République Algérienne Démocratique et Populaire
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique

Université Abderrahmane Mira de Bejaia
Faculté des Sciences Exactes
Département d'Informatique

ECOLE DOCTORALE RESEAUX ET SYSTEMES DISTRIBUES

Mémoire de Magistère

En Informatique

Option : Réseaux et Systèmes Distribués

Thème

Identification de la sémantique dans des documents répartis

Présenté par

Lamia HAMOUCHE

Devant le jury composé de :

Président	Abdelkamel TARI	M.C.A	Université de Bejaïa
Rapporteur	Marc GELGON	Professeur	Université de Nantes
Examinateur	Abdessamed-Reda GHOMARI	M.C.A	ESI
Examinateur	Youcef AKLOUF	M.C.A	USTHB
Invité	Azze-Eddine MAREDJ	Docteur	CERIST

Promotion : 2008/2009

2011

Remerciements

Louange à dieu tout puissant qui nous a guidés pour l'accomplissement de ce travail et je ne saurais me faire guider sans le guide de dieu, dieu merci.

A cela s'ajoute un groupe de personnes qui m'ont apporté aide est assistance auxquels je présente mes remerciements les plus sincères à commencer par

Remercier tout spécialement mon directeur de mémoire, Mr Marc GELGON, professeur à l'université de Polytech'Nantes. Son soutien, ainsi que sa disponibilité sans faille, m'ont été précieux. Nos passionnantes discussions scientifiques, ainsi que la grande liberté et confiance qu'il m'a accordée, ont constitué le terreau sur lequel ce travail est fondé. Outre les aspects purement scientifiques, il a su me guider dans la compréhension du métier de chercheur, et me transmettre son enthousiasme. Je ne le remercierai jamais assez de cette énergie dépensée, et espère lui avoir montré qu'elle avait été employée à bon escient.

Mr Azze-eddine MAREDJ mon co-directeur de mémoire, docteur au CERIST, de m'avoir orienté, corrigé mon travail, soutenu et encouragé. Merci pour sa disponibilité et sa gentillesse sans égale.

Je tiens également à remercier Mr Nadjib Badache, professeur à l'USTHB et Directeur du CERIST, de m'avoir donné la chance d'entrer dans le monde de la recherche.

Mes remerciements iront ensuite aux membres de l'équipes GRIM de LINA à Polytech'Nantes.

Mes remerciements iront aussi aux membres de mon équipe SMDS, Madjid et Nour eddine.

Mes profonds remerciements et respects aux membres du jury Mr Abdelkamel TARI, maitre de conférences à l'université de Bejaia, Mr Abdessamed-Reda GHOMARI, Docteur d'état en informatique ESI (ex INI), Mr Youcef AKLOUF, Docteur d'état en informatique qui ont accepté d'examiner mon travail.

Mes chaleureux remerciements vont également à mes amies du CERIST Nour El Houda, Saida, Fouzia et Amel, Amel,Faiza et Sahar pour leurs soutiens et encouragements.

Je remercie mes collègues de la promotion, Saida, Sahar, Dalila, faiza, Nabil, Madjid et Rafik pour la bonne ambiance vécue tout au long d'une année de cette belle expérience.

Merci à Mme Zaidi et tous les collègues du service formation pour leur disponibilité.

Enfin, mes derniers remerciements, mais non les moindres, vont à ma famille et à mon future mari Abdelatif. Leur patience face à mon humeur versatile due à mes obscurs problèmes de scientifique force l'admiration.

A toutes les personnes qui m'ont aidé de prés ou de loin
MERCI.

Je dédie ce modeste travail aux personnes qui me sont les plus chères au monde que j'aime et que j'adore, ceux que je vois s'effondrés pour éclairer ma voie, ceux qui ont tant sacrifié pour moi.

A l'honneur de ceux dont je suis fière d'être leur fille

A mes adorables parents

Mouloud et Chérifa

A

Mon futur époux Abdelatif qui m'a encouragé

A

Mes chères sœurs Fadia et Ibtissem qui m'ont soutenu et supporté durant ce mémoire

A

Mon mes chères frères Fouad et Salim que j'adore

A

Ma grand mère Djamila

A

Toute ma famille, mes tantes et oncles, mes cousins et cousines

Table des matières

Liste des Figures

Chapitre 1
Introduction générale

1.1. Introduction

Le développement des technologies de l'information et des télécommunications a entraîné la création et l'utilisation de documents électroniques pour la réalisation des activités quotidiennes au sein des organismes, ces documents électroniques s'accumulent à grande échelle assurant une facilité de création de corpus de textuelles. De telles collections de données textuelles font émerger des besoins fonctionnels notamment la catégorisation de documents (classification). Les méthodes d'extraction automatique, dans lesquelles on trouve la classification, permettent d'identifier des relations entre entités linguistiques dont la tâche semble péniblement réalisable voir infaisable à la main. Des méthodes mathématiques permettent d'automatiser un tel processus de repérage.

