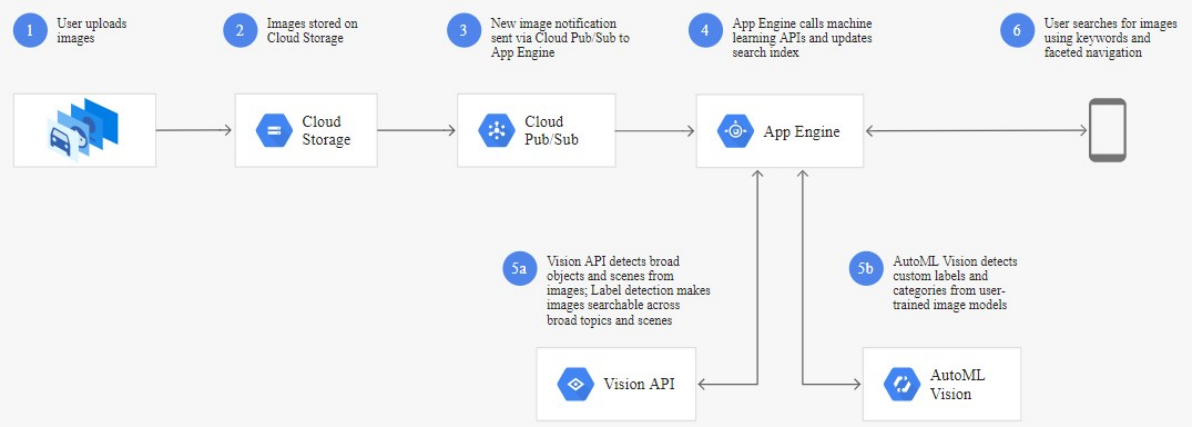


FIGURE 4.3 – Fonctionnement d'API Cloud Vision

4.5 Prétraitement du texte

Après l'extraction des articles, la phase de prétraitement suivante est effectuée :

- La segmentation : Ce processus sépare les mots, les nombres et les ponctuations dans le texte. Par exemple, la phrase *"The province is following through ..."* chaque mot serait séparé par un espace.
- La lemmatisation : les mots dans le texte sont transformés à sa forme de base. Par exemple les mots *playing, plays, played* seraient transformés en une seule forme commune *play* avec le verbe modifié à l'infinitif et le pluriel au singulier. Le lemmatiseur de Porter [96] a été sélectionné pour cette tâche, car il couvre très bien les mots anglais.
- Suppression des mots vides : Élimine les mots fonctionnels tels que les prépositions, les pronoms et les conjonctions qui sont très courants et ne permettent pas de caractériser le texte.

4.6 Représentation thématique du document avec le modèle LDA

Le contenu des articles doit être représenté dans le format que l'allocation de Dirichlet latente et les classificateurs peuvent gérer. Une transformation courante consiste à représenter les documents avec le modèle de sac à mots. Ensuite, une modélisation thématique avec le *LDA* est appliquée sur les caractéristiques textuelles et visuelles représentées en forme respective de sac à mots. La figure 4.4 illustre un exemple de deux premiers thèmes du corpus d'entrainement pour le modèle textuel, visuel et la combinaison de deux.

FIGURE 4.4 – Nuage de mots (wordcloud)

Un nombre de thèmes a priori doit être fixé pour construire le modèle *LDA*. Pour cela, il faut comparer les qualités d'apprentissage des modèles *LDA* à un nombre variable de thèmes. Afin d'évaluer la qualité d'apprentissage du modèle *LDA*, il faut calculer la perplexité d'un ensemble de documents destiné pour le test. La perplexité indique dans quelle mesure le modèle décrit un ensemble de documents. Une perplexité inférieure suggère un meilleur apprentissage. L'objectif est de choisir un certain nombre de thèmes qui minimisent la perplexité par rapport à d'autres nombres de thèmes. Ce n'est pas le seul critère pris en compte pour choisir le nombre optimal de thèmes, les modèles *LDA* appris à un plus grand nombre de thèmes peuvent prendre plus de temps à converger. La figure 4.5 montre le rapport entre les temps d'apprentissage versus la perplexité de chaque modèle. Le tracer nous suggère que l'apprentissage d'un modèle avec 70 thèmes peut être un bon choix.

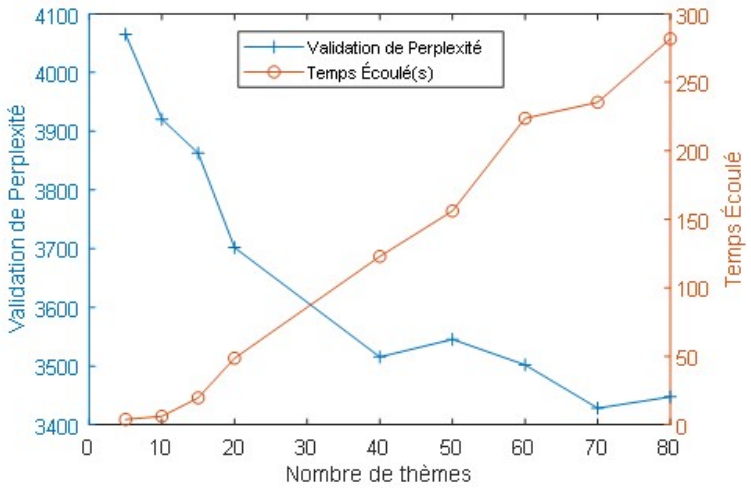


FIGURE 4.5 – Temps d'ajustement versus la perplexité du modèle

4.7 Choix de classificateur

Dans ce travail, l'algorithme utilisé pour la classification des documents multimédias est les forêts aléatoires. Elles sont parmi les méthodes d'apprentissage automatique les plus populaires grâce à leur précision relativement bonne, leur robustesse et leur facilité d'utilisation. C'est un algorithme facile à appliquer, en général, et il nécessite très peu d'ingénierie et d'apprentissage de paramètres.

4.8 Évaluation de performance

Afin d'évaluer les performances de notre modèle de classification, nous examinons la capacité du modèle à prédire correctement les catégories des documents. Nous représentons les résultats que nous avons obtenus en utilisant une matrice de confusion, qui montre comment les prédictions sont faites par le modèle. Les colonnes représentent les classes prédites et les lignes représentent les classes réelles. La figure 4.6 illustre la matrice de confusion du modèle textuel.

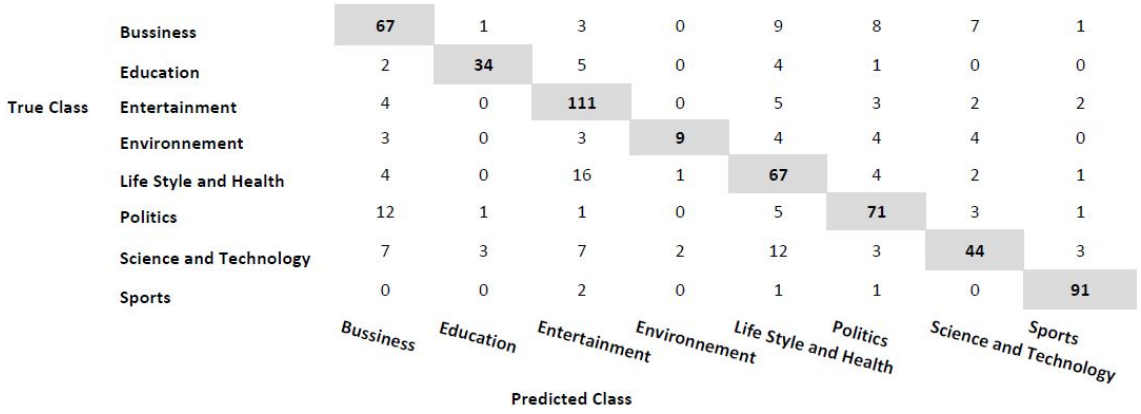


FIGURE 4.6 – Matrice de confusion pour le modèle de classification textuelle

À partir de cette matrice de confusion, nous pouvons calculer la précision et le rappel qui sont utilisés pour mesurer la performance d'un modèle de classification. Les deux sont basés sur une mesure de pertinence. La précision est une mesure d'exactitude à condition qu'une classe spécifique ait été récupérée. Il s'agit du rapport entre le nombre de documents pertinents récupérés et le nombre total de documents pertinents et non pertinents récupérés par le modèle. Le Rappel, d'autre part, mesure la capacité d'un modèle à sélectionner des documents d'une certaine catégorie à partir d'un ensemble de documents. Il s'agit du rapport entre le nombre de documents pertinents récupérés et le nombre total de documents pertinents dans l'ensemble de documents.

Par exemple, la précision pour la classe *Bussiness* est le nombre des pages Web de la classe *Bussiness* correctement prédites (67) sur toutes les pages Web de la classe *Bussiness* prédites ($67 + 2 + 4 + 3 + 4 + 12 + 7 = 99$), ce qui équivaut à $67/99 = 67.67\%$. Donc 2/3 des pages Web que notre modèle les classe comme *Bussiness* sont en fait des pages Web *Bussiness*. D'autre part, le rappel pour la classe *Bussiness* est le nombre des pages Web de la classe *Bussiness* correctement prédites (67) sur le nombre des pages Web de la classe *Bussiness* réelles ($67 + 1 + 3 + 0 + 9 + 8 + 7 + 1 = 96$), qui est de $67/96 = 69.79\%$. Cela signifie que notre modèle a classé 2/3 des pages Web de la classe *Bussiness* comme *Bussiness*.

La F_1-mesure est un autre outil pour mesurer les performances d'un modèle de classification de documents. Elle prend en compte à la fois la précision et le rappel. C'est la

moyenne harmonique de la précision et du rappel. Elle est calculée à l'aide de l'équation suivante :

$$F_1 - mesure = 2 \times \frac{\text{Précision} \times \text{Rappel}}{\text{Précision} + \text{Rappel}} \tag{4.1}$$

Une autre mesure utile utilisée pour évaluer les performances d'un modèle est l'exactitude (*Accuracy*), qui est l'exactitude globale du modèle. Il indique à quel point les prévisions sont proches des résultats réels. Il est calculé en divisant la somme des classifications correctes faites par le modèle sur le nombre total de classifications.

4.9 Expérimentation et résultats

Plusieurs expériences ont été effectuées en utilisant le modèle textuel, visuel et la combinaison de deux. Le classificateur des forêt aléatoires représentée par la classe *TreeBagger* de *Matlab* [97] a été choisi comme algorithme de classification. Nous avons divisé au hasard notre ensemble de données : 90% de l'ensemble de documents sont pour la phase d'apprentissage du modèle, tandis que les 10 autres % servent de tests afin d'évaluer la performance du système de classification.

Catégorie	Documents test	Documents d'apprentissage	**Total**
Affaire	96	855	951
Éducation	46	422	468
Loisir	127	1143	1270
Environnement	27	248	275
Style de vie et Santé	95	850	945
Politique	94	847	941
Science et Technologie	81	736	817
Sports	95	857	952
Total	1323	5296	6618

TABLE 4.2 – Distribution de documents d'apprentissage et test

4.9.1 Test 1 : Comparaison entre LDA et le modèle basé sur la fréquence de mots TF

La réduction de la dimensionnalité est une technique d'apprentissage automatique non supervisée souvent utilisée avec des modèles supervisés. Bien que la réduction de la dimensionnalité rend souvent un modèle basé sur les caractéristiques moins interprétables, il est toujours très efficace pour réduire le temps d'apprentissage en réduisant le nombre de caractéristiques. Dans une approche traditionnelle du sac à mots nous utilisons une matrice de fréquence de mots où chaque ligne représente un document du corpus, chaque colonne représente un mot du corpus et chaque cellule représente le nombre d'occurrences d'un terme particulier d'un document particulier. Cette approche aboutit souvent à une matrice énorme. Nous avons beaucoup de paramètres à estimer lorsque nous utilisons une telle matrice comme entrées de modèle.

Dans l'approche *LDA*, chaque document du corpus est caractérisé par une distribution de Dirichlet sur les variables latentes (thèmes). Et chaque thème est caractérisé par une autre distribution de Dirichlet sur tous les mots. Pour faire une comparaison entre l'utilisation de fréquence de mots *TF* et *LDA*, nous avons créé deux modèles de classifications. Le premier est basé sur *TF* en tenant compte des caractéristiques textuelles seulement. SVM a été choisi comme algorithme de classification, avec la matrice de fréquence de mots, grâce à sa meilleure performance par rapport à d'autres algorithmes. Et le deuxième est basé sur la fusion de deux modèles *LDA* textuel et visuel avec l'algorithme de classification des forêts aléatoires. Les résultats sont résumés dans les tableaux 4.3 et 4.4.

Modèle / Catégorie	TF		
	Précision(%)	Rappel(%)	F_1-mesure(%)
Affaire	78.08	59.37	67.74
Éducation	81.63	86.95	84.21
Loisir	74.65	85.82	79.85
Environnement	78.94	55.55	65.21
Style de vie et Santé	61.61	64.21	62.88
Politique	79.20	85.10	82.05
Scien.Techno.	63.95	67.90	65.86
Sport	95.45	88.42	91.80
Moyenne(%)	76.69	74.17	74.92
Accuracy(%)		75.79	

TABLE 4.3 – Résultats de classification du modèle basé sur la fréquence des mots

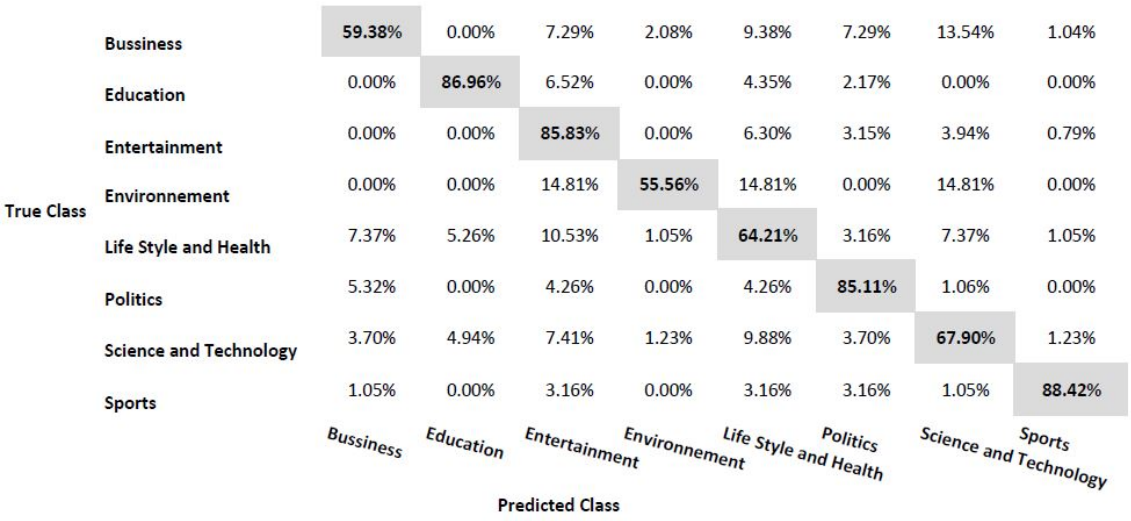


FIGURE 4.7 – Matrice de confusion pour le modèle basé sur la fréquence des mots

	Précision(%)	Rappel(%)	F-mesure(%)	Accuracy(%)	Nombre d'attributs	Temps écoulé
TF + SVM	76.69	74.17	74.92	75.79	29245	123(s)
LDA + RF	80.43	72.07	74.11	76.55	60	5(s)

TABLE 4.4 – Comparaison des résultats entre TF et LDA

Bien que la F_1-mesure du modèle basé sur *TF* est de 74.92% soit légèrement supérieure à celle du modèle basé sur *LDA* 74.11%, il atteint finalement 29245 attributs distincts avec 100% des données d'apprentissage, tandis que le nombre des attributs du modèle *LDA* est de 60 seulement. Aussi, le temps d'apprentissage écoulé pour le modèle de classification basé sur *TF* est de 123(s), tandis que pour le modèle basé sur *LDA* est de 25 fois plus rapide, soit de 5(s).Une différence significative !