La classification dans son sens le plus général désigne l'opération qui consiste à regrouper des entités en classes sur la base de caractéristiques communes, de sorte que les entités semblables soient regroupées et des entités non semblables soient séparées [27, 28]. La classification de textes est une des tâches classiques de la recherche d'information et, en tant que telle, elle a suscité de nombreuses études. Les modèles de classification de texte, considèrent, pour la plupart, des documents plats avec des représentations « sac de mots » qui ne prennent pas en compte l'ordre des mots dans les documents ni leur structure. Parmi les méthodes génératives, le modèle naïve Bayes est l'un des plus utilisés.

Dès le départ il est nécessaire de différencier la classification non supervisée et la classification supervisée.

1 ***La classification supervisée*** consiste à construire des règles de décision en se basant sur un ensemble de données pour lesquelles les étiquettes des classes sont connues a priori.
2 ***La classification non supervisée*** a pour but de trouver une organisation des données cohérente et valide, qui puisse mettre en évidence les vraies structures dans un ensemble de données sans aucune connaissance a priori sur les données traitées, [31].

1.2. Motivations & Objectifs

Notre travail se situe dans le contexte de la classification d'un corpus de documents multimédia par une approche d'apprentissage statistique. Il a été montré que des représentations de texte ou d'images avec des "sacs de mots", ou "sacs de mots visuels", dans le cas des images, permettent de déterminer des classes (thèmes de textes ou classes d'objets visuels) avec un bon rapport "qualité/prix".

Nous abordons un problème difficile et souvent délaissé, celui de la classification non supervisée distribuée de documents par des méthodes probabilistes. Notre étude se concentre en particulier sur le modèle de mélange de lois multinomiales avec variables latentes thématiques au niveau des documents. La classification distribuée est un problème émergent et seulement quelques travaux sur le sujet ont été réalisés dans le domaine [10, 27, 28, 65].

Dans le cadre de ce mémoire, nous nous plaçons donc dans une situation où nous avons une collection d'ensembles de données existantes. Dans notre cas les données sont représentées sous forme de corpus de documents situées sur plusieurs noeuds. Le but ultime de chaque organisation est de découvrir les principales relations dans son ensemble de données. Cette découverte peut être raffinée en tenant compte des dépendances entre les différentes analyses effectuées par les différents noeuds, afin de produire une image fidèle de la structure globale cachée dans les différents corpus sans en avoir un accès direct. Dans certains cas, il pourrait y avoir aussi des problèmes techniques, la classification d'un grand ensemble de données unique ne peut pas être réalisable. Une approche distribuée permettrait de distribuer les classifications et procéder à une fusion des différents résultats.

Dans cette étude, nous nous intéressons au problème de l'apprentissage non supervisé (clustering) et spécifiquement au clustering distribué, tout en préservant la confidentialité des données (échange uniquement de paramètres entre les nœuds), en utilisant des méthodes

de classification à base de modèles probabilistes.

1.3. Notre approche

Notre travail décrit une méthode de classification de données distribuées. L'approche qui sera étudiée est basée sur des méthodes probabilistes. Ayant une collection de corpus distribuée sur différents nœuds, le problème consiste à classifier chacun de ces corpus localement et procéder par la suite à une agrégation des différentes classifications au niveau de chaque nœud par échange de paramètres. La solution proposée est basée sur les principes des algorithmes EM (Expectation-Maximisation), elle fonctionne de manière locale (nœud) et asynchrone. Pour ce faire, nous proposons d'utiliser le modèle de l'analyse sémantique latente probabiliste (PLSA) subdivisé en deux phases : une phase locale et une phase distribuée.

- La phase locale où un algorithme de classification basé sur les algorithmes EM est appliqué au niveau de chaque nœud indépendamment des autres nœuds.
- La phase distribuée reviendrait à faire participer tous les nœuds du réseau pour générer la classification globale au niveau de chaque nœud. L'algorithme Gossib est utilisé à ce niveau.

Pour sa validation, nous comparons le résultat ainsi obtenu avec le résultat obtenu en centralisant la collection de l'ensemble des corpus.

1.4. Organisation du mémoire

Dans le chapitre 2, la première section aborde les problèmes de prétraitement des données, des questions pratiques de formation du vocabulaire à celles de la modélisation statistique des comptes et pose les notations utilisées dans la suite du mémoire. Le reste du chapitre est consacré à l'état de l'art proprement dit, en commençant par les méthodes les plus utilisées pour la classification de textes non supervisée. La seconde section regroupe sommairement les méthodes non probabilistes: K-moyennes, LSA, NMF, Regroupement spectral et Goulot d'information. Dans la troisième partie, nous nous intéressons aux algorithmes reposant sur des modèles probabilistes. Nous y abordons le modèle de mélange de multinomiales, l'Analyse sémantique latente probabiliste (pLSA) et LDA. En fin de chapitre, nous consacrons une section aux liens entre les différents modèles avant de justifier

notre choix pour le modèle pLSA.

Le chapitre 3, qui est la principale contribution de ce travail, présente notre approche de classification distribuée. Nous nous concentrons sur le modèle pLSA, qui se base sur le modèle de mélange multinomiales pour l'apprentissage automatique. Ensuite, nous justifions notre choix de combiner pLSA pour l'apprentissage statistique pour la classification et l'Algorithme Gossip pour le calcul distribué. Par la suite, nous présentons l'architecture de notre solution et nous détaillons l'algorithme pLSA_Gossip proposé.