Il est intéressant de noter que le modèle de classification basé sur *LDA* surpasse le modèle basé sur *TF* en termes d'interprétabilité. C'est une technique prometteuse dans la classification textuelle, car nous pouvons compresser toutes les informations contenues dans le texte dans une matrice dense et de faible dimension, ce qui nous permet d'ajouter de nombreuses autres attributs pour améliorer les performances du modèle sans se soucier de surapprentissage.

4.9.2 Test 2 : Modèle textuel vs Modèle visuel

les résultats de l'application du modèle *LDA* et du classificateur des forêts aléatoires sur les caractéristiques textuelles et visuelles séparément sont résumés dans le tableau 4.5.

Modèle / **Catégorie**	**RF [LDA Textuel]**			**RF [LDA Visuel]**		
	Précision(%)	Rappel(%)	F_1-mesure(%)	Précision(%)	Rappel(%)	F_1-mesure(%)
Affaire	67.67	69.79	68.71	30.30	31.25	30.76
Éducation	87.17	73.91	80	40	26.08	31.57
Loisir	75	87.40	80.72	45.33	53.54	49.09
Environnement	75	33.33	46.15	42.85	33.33	37.5
Style de vie et Santé	62.61	70.52	66.33	33.33	32.63	32.97
Politique	74.73	75.53	75.13	39.77	47.87	43.26
Scien.Techno.	70.96	54.32	61.53	31.25	24.69	27.58
Sport	91.91	95.78	93.81	76.66	72.63	74.59
Moyenne(%)	75.64	70.08	71.55	42.40	40.26	40.92
Accuracy(%)		74.74			42.97	

TABLE 4.5 – Résultats de classification de deux modalités séparément

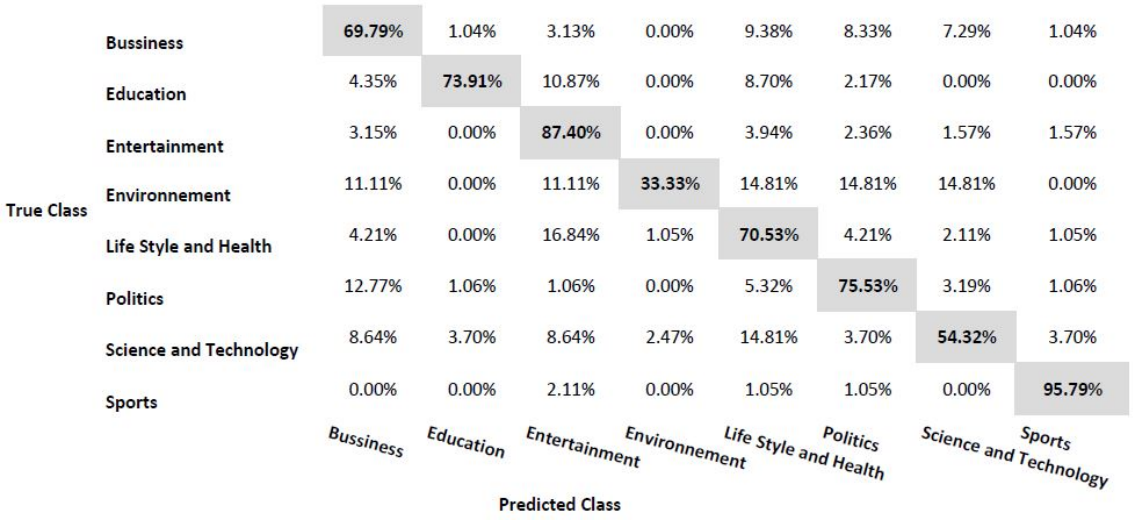


FIGURE 4.8 – Matrice de confusion pour le modèle textuel

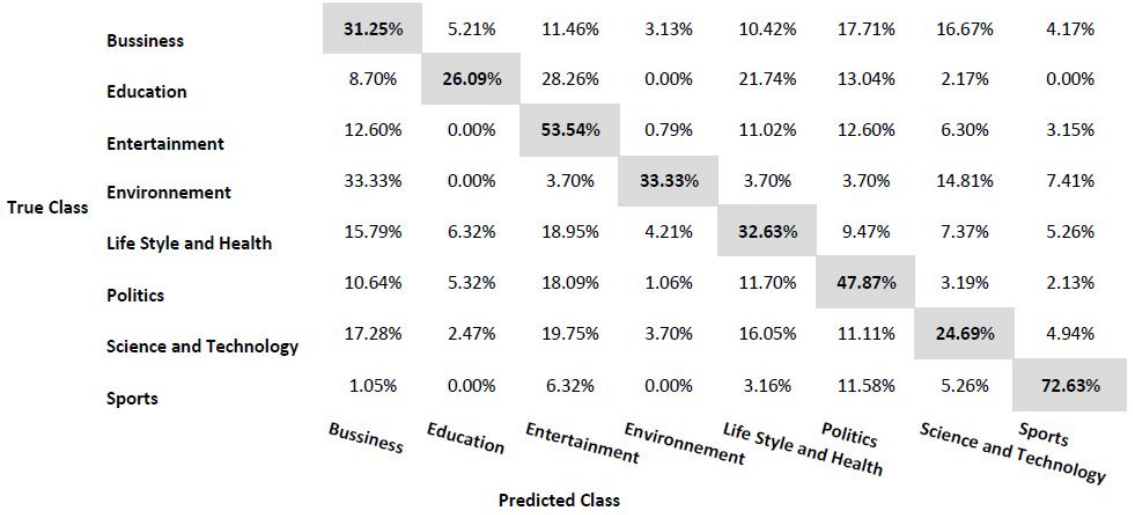


FIGURE 4.9 – Matrice de confusion pour le modèle visuel

Nous remarquons que la modalité textuelle surpasse la modalité visuelle dans toutes les mesures. La valeur de F_1-mesure pour la classification basée sur les caractéristiques textuelles est de 71.55% tandis qu'elle est de 40.92% pour la classification basée sur les caractéristiques visuelles. Cela indique que les caractéristiques textuelles constituent une source d'information plus fiable et plus solide que les données visuelles.

Aussi, on peut remarquer que les résultats du modèle de classification basée sur les caractéristiques visuelles de la catégorie *Sport* sont jugés satisfaisants (Précision 76.66%, Rappel 72.63%, F_1-mesure 74.59%). Une explication possible de cela est le fait que les images dans les documents multimédias des articles de cette catégorie contiennent des informations assez spécifiques telles que des stades et des joueurs. Cependant, les images des catégories *Affaire* et *Éducation* et *Environnement* décrivent divers sujets et que leur aspect visuel varie donc fortement ce qui explique les résultats obtenus.

4.9.3 Test 3 : Classification multimodale

L'utilisation de méthodes permettant de combiner plusieurs ensembles de caractéristiques, constitue un défi à la fois intéressant et important pour les tâches de classification. Dans ce contexte, nous avons utilisé la fusion précoce qui s'applique au niveau des caractéristiques, dans lesquelles ces dernières sont concaténées afin de créer un vecteur de caractéristiques commun. Ensuite, un classificateur est formé en utilisant ce vecteur pour construire le modèle de classification final. Le principal avantage de la fusion précoce est que le classificateur peut voir toutes les caractéristiques en même temps et qu'une seule phase d'apprentissage est requise.