Dans le chapitre 4, nous décrivons l'environnement de travail ainsi que la collection utilisé (*20Newsgroups),* et les résultats expérimentaux obtenus. Á la fin, une discussion des résultats est présentée pour valider l'approche.

Enfin, le chapitre 5 conclut ce travail en mettant en avant nos contributions et en présentant les différentes perspectives possibles.

Chapitre 2

Etat de l’art

2.1 Introduction

Nous allons aborder dans ce chapitre le problème de la classification non supervisée de documents par différentes méthodes notamment les méthodes probabilistes. Notre étude se concentre en particulier sur le modèle de mélange de lois multinomiales avec variables latentes thématiques au niveau des documents.

Parmi les nombreux modèles existants pour la classification non supervisée, nous avons choisi ceux qui ont fait leurs preuves pour l'analyse de données textuelles. Le domaine d'application du texte a en effet certaines spécificités, notamment dans la modélisation des documents, qui font que des algorithmes pourtant performants par ailleurs y sont manifestement moins adaptés. C'est le cas par exemple de tous ceux qui se fondent sur une distance euclidienne sur la matrice de comptes, notamment les algorithmes de type hiérarchique agglomératif et la version la plus élémentaire de l'algorithme des K-means.

Nous adoptons l'hypothèse du «sac de mots» (comme vecteur sur l'espace du vocabulaire) comme point de départ du traitement des documents. Il faut certainement y voir l'influence considérable du modèle d'espace vectoriel (Vector Space Model) en fouille de textes et en recherche d'information.

Notre présentation des méthodes non probabilistes commence par les méthodes les plus classiques en classification non supervisée, avant de détailler les spécificités d'application aux données textuelles. Nous nous intéresserons ensuite en détail aux quatre méthodes de classification non supervisée de documents qui nous semblent les plus importantes (analyse sémantique latente, regroupement spectral, factorisation en matrices non-négatives et goulot

d'information) et en citerons plus brièvement d'autres. Nous nous intéresserons ensuite aux modèles probabilistes, d'abord par une succession de présentations des modèles, du plus simple (le mélange de multinomiales) au plus évolué (allocation Dirichlet latente).

2.2 Modèles non probabilistes

Le problème du partitionnement de données existe bien au-delà de la fouille de textes et de très nombreux modèles, éventuellement adaptés d'autres domaines, peuvent servir à regrouper des documents similaires. D'une façon générale, il suffit de définir une représentation des documents induisant une distance et une méthode pour diviser l'espace à partir de cette matrice de proximité. Ce point de vue engage à dissocier le problème de la classification de celui de la représentation des données. Les modèles de classification non supervisée les plus courants : algorithmes hiérarchique agglomératif et K-means, vont être présenté dans la partie suivante.

Nous nous intéressons ensuite à quatre méthodes qui ont reçu beaucoup d'échos ces dernières années en apprentissage automatique et/ou en fouille de textes: l'analyse sémantique latente (LSA), le regroupement spectral, la factorisation en matrices non-négatives (NMF) et le goulot d'information. Aussi nous allons présenter brièvement d'autres idées relatives à la classification non supervisée et s'inscrivant dans des cadres non probabilistes.

Notations Notons n_T le nombre de groupes formés par l'algorithme de classification non supervisée, qui correspondront plus intuitivement, dans les cas que nous étudierons, aux thèmes.

Nous notons n_v le nombre de documents dans le corpus d'apprentissage. Le vocabulaire est construit uniquement à partir des formes apparaissant au moins une fois dans le corpus d'apprentissage. Nous notons n_w le nombre de formes différentes, c'est-à-dire la taille du vocabulaire.

Nous supposons qu'en sortie de la phase de prétraitement, nous obtenons une matrice de comptes ou matrice mot/document C, de terme général C_{wd} avec

$w \in \{1,\dots,n_w\}, d \in \{1,\dots,n_v\}$ et représentant le nombre d'occurrences du mot d'indice *w* dans le document d'indice *d* du corpus d'apprentissage.

$$c = \begin{matrix} & documents \\ \text{termes} \end{matrix} \begin{matrix} 10 & 6 & \cdots & 8 \\ 3 & 0 & \cdots & 2 \\ 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 1 & 2 & \cdots & 3 \end{matrix}$$

Pour $= 1, \ldots, n_D$, on note

$$l_d = \sum_{w=1}^{n_W} C_{wd}$$

le nombre d'occurrences dans le texte d et

$$l = \sum_{d=1}^{n_D} l_d$$

le nombre total d'occurrences dans le corpus d'apprentissage, somme de tous les termes de la matrice de comptes.

De façon similaire à ce qui précède, nous définissons pour le corpus de test : le nombre de documents n_D^*, la matrice termes/documents C^*, de terme général $C{\sim}d^*$ (avec $d^* \in \{l, \ldots, n_v^*\}$), la longueur l_d^* du document d^* et le nombre total d'occurrences dans le corpus de test 1^*.

2.2.1 Modèles de classification non supervisée

Les modèles de classification non supervisée sont couramment divisés en deux classes: modèles hiérarchiques ou modèles de partitionnements.

Modèles hiérarchiques Le principe des modèles hiérarchiques est de procéder itérativement, en recherchant à chaque étape un groupe à diviser selon un certain critère qui dépend de la distance entre deux groupes [12, 31]. Supposons dans un premier temps que nous sachions calculer une telle distance. Il est alors possible de procéder par regroupement *agglomératif :*

- le point de départ est de considérer qu'il y a autant de groupes que d'exemples *(n_T=n_v);*
- à chaque étape, il s'agit de calculer les distances entre chaque couple de groupes et de fusionner les deux plus proches.

L'algorithme peut s'arrêter une fois qu'un nombre de thèmes déterminé a été atteint ou

continuer jusqu'à obtention d'un groupe unique, ce qui permet d'exhiber une suite de partitionnement imbriqués les uns dans les autres pour un nombre de groupes quelconque entre 1 et n_D. Le résultat peut se représenter de façon synthétique sous la forme d'un arbre dont les feuilles sont les documents. On appelle ce résultat un *dendrogramme* [31].

L'application de ces algorithmes repose donc entièrement sur la définition d'une distance entre deux groupes de documents T_t et T_t'.

En classification non supervisée en général, les algorithmes de regroupement hiérarchique donnent de bonnes performances si des distances (entre observations, et entre groupes) pertinentes vis-à-vis des données et du problème sont sélectionnées, mais au prix d'un temps de calcul important (en l'absence d'heuristique astucieuse, il faudra calculer les distances entre chaque couple de points) [31].

K-means Les algorithmes de partitionnement se distinguent des algorithmes hiérarchiques par le fait qu'ils produisent en général un regroupement unique pour un nombre de thèmes donnés et non un dendrogramme. Leur famille est plus générale que celles des algorithmes hiérarchiques et il est de ce fait difficile d'en donner une caractérisation précise. Des traits communs aux algorithmes de partitionnements sont qu'ils optimisent en général un critère sur l'ensemble des données de façon itérative et qu'ils sont souvent sensibles à la procédure d'initialisation [31].

De par sa simplicité et ses bons résultats sur des problèmes variés, l'algorithme des K-means [46] reste l'une des méthodes les plus utilisées en classification non supervisée. Le principe des K- means est que les documents et les thèmes sont des points d'un espace multidimensionnel, sur lequel nous supposons donnée une distance *d*. Les documents sont dans un premier temps affectés aléatoirement à chaque groupe. Puis le centroïde de chaque thème T_t est calculé comme étant le barycentre de l'ensemble des individus de ce groupe.

Les centroïdes et assignation des individus aux thèmes sont ensuite ajustés en alternance, comme le montre la figure 1, dans laquelle chaque document est représenté par son vecteur de comptes.

- Tirer selon une loi uniforme un thème pour chaque document dans l'ensemble $\{T_1, \ldots, T_{n_T}\}$

- Pour un certain nombre d'itérations, répéter:

  - Calcul des nouveaux barycentres:

$$\forall t \in 1, \ldots, n_T \ , \forall w \in 1, \ldots, n_W \ , \beta_{tw} \frac{d \in Tt \, C_{wd}}{|Tt|}$$

  - Réaffectation des documents, pour $d \in 1, \ldots, n_D$

$$\underset{t = 1, \ldots, n_T}{arg \min} d(C_d \ , T_t)$$

Figure 1. Principe de l'algorithme des K-means.

Comme pour les algorithmes de classification hiérarchiques, le choix de la représentation et de la distance sont primordiaux pour l'efficacité des K-means. Pour les données textuelles représentées sous forme de comptes, la distance euclidienne étant manifestement inadaptée, il est d'usage de faire appel à la distance cosinus *d(Cd, Tt)*, calculée par :

$$d(C_d, \tau_t)^{-1} = \frac{\sum_{w=1}^{n_W} C_{wd} \beta_{tw}}{\sqrt{\sum_{w=1}^{n_W} C_{wd}^2} \sqrt{\sum_{w=1}^{n_W} \beta_{tw}^2}} ,$$

selon la définition classique du cosinus entre deux vecteurs, qui mesure leur similarité dans un espace euclidien. Cependant, il a été également observé que de meilleurs résultats pouvaient être obtenus en choisissant d'autres représentations que les vecteurs de comptes non modifiés [34].

2.2.2 Transformation des comptes

L'application des algorithmes de regroupement aux vecteurs de comptes donne des résultats mitigés [34]. C'est la raison pour laquelle il est légitime de s'interroger sur l'opportunité d'utiliser d'autres représentations et d'autres distances. L'une des raisons des mauvaises résultats sur les vecteurs de comptes bruts, semble être un mauvais équilibre entre la représentation des mots les plus fréquents et celle des mots plus rares. Il existe deux idées modifiant cet équilibre, l'une consistant à ignorer les comptes, l'autre à les pondérer de façon adaptée.