Modèle / **Catégorie**	**RF [LDA Textuel + LDA Visuel]**		
	Précision(%)	Rappel(%)	F_1-mesure(%)
Affaire	68.36	69.79	69.07
Éducation	92.10	76.08	93.33
Loisir	74.19	90.55	81.56
Environnement	100	37.03	54.05
Style de vie et Santé	65.97	67.36	66.66
Politique	74.51	80.85	77.55
Scien.Techno.	73.43	58.02	64.82
Sport	94.84	96.84	95.83
Moyenne(%)	80.43	72.07	74.11
Accuracy(%)		76.55	

TABLE 4.6 – Résultats de classification de la fusion de deux modalités

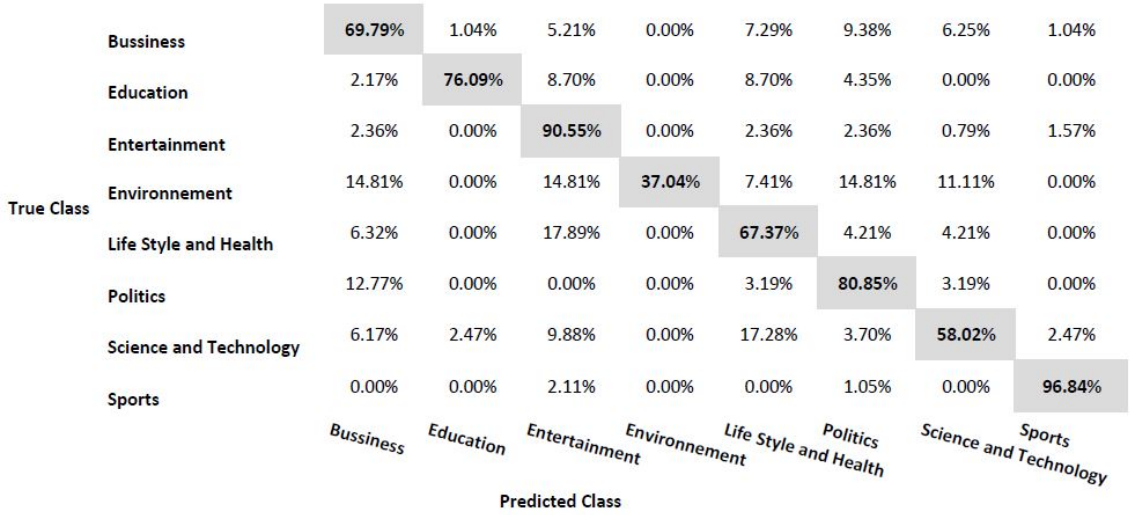


FIGURE 4.10 – Matrice de confusion pour le modèle de fusion

Pour appliquer cette fusion, nous avons concaténé les deux matrices de chacun de modèles *LDA* textuel et visuel. Ces matrices représentent les probabilités d'observer chaque sujet dans chaque document utilisé pour s'adapter au modèle *LDA*. Ensuite un modèle de classification est construit à partir de cette fusion multimodale. Le tableau 4.6 présente les résultats de la fusion.

On remarque que les performances de classification sont améliorées de 2.5% avec une valeur de F_1-mesure de 74.11% tandis qu'elle est de 71.55% pour le modèle basé sur les caractéristiques textuelles uniquement. Cette expérience montre que les caractéristiques visuelles peuvent améliorer considérablement la précision de la classification lorsque l'image dans un article numérique est étroitement liée au contenu du texte. Par exemple, on peut constater que la valeur de F_1-mesure dans le modèle textuel pour la catégorie *Sport* est augmenté de plus que 3% dans le modèle de fusion (93.81% à 95.83%).

Pour vérifier que les résultats de la comparaison du modèle textuel avec le modèle de fusion ne varient pas beaucoup avec l'utilisation d'autre ensemble de documents d'apprentissage et de test, nous avons utilisé une méthode qui sélectionne au hasard les documents à conserver pour l'ensemble d'apprentissage et pour l'ensemble de test. Cette opération a été répéter 10 fois. Le tableau 4.7 présente les résultats de ce test.

Modèle \ Itération	$F1$-mesure (%)										
	1	2	3	4	5	6	7	8	9	10	Moyenne
Textuel	71.55	70.15	70.84	71.04	70.02	71.23	69.79	70.32	69.65	72.04	70.66
Textuel + Visuel	74.11	72.42	72.81	72.97	72.1	73.51	72.16	72.3	71.86	74.59	72.88
Différence(%)	2.56	2.27	1.97	1.93	2.08	2.28	2.37	1.98	2.21	2.55	2.22

TABLE 4.7 – Signification des résultats

On remarque que le modèle de classification de fusion (Textuel + Visuel) a toujours une valeur supérieure à celui du modèle textuel. La valeur minimum de la différence entre les deux modèles est de 1.93% et la valeur maximum est de 2.56%, avec une moyenne de 2.22%.

4.9.4 Test 4 : Comparaison avec d'autres algorithmes

Le tableau 4.8 présente les résultats de notre approche de classification en utilisant d'autres classificateurs. On remarque que le modèle de classification des forêts aléatoires a obtenu une meilleure classification dans les trois modèles textuel, visuel et la combinaison de deux par rapport aux autres classificateurs.

	LDA Textuel	LDA Visuel	Fusion
	Accuracy(%)		
KNN	61.72	34.19	61.83
Forêt aléatoire	**74.74**	**42.97**	**76.55**
Arbre de décision	61.72	34.19	61.89
SVM	65.51	41.20	65.81

TABLE 4.8 – Comparaison avec d'autres algorithmes de classification

Chapitre 5

Conclusion

Dans ce travail, nous avons proposé une approche pour la classification des documents multimédias basée sur l'allocation de Dirichlet latente et les techniques d'apprentissage automatique. Nous avons démontré que la combinaison de deux modalités textuelles et visuelles donne des résultats prometteurs et une précision de classification légèrement supérieure par rapport aux autres modèles qui exploitent une seule modalité. Nous avons démontré aussi que le modèle *LDA* peut être utilisé comme une méthode efficace de réduction des dimensions pour la modélisation des données multimédias. Entre autres, il permet de découvrir des structures thématiques cachées dans les documents.

Dans le cadre des travaux futurs, ce travail peut être amélioré en utilisant un classificateur d'apprentissage profond pour les documents à la place des forêts aléatoires. Aussi, exploiter une troisième modalité, comme les vidéos dans les documents multimédias, donnera potentiellement une meilleure classification de ces derniers.

Annexe A

Code Matlab

Charger et extraire les données

Le fichier NewsData.mat contient les informations des articles des journaux électroniques,comme le lien, le texte de l'article, ainsi que le lien de la première image de l'article avec sa description (extrait avec API Google Vision) et enfin la catégorie de chaque article (Bussiness - Education - Entertainement - Environnement - Life and Style and Health - Politics - Science and Technology - Sports).

```
load('Articles.mat');
data=Articles;
```

Visualiser les premières lignes du tabaleau data.

```
head(data)
```

ans = *8×6 table*

	ID	articles_link	articles_text	articles_top_i…	articles_labeli…	category
1	1	'http://www.…	'A group of r…	'https://ichef…	'{'fast food r…	'Business'
2	7	'http://www.…	'Has the cos…	'https://ichef…	'{'forehead', …	'Business'
3	8	'http://www.…	'How about …	'https://ichef…	'{'video gam…	'Business'
4	9	'http://www.…	'A daigou or…	'https://ichef…	'{'recreation'…	'Business'
5	10	'http://www.…	'An interior …	'https://ichef…	'{'house', 'lof…	'Business'
6	11	'http://www.…	'Anne Bode…	'https://ichef…	'{'smile', 'pu…	'Business'
7	12	'http://www.…	'At a speed …	'https://ichef…	'{'drink'}'	'Business'
8	2	'http://www.…	'Annual hou…	'https://ichef…	'{'material', '…	'Business'

Partitionnez les données en un ensemble d'apprentissages *dataTrain* et un autre de tests *dataTest*. Spécifier le pourcentage de rétention à 10%.

```
cvp = cvpartition(data.category,'Holdout',0.1);
% 90% de données d'apprentissage
dataTrain = data(cvp.training,:);
% 10% de données test
dataTest = data(cvp.test,:);
```

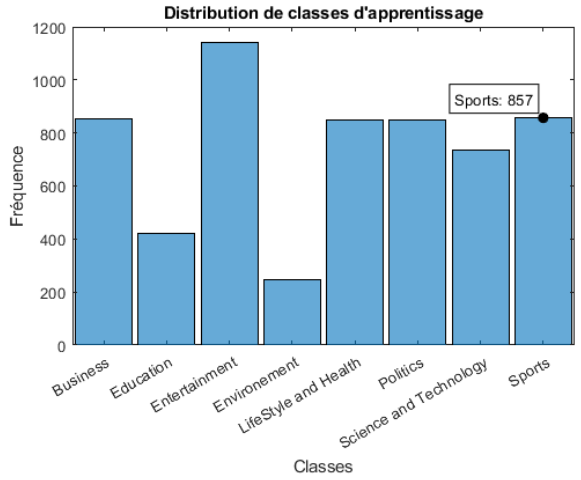
Convertir la colonne *category* en *categorical* afin qu'elle puisse être utilisée dans des modèles d'apprentissage automatique.

```
dataTrain.category = categorical(dataTrain.category);
```