La représentation la plus simple, fait abstraction de la longueur des textes: un document peut être vu comme un ensemble de n_w indicateurs binaires, chacun d'entre eux correspondant à la présence ou l'absence d'un mot du vocabulaire dans le document. Néanmoins, [48] montre en classification supervisée que ce modèle, connu sous le nom *Bernoulli multivarié,* donne systématiquement de moins bonnes performances, pour des vocabulaires de grande taille, qu'une représentation fondée sur le «sac de mots». Cette analyse confirme l'intuition que le nombre d'occurrences d'un terme dans un texte est important, au-delà du simple fait qu'il soit présent.

Une autre remarque sur la représentation sous forme de comptes bruts est que la longueur des documents a une grande importance. De fait, si la distance considérée est le simple produit scalaire entre vecteurs de comptes, un texte contenant beaucoup d'occurrences obtiendra nécessairement des scores de similarité plus grands qu'un texte contenant exactement les mêmes mots mais avec des comptes plus petits. L'intuition dicte que cet effet de la longueur doit être compensé, l'usage de la distance cosinus normalisée sur les comptes bruts est exactement équivalente à l'usage du produit scalaire sur les vecteurs de comptes normalisés par la norme $\mathcal{L}_2$. Notons ici que la normalisation par la norme $\mathcal{L}_1$ a elle pour effet de diviser les comptes par la longueur du document, pour obtenir des *profils* sur le vocabulaire.

Les normalisations ne suffisent pas à compenser les effets de déséquilibre entre les comptes évoqués ci-dessus. A cet effet, nous introduisons à présent la *fréquence inversée sur documents,* appelée plus couramment *idf* (pour *inverse document frequency).* Ce facteur modificateur, qui connaît de nombreuses variantes, corrige une lacune de la représentation sous forme de vecteurs de comptes: les mots qui présentent les comptes les plus forts ne sont pas forcément les plus pertinents. Ainsi, les anti-mots, qui apparaissent dans tous les documents de façon très fréquente, sont en général tout-à-fait dénués d'intérêt pour la classification thématique. L'idée est de pénaliser ces termes apparaissant dans tous les documents (et, parallèlement, de récompenser les mots plus rares) par l'idf égale à :

$$\forall w \in 1, \dots, n_W \ , idf\ w\ = \log \frac{n_D}{\sum_{d=1}^{n_D} 1_{c_{wd}>0}}$$

Ce facteur connaît de nombreuses autres définitions que celle que nous présentons ici mais le principe est toujours le même: pour chaque mot, il est d'autant plus grand que le terme apparaît dans peu de documents, ou, en d'autres termes, à nombres d'occurrences égaux, il

privilégie les mots pour lesquelles les apparitions sont les plus concentrées dans un nombre limité de documents. Il faut en particulier noter que, pour les mots figurant dans tous les documents, nous avons $\sum_{d=1}^{n_D} 1_{c_{wd}>0} = n_D$, et donc $idf(w) = 0$.

L'usage de cette représentation, connue sous le nom de ***TFIDF*** est courante en classification non supervisée, en particulier avec l'algorithme des K-means associé à une distance cosinus [34]. [64] rapporte des expériences selon lesquelles l'algorithme des K-means est plus rapide et obtient de meilleures performances que les algorithmes hiérarchiques. L'algorithme des K- means avec **TFIDF** et la distance cosinus constitue donc une bonne référence pour juger des performances d'un algorithme de classification non supervisée de documents.

2.2.3 Analyse sémantique latente (LSI/LSA)

L'indexation sémantique latente (Latent Semantic Indexing, LSI) [13] s'inscrit dans une optique traditionnelle en analyse de données. Il s'agit de la décomposition de matrices selon leurs directions propres (ou singulières) pour conserver un maximum d'« information» sur un nombre minimum de dimensions. *Analyse des correspondances* [3,44], *analyse en composantes principales* ou *échelonnement multidimensionnel (multidimensional scaling* [40] sont autant de techniques faisant appel à des décompositions en valeurs singulières, elles-mêmes équivalentes à des décompositions en valeurs propres d'une matrice carrée associée (matrice de similitude, de covariance empirique, etc.). Nous allons voir que c'est le même principe qui est à l'œuvre ici, même si les motivations premières des auteurs de l'indexation sémantique latente viennent du domaine de la recherche d'information. Deerwester et al. constatent que la technique la plus simple, consistant à répondre à une requête d'utilisateur en retournant les documents qui contiennent le plus d'occurrences des mots contenus dans la requête, se heurte à des limites liées à des problèmes de polysémie et de synonymie:

- il est possible que dans certains documents du corpus, les termes de la requête soient présents mais employés dans un autre sens que celui recherché par l'utilisateur (renvoi de documents non pertinents, dégradation du score de précision) ;
- il arrive à l'inverse que le mot demandé ne se trouve pas dans un document pourtant pertinent pour le thème car c'est un synonyme qui est employé (oubli de documents pertinents, dégradation du score de rappel).