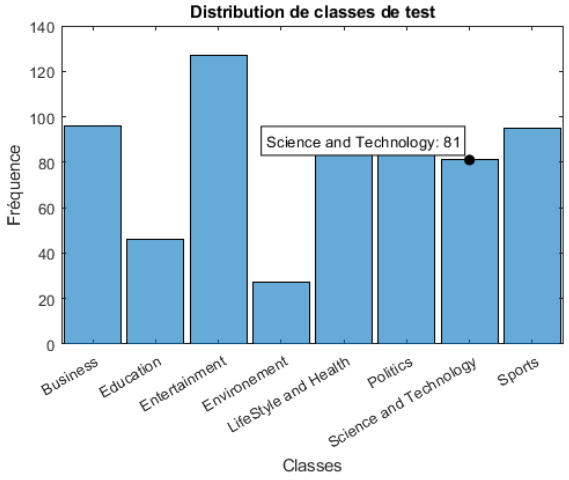
Visualiser la répartition de classes dans les données à l'aide d'un histogramme.

```
% Répartition de classes dans l'ensemble d'apprentissage
figure
h = histogram(dataTrain.category);
xlabel("Classes")
ylabel("Fréquence")
```

```
title("Distribution de classes d'apprentissage")
```




```
dataTest.category = categorical(dataTest.category);
figure
h = histogram(dataTest.category);
xlabel("Classes")
ylabel("Fréquence")
title("Distribution de classes de test")
```



Prétraitement de données

Extraire les données textuelles et visuelles pour l'ensemble d'apprentissages et l'ensemble test

```
% Données d'apprentissage
textTrain=dataTrain.articles_text;
imageTrain=dataTrain.articles_labelimage;
imageTrain(1:5)
```

```
ans = 5×1 cell array
    {'{'forehead', 'beard', 'muscle', 'chin', 'jaw', 'cheek'}'                                                                }
    {'{'video game software', 'art', 'technology', 'fictional character', 'cartoon', 'pc game', 'fiction', 'product', 'recreation'}'       }
    {'{'recreation', 'fictional character', 'design', 'art', 'product', 'pattern', 'fiction'}'                                }
    {'{'house', 'loft', 'floor', 'living room', 'room', 'real estate', 'flooring', 'home', 'interior design'}'                }
    {'{'smile', 'public relations', 'professional', 'journalist', 'entrepreneur', 'girl', 'official', 'television presenter', 'socialite'}'}
```

```
% Données de test
textTest=dataTest.articles_text;
imageTest=dataTest.articles_labelimage;
```

Utiliser la fonction *PretraitementNewsData (détaillée à la fin de ce document)* pour la prétraitement de données

textuelles (colonne *articles_text*) et visuelles (colonne *articles_labelimage*) pour l'ensemble d'apprentissages

et l'ensemble de tests.

```
% Prétraitement des données d'apprentissage
textTrain = PretraitementNewsData(textTrain);
imageTrain = PretraitementNewsData(imageTrain);

% Prétraitement des données test
textTest = PretraitementNewsData(textTest);
imageTest = PretraitementNewsData(imageTest);
```

Visualiser les 5 premiers documents textuels et visuels prétraités

```
imageTrain(1:5)
```

```
ans =
  5×1 tokenizedDocument:

(1,1)  6 tokens: forehead beard muscl chin jaw cheek
(2,1) 12 tokens: video game softwar art technolog fiction charact cartoon game fiction product recreat
(3,1)  8 tokens: recreat fiction charact design art product pattern fiction
(4,1) 12 tokens: hous loft floor live room room real estat floor home interior design
(5,1) 11 tokens: smile public relat profession journalist entrepreneur girl offici televis present socialit
```

Création du modèle de sac de mots (Bag of words)

```
% Sac de mots textuels
textBag = bagOfWords(textTrain);

%Supprimer les mots du sac de mot qui ont une fréquence < 2.
textBag = removeInfrequentWords(textBag,2);
textBag
```

```
textBag =
bagOfWords with 29297 words and 5958 documents:

    cost   live   creep   bank   england   …
       2      2       1      1         1
       0      0       0      0         0
       …
```

```
% Sac de mots visuels
imageBag=bagOfWords(imageTrain);

%Supprimer les mots du sac de mot qui ont une fréquence < 2.
```

```
imageBag = removeInfrequentWords(imageBag,2);
imageBag
```

```
imageBag =
bagOfWords with 1313 words and 5958 documents:

    forehead   beard   muscl   chin   jaw   …
           1       1       1      1     1
           0       0       0      0     0
           …
```

Création du modèle d'allocation de Dirichlet latente

Ajuster un modèle LDA avec 30 sujets pour le modèle textuel et 25 sujets pour le modèle visuel.

```
% Nombre de topics pour le modèle visuel
numTopicsImage = 30;

% Nombre de topics pour le modèle textuel
numTopicsText = 30;

% Modèle LDA_textuel
LDA_textuel = fitlda(textBag,numTopicsText,'Verbose',0);
LDA_textuel.FitInfo
```

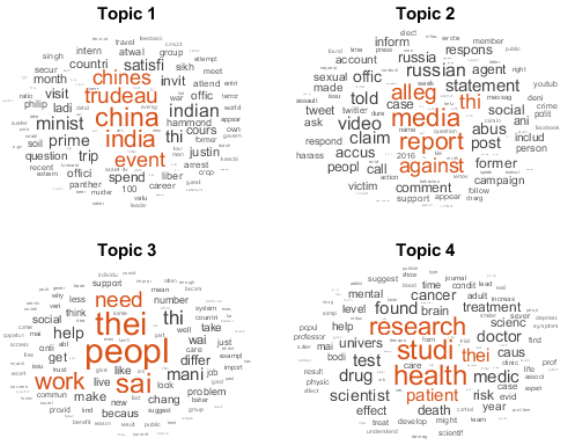
```
ans = struct with fields:
          TerminationCode: 1
        TerminationStatus: "Relative tolerance on log-likelihood satisfied."
            NumIterations: 13
    NegativeLogLikelihood: 1.6469e+07
               Perplexity: 2.8811e+03
                   Solver: "cgs"
                  History: [1×1 struct]
```

```
% Modèle LDA_visuel
LDA_visuel= fitlda(imageBag,numTopicsImage,'Verbose',0);
LDA_visuel.FitInfo
```

```
ans = struct with fields:
          TerminationCode: 1
        TerminationStatus: "Relative tolerance on log-likelihood satisfied."
            NumIterations: 22
    NegativeLogLikelihood: 2.0082e+05
               Perplexity: 48.2246
                   Solver: "cgs"
                  History: [1×1 struct]
```

Visualiser les 4 premiers topiques du modèle LDA textuel et visuel sous forme de nuage des mots

```
figure;
for topicIdx = 1:4
    subplot(2,2,topicIdx)
    wordcloud(LDA_textuel,topicIdx);
    title("Topic " + topicIdx)
end
```




```
figure;
for topicIdx1 = 1:4
    subplot(2,2,topicIdx1)
    wordcloud(LDA_visuel,topicIdx1);
    title("Topic " + topicIdx1)
end
```

Création des modèles de classification

- **Modèle de classification basé sur la fréquence des mots**

Term Frequency matrix ou matrice de fréquence de mots est une matrice où chaque ligne représente un document du corpus, chaque colonne représente un terme du corpus et chaque cellule représente le nombre de fréquences d'un terme particulier dans un document particulier. Former un modèle de classification linéaire multiclasses à l'aide de *`fitcecoc`*. Spécifier la propriété `Counts` du modèle de sac de mots textuels comme prédicteurs et les étiquettes *category* comme réponse.

```
% Modèle de classification basé sur la fréquence de mots
MdlBagCount=fitcecoc(textBag.Counts,dataTrain.category,'Learners','linear')
```

```
MdlBagCount =
  classreg.learning.classif.CompactClassificationECOC
      ResponseName: 'Y'
        ClassNames: [Business    Education    Entertainment    Environement    LifeStyle and Health    Politics    Science and Technology    Sports]
    ScoreTransform: 'none'
    BinaryLearners: {28×1 cell}
      CodingMatrix: [8×28 double]


  Properties, Methods
```