L'hypothèse de Deerwester et al. est que trouver un moyen de considérer le contexte d'un

mot et les liens sous-jacents entre des termes dans le corpus permet de régler en partie ces problèmes, au moins celui des synonymies. Si, par exemple, l'utilisateur recherche le mot *voiture* et qu'un document utilise uniquement le terme *automobile,* le fait que d'autres termes du même champ lexical tels que *moteur* ou *carrosserie* soient, d'une part, présents en nombre dans ce texte et, d'autre part, en cooccurrence fréquente avec le terme de la requête *voiture* ailleurs dans le corpus, permet d'affirmer que le texte est probablement pertinent.

Pour découvrir une telle structure *latente* dans le corpus, l'idée est de faire appel à une décomposition en valeurs singulières *(Singltlar Value Decomposition,* SVD) de la matrice de comptes C. On suppose que C est de rang m. C est alors décomposable en valeurs singulières, c'est-à-dire qu'il existe des matrices orthonormales *U (*n_w X m) et *V (*n_D X m) et une matrice diagonale *S* (m X *m),* de valeurs propres strictement positives et rangées en ordre décroissant, telles que:

$$C = USV^T$$

L'avantage de cette représentation est qu'elle fournit un nouvel espace de dimension m pour représenter C, les valeurs singulières sur la diagonale de *S* illustrant 1'« importance » des différentes dimensions. Nous savons que m est inférieur à min (n_w, n_D*)* mais cela ne garantit en rien qu'il soit petit. Or remplacer la représentation sous forme de comptes par une autre n'a pas grand intérêt si les dimensions des matrices sont sensiblement les mêmes. Par conséquent,

l'étape suivante est de réduire substantiellement la dimension de l'espace latent, tout en perdant le moins d'« information» possible. Une solution consiste à supprimer les dimensions associées aux valeurs propres les plus faibles.

Plus formellement, pour tout *k* = 1, ..., m, on note $C^{(k)} = U^{(k)}S^{(k)}V^{(k)^T}$, avec :

- $U^{(k)}$ *(*n_w X *k), k* premières colonnes de *U;*
- $S^{(k)}$ *(k* x *k),* matrice diagonale correspondant à *S* sans ses m -*k* plus petites valeurs propres;
- $V^{(k)}$ *(k* x n_D*), k* premières lignes de *V.* La matrice $C^{(k)}$est une approximation de C dont on peut montrer qu'elle est la meilleure de rang *k* au sens de la norme L^2, c'est-à-dire encore celle qui retient le mieux la variance des données initiales.

La décomposition en valeurs singulières est liée à la diagonalisation des matrices carrées symétriques réelles CC^T et C^TC, ce que l'on constate assez facilement avec les formules suivantes:

$$C^T C = V S^2 V^T$$

$$C C^T = U S^2 U^T$$

Or, $\boldsymbol{C^T C}$ est une matrice de similarité entre les documents alors que $\boldsymbol{C C^T}$ est une matrice de similarité entre les mots (vus comme des vecteurs de documents). Avec les approximations ci-dessus, $C^{(k)^T} C^{(k)}$ et $C^{(k)} C^{(k)^T}$ nous donnent donc un moyen de comparer deux éléments dans le nouvel espace. [13] explique aussi comment projeter un nouveau vecteur de mots (requête ou document) C_d dans le nouvel espace: sa représentation est déterminé par $S^{(k)^{-1}} U^{(k)^T} C_d$

L'évaluation porte sur une tâche de recherche d'information : la décomposition en valeurs singulières est appliquée à la matrice de comptes d'un corpus pour lequel sont disponibles des couples requête/liste de documents pertinents. Les requêtes sont ensuite projetées dans l'espace latent et les documents retenus sont ceux qui sont les plus proches dans ce nouvel espace. Les résultats sont présentés sous forme de courbes précision/rappel.

Ils sont modérément encourageants. LSI semble toujours faire au moins aussi bien qu'une méthode basique de renvoi par mots communs *(term matching)* mais se comporte de façon inégale par rapport à d'autres techniques plus évoluées *(SMART* [57] et *clluster-based retrieval* [71]). Les auteurs soulignent que les différences entre les étapes de prétraitement (filtrage, document fréquence inverse, racinisation ...) peuvent avoir une influence non-négligeable sur la performance finale. Ils suggèrent, par conséquent, que LSI tel qu'ils le présentent (avec un simple filtrage des mots trop rares et communs) devrait être précédé et/ou suivi d'autres techniques de recherche d'information pour obtenir un système complet avec des performances supérieures à l'état de l'art sur tous les corpus de test.

Certains problèmes ne sont évoqués que brièvement et restent à approfondir:

- introduction de nouveaux documents dans la base sans ré-effectuer la SVD entière ;
- longueurs inhomogènes des différents documents (ou comment relativiser l'importance de textes très longs) ;
- choix du nombre *k* de thèmes à conserver.

[19] montre qu'un effort sur l'étape de prétraitement permet de gagner sensiblement en performances. En particulier, l'utilisation de comptes modifiés par la fonction log et une pondération par un coefficient d'entropie permettent de rendre LSI compétitif avec l'état de l'art en recherche d'information.

L'utilisation de la décomposition en valeurs singulières de la matrice de comptes peut

être réalisée dans d'autres buts que l'indexation. Par exemple, il s'agira pour nous d'obtenir les thèmes dominants dans le corpus, chacun étant associé à un sous-espace singulier. Nous parlerons donc plutôt de LSA *(Latent Semantic Analysis)*, ce qui désigne simplement l'opération de décomposition en valeurs singulières sans but d'indexation. Il est surprenant de constater que l'étude de LSA a profité à bien d'autres domaines, parfois éloignés de l'indexation. Citons notamment :

- en psychologie/sciences cognitives, l'application de LSA peut être reliée à l'énigme dite «de Platon», qui est que la vitesse d'acquisition du vocabulaire par les individus, et en particulier les enfants, semble nettement supérieure au nombre de nouveaux mots rencontrés par jour. [42] montre, par des résolutions automatiques de tests de type TOEFL avec des corpus censés imiter l'ensemble des textes auxquels a pu être exposé un enfant à un âge donné, qu'une grande partie de l'information sur des mots inconnus peut être inférée de façon contextuelle au moyen d'un modèle comme LSA;

- pour la résolution de problèmes *(problem solving)*, les mots étant remplacés par des actions dans un monde clos et les documents devenant des séquences de résolution du problème issues de l'ensemble d'apprentissage, l'apport de LSA est discuté dans [56].

L'article [43] recense l'ensemble des applications de LSA, des questionnaires de synonymie à la correction automatique de rédactions. D'autre part, l'université du Colorado à Boulder a développé une page web http://lsa.colorado.edu sur laquelle il est possible de tester en ligne l'analyse sémantique latente sur divers corpus (dont certains en français) notamment pour trouver des associations de mots ou effectuer des comparaisons de termes ou de documents.

[53] essaie de justifier théoriquement l'utilisation de LSA pour la recherche d'information. En proposant un modèle génératif de corpus, le théorème principal de l'article prouve que, lorsque les vocabulaires propres à chaque thème contiennent suffisamment de mots, les vecteurs associés à deux documents donnés forment, dans l'espace latent, un angle très faible s'ils appartiennent au même thème et presque droit s'ils appartiennent à deux thèmes différents et ce, avec une probabilité d'autant plus forte que le nombre de documents considéré est grand.

La preuve est d'abord faite dans le cas où les thèmes sont complètement disjoints puis étendue à un cas plus général grâce à un théorème concernant la stabilité de la décomposition en valeurs singulières à de faibles perturbations (c'est-à-dire dont la norme

est contrôlée). Mais, même dans ce cas, les hypothèses requises par l'article:

- disjonction des vocabulaires associés à chaque thème;
- étalement d'un thème sur un nombre suffisamment grand de mots dominants;
- grand nombre de textes dans le corpus semblent encore très fortes par rapport à la qualité connue des performances de LSA dans des cas bien moins favorables. Pour assouplir un peu la dernière condition, les auteurs montrent dans la dernière partie de l'article qu'en appliquant LSA sur un sous-ensemble de documents choisis au hasard *(random projection),* on obtient une approximation satisfaisante de la classification obtenue avec le corpus entier.

2.2.4 Regroupement spectral, méthodes à noyaux

L'analyse sémantique latente conduit à effectuer une décomposition en valeurs propres (ou analyse en composantes principales) sur la matrice des produits scalaires des vecteurs de comptes *(CTC).* Le *regroupement spectral* part d'un point de vue plus général, en supposant donnée une matrice *W* de dimensions n_D X n_D d'affinité entre éléments. Dans le cas de LSA, l'affinité entre deux textes d et d' était donc $\sum_{w=1}^{n_w} C_{wd}\, C_{wd'}$ alors que le regroupement spectral choisit plus volontiers des définitions telles que $w_{dd'} = e^{-|\, c_d - c_{d'}\, |^2/2\sigma^2}$, plus adaptées à la segmentation d'image, application d'origine de cette famille d'algorithmes. Le principe général est de grouper les données à partir de l'étude des vecteurs et valeurs propres de *W.* Les différentes méthodes de regroupement spectral diffèrent par la matrice considérée, les étapes de normalisation et le ou les sous-espaces propres utilisés [72].

L'idée générale est décrite de façon synthétique dans [35]. La première étape consiste à procéder à une décomposition de *W* en composantes principales et à garder autant de vecteurs propres u_1,..., u_{nT} que de groupes souhaités, en les classant naturellement par valeurs propres décroissantes. Si l'on note ensuite W_d la d-ième ligne de *W,* le regroupement consiste à projeter chaque ligne dans les sous-espaces propres et à affecter l'élément *d* au thème *t* qui maximise le produit scalaire $u_t^T w_d$. Un prétraitement classique est de normaliser la matrice d'affinité par la somme des poids affectés à chaque document. Cette opération permet de tenir compte de l'aspect des données dans leur globalité, pour éviter les groupes trop petits, résultant de particularités locales [61]. D'autres variantes [49] suggèrent d'appliquer des méthodes d'affectation plus évoluées aux données projetées dans les sous-espaces propres, par exemple un algorithme des K-means.