Encodez l'ensemble de test sous forme d'une matrice de fréquence de mots selon le modèle de sac de mots textuels:

```
XTest=encode(textBag,textTest);
YTest=nominal(dataTest.category);
```

Prédire les étiquettes de l'ensemble test à l'aide du modèle formé :

```
YPred = predict(MdlBagCount,XTest);
```

Préparer les variables pour claculer la matrice de confusion avec la fonction *confusionmat*

```
TargetVector=transpose(grp2idx(categorical(YTest)));
OutputVector=transpose(grp2idx(categorical(YPred)));
[Cm,order] =confusionmat(TargetVector,OutputVector);
```

Table de mesure pour le modèle de classiffication (*Accuracy, recall(Sensitivity), F1_score etc.*)

```
confusionp.getValues(Cm)
```

```
ans = struct with fields:
                         Accuracy: 0.7579
                            Error: 0.2421
                      Sensitivity: 0.7417
                      Specificity: 0.9646
                        Precision: 0.7669
                FalsePositiveRate: 0.0354
                         F1_score: 0.7492
    MatthewsCorrelationCoefficient: 0.7171
                            Kappa: 0.0963
```

```
confusionmat(TargetVector,OutputVector)
```

```
ans =
    57     0     7     2     9     7    13     1
     0    40     3     0     2     1     0     0
     0     0   109     0     8     4     5     1
     0     0     4    15     4     0     4     0
     7     5    10     1    61     3     7     1
     5     0     4     0     4    80     1     0
     3     4     6     1     8     3    55     1
     1     0     3     0     3     3     1    84
```

- **Modèle de classification basé sur LDA (allocation de Dirichlet latente)**

Former un modèle de classification avec *TreeBagger* (*Forêts aléatoires*)

```
numTree=100;

% Extraire la matrice de probabilité Document/Topic
P1 = (LDA_textuel.DocumentTopicProbabilities);
C=nominal(dataTrain.category);
P2 = (LDA_visuel.DocumentTopicProbabilities);

%Former le modèle de classification
MdlClass_textuel = TreeBagger(numTree,P1,C,'OOBPrediction','On',...
    'Method','classification');
[label1,posterior1] = oobPredict(MdlClass_textuel);

MdlClass_visuel = TreeBagger(numTree,P2,C,'OOBPrediction','On',...
    'Method','classification');
[label2,posterior2] = oobPredict(MdlClass_visuel);
```

Combiner les probabilités Document/Topic du modèle respectifs LDA_textuel et LDA_visuel

```
FusionLDA=[P1 P2];

%Former le modèle de calssification de la fusion de deux modalités
MdlClassFusion = TreeBagger(350,FusionLDA,C,'OOBPrediction','On',...
    'Method','classification');
[label3,posterior3] = oobPredict(MdlClassFusion);
```

Prédire les étiquettes de l'ensemble de test et évaluer le modèle de classification

```
% Modèle textuel
Ttest=tokenizedDocument(textTest);
XTest2=transform(LDA_textuel,Ttest);
YPred2 = predict(MdlClass_textuel,XTest2);

%Modèle visuel
ITest=tokenizedDocument(imageTest);
XTest3=transform(LDA_visuel,ITest);
YPred3 = predict(MdlClass_visuel,XTest3);

%Modèle de fusion
XTest4=[XTest2 XTest3];
YPred4 = predict(MdlClassFusion,XTest4);
```

Préparer les variables pour claculer la matrice de confusion avec la fonction *confusionmat du Matlab*

```
% matrice de confusion du modèle textuel
```

```
OutputVector2=transpose(grp2idx(categorical(YPred2)));
[Cm2,order2] =confusionmat(TargetVector,OutputVector2);
confusionp.getValues(Cm2)
```

```
ans = struct with fields:
                        Accuracy: 0.7474
                           Error: 0.2526
                     Sensitivity: 0.7008
                     Specificity: 0.9629
                       Precision: 0.7564
                FalsePositiveRate: 0.0371
                         F1_score: 0.7155
    MatthewsCorrelationCoefficient: 0.6861
                           Kappa: 0.1342
```

```
confusionmat(TargetVector,OutputVector2)
```

```
ans =
    67     1     3     0     9     8     7     1
     2    34     5     0     4     1     0     0
     4     0   111     0     5     3     2     2
     3     0     3     9     4     4     4     0
     4     0    16     1    67     4     2     1
    12     1     1     0     5    71     3     1
     7     3     7     2    12     3    44     3
     0     0     2     0     1     1     0    91
```

```
% matrice de confusion du modèle visuel
OutputVector3=transpose(grp2idx(categorical(YPred3)));
[Cm3,order3] =confusionmat(TargetVector,OutputVector3);
confusionp.getValues(Cm3)
```

```
ans = struct with fields:
                        Accuracy: 0.4297
                           Error: 0.5703
                     Sensitivity: 0.4026
                     Specificity: 0.9165
                       Precision: 0.4240
                FalsePositiveRate: 0.0835
                         F1_score: 0.4092
    MatthewsCorrelationCoefficient: 0.3286
                           Kappa: 0.6165
```

```
confusionmat(TargetVector,OutputVector3)
```

```
ans =
```

```
    30     5    11     3    10    17    16     4
     4    12    13     0    10     6     1     0
    16     0    68     1    14    16     8     4
     9     0     1     9     1     1     4     2
    15     6    18     4    31     9     7     5
    10     5    17     1    11    45     3     2
    14     2    16     3    13     9    20     4
     1     0     6     0     3    11     5    69
```

```
%matrice de confusion de la combinaison de deux modèle, textuel et visuel
OutputVector4=transpose(grp2idx(categorical(YPred4)));
[Cm4,order4] =confusionmat(TargetVector,OutputVector4);
confusionp.getValues(Cm4)
```

```
ans = struct with fields:
                         Accuracy: 0.7655
                            Error: 0.2345
                      Sensitivity: 0.7207
                      Specificity: 0.9654
                        Precision: 0.8043
                FalsePositiveRate: 0.0346
                         F1_score: 0.7411
    MatthewsCorrelationCoefficient: 0.7183
                            Kappa: 0.0671
```

```
confusionmat(TargetVector,OutputVector4)
```

```
ans =
    67     1     5     0     7     9     6     1
     1    35     4     0     4     2     0     0
     3     0   115     0     3     3     1     2
     4     0     4    10     2     4     3     0
     6     0    17     0    64     4     4     0
    12     0     0     0     3    76     3     0
     5     2     8     0    14     3    47     2
     0     0     2     0     0     1     0    92
```

La fonction *PretraitementNewsData* :

```
function [documents] = PretraitementNewsData(textData)
% 1- Supprimer la ponctuation enutilisant erasePunctyuation
cleanTextData = erasePunctuation(textData);
% 2- Convertir les données textuelles en miniscules en utilisant lower.
cleanTextData = lower(cleanTextData);
% 3- Segmentation du text en utilisant tokenizedDocument.
documents = tokenizedDocument(cleanTextData);
```

```
% 4- Lemmatiser les mots avec normalizeWords.
documents = normalizeWords(documents);
% 5- suppression une liste des mots vides en utilisant removeWords.
documents = removeWords(documents,stopWords);
% 6-Supprimer les mots avec 2 caractères ou moins en utilisant removeShortWords.
documents = removeShortWords(documents,2);
% 7- Supprimer les mots avec 15 caractères ou plus en utilisant removeLongWords.
documents = removeLongWords(documents,15);
end
```

Bibliographie

[1] David M Blei, Andrew Y Ng, and Michael I Jordan. Latent dirichlet allocation. *Journal of machine Learning research*, 3(Jan) :993–1022, 2003.

[2] Vladimir Vapnik. *The nature of statistical learning theory*. Springer science & business media, 2013.

[3] Irina Rish et al. An empirical study of the naive bayes classifier. In *IJCAI 2001 workshop on empirical methods in artificial intelligence*, volume 3, pages 41–46, 2001.

[4] Aixin Sun, Ee-Peng Lim, and Wee-Keong Ng. Web classification using support vector machine. In *Proceedings of the 4th international workshop on Web information and data management*, pages 96–99. ACM, 2002.

[5] Rung-Ching Chen and Chung-Hsun Hsieh. Web page classification based on a support vector machine using a weighted vote schema. *Expert Systems with Applications*, 31(2) :427–435, 2006.

[6] Dou Shen, Zheng Chen, Qiang Yang, Hua-Jun Zeng, Benyu Zhang, Yuchang Lu, and Wei-Ying Ma. Web-page classification through summarization. In *Proceedings of the 27th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 242–249. ACM, 2004.

[7] J Alamelu Mangai, Satej Milind Wagle, and V Santhosh Kumar. A novel web page classification model using an improved k nearest neighbor algorithm. In *Proceedings of the 3rd International Conference on Intelligent Computational Systems, Singapore*, 2013.

[8] Viktor de Boer, Maarten van Someren, Tiberiu Lupascu, et al. Classifying web pages with visual features. In *WEBIST (1)*, pages 245–252, 2010.

[9] Arul Prakash Asirvatham, Kranthi Kumar Ravi, Arul Prakash, et al. Web page classification based on document structure. In *IEEE National Convention*, 2001.

[10] Tom M Mitchell. Machine learning, 1997.

[11] Richard S Sutton, Andrew G Barto, et al. *Introduction to reinforcement learning*, volume 2. MIT press Cambridge, 1998.

[12] Xiaojin Jerry Zhu. Semi-supervised learning literature survey. Technical report, University of Wisconsin-Madison Department of Computer Sciences, 2005.

[13] James M Keller, Michael R Gray, and James A Givens. A fuzzy k-nearest neighbor algorithm. *IEEE transactions on systems, man, and cybernetics*, (4) :580–585, 1985.

[14] Yiming Yang, Xin Liu, et al. A re-examination of text categorization methods. In *Sigir*, volume 99, page 99, 1999.

[15] Ammar Ismael Kadhim. Survey on supervised machine learning techniques for automatic text classification. *Artificial Intelligence Review*, 52(1) :273–292, 2019.

[16] Charu C Aggarwal. An introduction to neural networks. In *Neural Networks and Deep Learning*, pages 1–52. Springer, 2018.

[17] Leo Breiman. Bagging predictors. *Machine learning*, 24(2) :123–140, 1996.

[18] Yoshua Bengio, Aaron Courville, and Pascal Vincent. Representation learning : A review and new perspectives. *IEEE transactions on pattern analysis and machine intelligence*, 35(8) :1798–1828, 2013.

[19] Yoshua Bengio et al. Learning deep architectures for ai. *Foundations and trends® in Machine Learning*, 2(1) :1–127, 2009.

[20] Yoshua Bengio. Deep learning of representations : Looking forward. In *International Conference on Statistical Language and Speech Processing*, pages 1–37. Springer, 2013.

[21] Yoshua Bengio, Pascal Lamblin, Dan Popovici, and Hugo Larochelle. Greedy layer-wise training of deep networks. In *Advances in neural information processing systems*, pages 153–160, 2007.

[22] George Dahl, Abdel-rahman Mohamed, Geoffrey E Hinton, et al. Phone recognition with the mean-covariance restricted boltzmann machine. In *Advances in neural information processing systems*, pages 469–477, 2010.

[23] George E Dahl, Dong Yu, Li Deng, and Alex Acero. Context-dependent pre-trained deep neural networks for large-vocabulary speech recognition. *IEEE Transactions on audio, speech, and language processing*, 20(1) :30–42, 2011.

[24] Geoffrey E Hinton, Simon Osindero, and Yee-Whye Teh. A fast learning algorithm for deep belief nets. *Neural computation*, 18(7) :1527–1554, 2006.

[25] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

[26] Tomáš Mikolov, Anoop Deoras, Stefan Kombrink, Lukáš Burget, and Jan Černockỳ. Empirical evaluation and combination of advanced language modeling techniques. In *Twelfth Annual Conference of the International Speech Communication Association*, 2011.

[27] Antoine Bordes, Xavier Glorot, Jason Weston, and Yoshua Bengio. Joint learning of words and meaning representations for open-text semantic parsing. In *Artificial Intelligence and Statistics*, pages 127–135, 2012.

[28] Yves Kodratoff. Knowledge discovery in texts : a definition, and applications. In *International Symposium on Methodologies for Intelligent Systems*, pages 16–29. Springer, 1999.

[29] Robert C Berwick, Steven P Abney, and CL Tenny. *Principle-based parsing : Computation and psycholinguistics*, volume 44. Springer Science & Business Media, 2012.

[30] Joël Plisson, Nada Lavrac, Dunja Mladenic, et al. A rule based approach to word lemmatization. *Proceedings of IS-2004*, pages 83–86, 2004.

[31] Xiaoshan Pan and Hisham Assal. Providing context for free text interpretation. In *International Conference on Natural Language Processing and Knowledge Engineering, 2003. Proceedings. 2003*, pages 704–709. IEEE, 2003.

[32] Fabrizio Sebastiani. Classification of text, automatic. *The encyclopedia of language and linguistics*, 14 :457–462, 2006.

[33] Gerard Salton. Automatic text processing : The transformation, analysis, and retrieval of. *Reading : Addison-Wesley*, 169, 1989.

[34] Gerard Salton and Michael J McGill. *Introduction to modern information retrieval*. mcgraw-hill, 1983.

[35] Gerard Salton, Chung-Shu Yang, and CLEMENT T Yu. A theory of term importance in automatic text analysis. *Journal of the American society for Information Science*, 26(1) :33–44, 1975.

[36] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. *arXiv preprint arXiv :1301.3781*, 2013.

[37] Shoushan Li, Rui Xia, Chengqing Zong, and Chu-Ren Huang. A framework of feature selection methods for text categorization. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP : Volume 2-Volume 2*, pages 692–700. Association for Computational Linguistics, 2009.

[38] Son Doan and Susumu Horiguchi. An efficient feature selection using multi-criteria in text categorization. In *Fourth International Conference on Hybrid Intelligent Systems (HIS'04)*, pages 86–91. IEEE, 2004.

[39] Yvan Saeys, Iñaki Inza, and Pedro Larrañaga. A review of feature selection techniques in bioinformatics. *bioinformatics*, 23(19) :2507–2517, 2007.

[40] Ricardo Gutierrez-Osuna. Introduction to pattern analysis. *Lecture Notes, Texas A&M University*, 2005.

[41] Wikipedia. Feature selection, 2019. URL `https://en.wikipedia.org/wiki/Feature_selection`.

[42] Julie Hamon. *Optimisation combinatoire pour la sélection de variables en régression en grande dimension : Application en génétique animale*. PhD thesis, Université Lille 1 France, 2013.

[43] Tu Minh Phuong, Zhen Lin, and Russ B Altman. Choosing snps using feature selection. In *2005 IEEE Computational Systems Bioinformatics Conference (CSB'05)*, pages 301–309. IEEE, 2005.

[44] Yvan Saeys, Iñaki Inza, and Pedro Larrañaga. A review of feature selection techniques in bioinformatics. *bioinformatics*, 23(19) :2507–2517, 2007.

[45] Ron Kohavi and George H John. Wrappers for feature subset selection. *Artificial intelligence*, 97(1-2) :273–324, 1997.

[46] George H John, Ron Kohavi, and Karl Pfleger. Irrelevant features and the subset selection problem. In *Machine Learning Proceedings 1994*, pages 121–129. Elsevier, 1994.

[47] Kevin Jarrett, Koray Kavukcuoglu, Yann LeCun, et al. What is the best multi-stage architecture for object recognition ? In *2009 IEEE 12th international conference on computer vision*, pages 2146–2153. IEEE, 2009.

[48] Min Lin, Qiang Chen, and Shuicheng Yan. Network in network. *arXiv preprint arXiv :1312.4400*, 2013.

[49] Dengsheng Lu and Qihao Weng. A survey of image classification methods and techniques for improving classification performance. *International journal of Remote sensing*, 28(5) :823–870, 2007.

[50] HB Kekre, Tanuja K Sarode, Sudeep D Thepade, and Vaishali Vaishali. Improved texture feature based image retrieval using kekreâĂŹs fast codebook generation algorithm. In *Thinkquest˜ 2010*, pages 143–149. Springer, 2011.

[51] Guang-Hai Liu and Jing-Yu Yang. Content-based image retrieval using color difference histogram. *Pattern recognition*, 46(1) :188–198, 2013.

[52] Subrahmanyam Murala, RP Maheshwari, and R Balasubramanian. Local tetra patterns : a new feature descriptor for content-based image retrieval. *IEEE transactions on image processing*, 21(5) :2874–2886, 2012.

[53] David G Lowe. Distinctive image features from scale-invariant keypoints. *International journal of computer vision*, 60(2) :91–110, 2004.

[54] Navneet Dalal and Bill Triggs. Histograms of oriented gradients for human detection. 2005.

[55] Aude Oliva and Antonio Torralba. Modeling the shape of the scene : A holistic representation of the spatial envelope. *International journal of computer vision*, 42 (3) :145–175, 2001.

[56] Hongliang Jin, Qingshan Liu, Hanqing Lu, and Xiaofeng Tong. Face detection using improved lbp under bayesian framework. In *Third International Conference on Image and Graphics (ICIG'04)*, pages 306–309. IEEE, 2004.

[57] Kraisak Kesorn and Stefan Poslad. An enhanced bag-of-visual word vector space model to represent visual content in athletics images. *IEEE Transactions on Multimedia*, 14(1) :211–222, 2011.

[58] Li-Jia Li, Chong Wang, Yongwhan Lim, David M Blei, and Li Fei-Fei. Building and using a semantivisual image hierarchy. In *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 3336–3343. IEEE, 2010.

[59] Zhiwu Lu and Liwei Wang. Learning descriptive visual representation for image classification and annotation. *Pattern Recognition*, 48(2) :498–508, 2015.

[60] Kevin Jarrett, Koray Kavukcuoglu, Yann LeCun, et al. What is the best multi-stage architecture for object recognition ? In *2009 IEEE 12th international conference on computer vision*, pages 2146–2153. IEEE, 2009.

[61] Paul E Rybski, Daniel Huber, Daniel D Morris, and Regis Hoffman. Visual classification of coarse vehicle orientation using histogram of oriented gradients features. In *2010 IEEE Intelligent vehicles symposium*, pages 921–928. IEEE, 2010.

[62] Pedro F Felzenszwalb, Ross B Girshick, David McAllester, and Deva Ramanan. Object detection with discriminatively trained part-based models. *IEEE transactions on pattern analysis and machine intelligence*, 32(9) :1627–1645, 2009.

[63] Yang Bai, Lihua Guo, Lianwen Jin, and Qinghua Huang. A novel feature extraction method using pyramid histogram of orientation gradients for smile recognition. In *2009 16th IEEE International Conference on Image Processing (ICIP)*, pages 3305–3308. IEEE, 2009.

[64] Yann LeCun, Léon Bottou, Yoshua Bengio, Patrick Haffner, et al. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11) :2278–2324, 1998.

[65] Yann LeCun, Bernhard E Boser, John S Denker, Donnie Henderson, Richard E Howard, Wayne E Hubbard, and Lawrence D Jackel. Handwritten digit recognition with a back-propagation network. In *Advances in neural information processing systems*, pages 396–404, 1990.

[66] Kevin Jarrett, Koray Kavukcuoglu, Yann LeCun, et al. What is the best multi-stage architecture for object recognition ? In *2009 IEEE 12th international conference on computer vision*, pages 2146–2153. IEEE, 2009.

[67] Maxime Oquab, Leon Bottou, Ivan Laptev, and Josef Sivic. Learning and transferring mid-level image representations using convolutional neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1717–1724, 2014.

[68] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

[69] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet : A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009.

[70] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1–9, 2015.

[71] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv :1409.1556*, 2014.

[72] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[73] Gary Anthes. Topic models vs. unstructured data. *Communications of the ACM*, 53(12) :16–18, 2010.

[74] Ashok N Srivastava and Mehran Sahami. *Text mining : Classification, clustering, and applications*. Chapman and Hall/CRC, 2009.

[75] M Steyvers and T Griffiths. Probabilistic topic models in latent semantic analysis : A road to meaning, landauer, t. and mc namara, d. and dennis, s. and kintsch, w., eds, 2007.

[76] ChengXiang Zhai. Statistical language models for information retrieval. *Synthesis Lectures on Human Language Technologies*, 1(1) :1–141, 2008.

[77] Barbara Rosario. Latent semantic indexing : An overview. *Techn. rep. INFOSYS*, 240 :1–16, 2000.

[78] Thomas Hofmann. Probabilistic latent semantic indexing. In *ACM SIGIR Forum*, volume 51, pages 211–218. ACM, 2017.

[79] Thomas Hofmann. Unsupervised learning by probabilistic latent semantic analysis. *Machine learning*, 42(1-2) :177–196, 2001.

[80] Thomas K Landauer, Peter W Foltz, and Darrell Laham. An introduction to latent semantic analysis. *Discourse processes*, 25(2-3) :259–284, 1998.

[81] Christos H Papadimitriou, Prabhakar Raghavan, Hisao Tamaki, and Santosh Vempala. Latent semantic indexing : A probabilistic analysis. *Journal of Computer and System Sciences*, 61(2) :217–235, 2000.

[82] Scott Deerwester, Susan Dumais, Thomas Landauer, George Furnas, and Laura Beck. Improving information-retrieval with latent semantic indexing. In *Proceedings of the ASIS annual meeting*, volume 25, pages 36–40. INFORMATION TODAY INC 143 OLD MARLTON PIKE, MEDFORD, NJ 08055-8750, 1988.

[83] Ashok N Srivastava and Mehran Sahami. *Text mining : Classification, clustering, and applications*. Chapman and Hall/CRC, 2009.

[84] Wikipedia. Loi betâ, 2019. URL `https://fr.wikipedia.org/wiki/Loi_betâ`.

[85] Mark Steyvers, Padhraic Smyth, Michal Rosen-Zvi, and Thomas Griffiths. Probabilistic author-topic models for information discovery. In *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 306–315. ACM, 2004.

[86] Gregor Heinrich. Parameter estimation for text analysis. Technical report, Technical report, 2005.

[87] Chong Wang, Bo Thiesson, Chris Meek, and David Blei. Markov topic models. In *Artificial intelligence and statistics*, pages 583–590, 2009.

[88] Michal Rosen-Zvi, Thomas Griffiths, Mark Steyvers, and Padhraic Smyth. The author-topic model for authors and documents. In *Proceedings of the 20th conference on Uncertainty in artificial intelligence*, pages 487–494. AUAI Press, 2004.

[89] Mark Girolami and Ata Kabán. On an equivalence between plsi and lda. In *SIGIR*, volume 3, pages 433–434, 2003.

[90] Yue Lu, Qiaozhu Mei, and ChengXiang Zhai. Investigating task performance of probabilistic topic models : an empirical study of plsa and lda. *Information Retrieval*, 14(2) :178–203, 2011.

[91] Tomonari Masada, Senya Kiyasu, and Sueharu Miyahara. Comparing lda with plsi as a dimensionality reduction method in document clustering. In *International Conference on Large-Scale Knowledge Resources*, pages 13–26. Springer, 2008.

[92] Tse-Hsun Chen, Stephen W Thomas, and Ahmed E Hassan. A survey on the use of topic models when mining software repositories. *Empirical Software Engineering*, 21(5) :1843–1919, 2016.

[93] Github. Newsscraper, 2019. URL `https://github.com/holwech/`.

[94] Leonard Richardson. Beautifulsoup, 2019. URL `https://www.crummy.com/software/BeautifulSoup/bs4/doc/`.

[95] Google. Vision ai, 2019. URL `https://cloud.google.com/vision/?hl=fr/`.

[96] Mathworks. Stemming words with nltk, 2019. URL `https://pythonprogramming.net/stemming-nltk-tutorial/`.

[97] Mathworks. Statistics and machine learning toolbox, 2019. URL https://www.mathworks.com/help/stats/regression-treeBagger-examples.html.