Lorsque le choix de l'affinité est particulièrement bien adapté aux données, l'intérêt de la

démarche est très clair: alors que les regroupements n'étaient pas visuellement apparents dans l'espace de départ, ils peuvent devenir évidents dans un espace d'arrivée judicieusement choisi [49]. Cette idée, connue sous le nom d'« astuce du noyau» *(kernel trick),* et popularisée dans l'application des séparateurs à vaste marge *(support vector machines)* [6], permet d'exporter avec succès à des cadres non linéaires des algorithmes a priori conçus pour des données linéairement séparables [60].

L'idée générale des méthodes à noyaux est que certains algorithmes n'ont besoin d'accéder aux données que par le biais des produits scalaires entre exemples. Ainsi la matrice symétrique des produits scalaires entre chaque paire d'exemples (dite *matrice de Gram)* regroupe l'ensemble des informations à propos du corpus d'apprentissage dont ces méthodes ont besoin. Par conséquent, si nous sommes capables de « déformer» la structure de l'espace en proposant de nouveaux produits scalaires entre exemples (c'est-à-dire un nouveau *noyau),* plus pertinents que le produit scalaire classique, nous pouvons également transposer tous ces algorithmes initialement dédiés au cas linéaire. L'astuce du noyau a été utilisée dans un très grand nombre d'applications en apprentissage statistique.

[11] décrit une autre façon d'utiliser les résultats de LSA dans le cadre des méthodes à noyaux que le regroupement spectral. Alors que, dans ce dernier, l'enjeu est de trouver un espace pertinent dans lequel appliquer l'analyse en composantes principales, l'idée de [11] est d'utiliser LSA comme un prétraitement permettant de définir un noyau sémantiquement pertinent à d'autres fins. Les auteurs décrivent d'abord brièvement certains modèles utilisés en analyse textuelle sous l'angle «noyaux», notamment le *modèle vectoriel* [57] et des variantes plus évoluées prenant en compte les cooccurrences entre termes ou les relations synonymiques. Puis ils remarquent que le processus de décomposition en valeurs singulières de LSA permet de calculer une nouvelle similarité entre documents (dans l'espace latent) et donc un nouveau noyau: le *noyau latent sémantique.* L'intuition de ce travail est déjà présente dans [13] mais elle est ici expliquée de façon plus systématique puis utilisée en composition (noyaux polynomiaux ou gaussiens) dans des séparateurs à vaste marge (SVM). Conscients des difficultés numériques liées à l'application de la décomposition en valeurs singulières en grande dimension, les auteurs proposent un algorithme approximatif utilisable dans des cas réels, c'est-à-dire lorsque le nombre de documents est grand. Cet algorithme est ensuite modifié dans les dernières sections par une heuristique accordant plus d'importance aux documents les plus pertinents.

Les résultats sont mitigés: bien que l'utilisation de noyaux spécifiques donne des performances en général au moins aussi bonnes que le noyau linéaire, elles ne font beaucoup

mieux que dans certains cas très précis et la lecture de l'article ne permet pas de déterminer clairement dans quelles conditions. Cette étude demeure intéressante car elle présente une utilisation originale de LSA et, plus généralement, démontre en quoi une méthode d'analyse non supervisée peut fournir un cadre de travail plus général pour l'apprentissage, en préambule à d'éventuelles méthodes supervisées.

Pour conclure sur le regroupement spectral et les méthodes à noyaux, malgré quelques tentatives de noyaux plus évolués, il est difficile d'obtenir pour le texte des performances significativement meilleures que le noyau linéaire [33]. Devant les difficultés pour choisir un noyau, la solution d'en apprendre un parmi une famille de façon supervisée à partir de corpus a été proposée pour d'autres domaines [2]. Mais, là encore, l'applicabilité de la méthode à l'analyse de données textuelles semble compliquée à établir.

2.2.5 Factorisation en matrices non-négatives (NMF)

L'application de la factorisation en matrices non-négatives [45] ne produit pas directement un partitionnement mais consiste à trouver une approximation matricielle optimale (caractéristique qu'elle partage avec LSA) sous une contrainte de positivité. Il s'agit de déterminer deux matrices W et H de dimensions respectives n_w X n_T et n_T X n_D et de termes tous positifs ou nuls telles que :

$$C \approx WH$$

Chaque colonne de la matrice de comptes C_d est approchée par une somme pondérée (avec uniquement des poids positifs) de vecteurs sur le vocabulaire, les colonnes de W, qui représentent les «thèmes ». La matrice H, qui contient les pondérations, représente 1'« influence» de chaque thème dans les documents.

Le symbole ≈ est volontairement vague ici puisque la notion d'approximation est équivalente à celle de minimisation d'une fonction de cotît et que Lee et Seung en proposent deux pour deux matrices *A* et *B :*

- La distance euclidienne: $A - B^{\ 2} = {}_{i,j}\ A_{ij} - B_{ij}^{\ 2}$
- La divergence de Kullback-Leibler : $D\ A||B = {}_{i,j}\ A_{ij} \log \frac{A_{ij}}{B_{ij}} - A_{ij} + B_{ij}$ avec les prolongements par continuité classiques et en admettant que cette divergence puisse être infinie.

Le fait de proposer deux distances permet de choisir celle qui semble le mieux convenir au

problème, ou, le cas échéant d'en rejeter une qui est manifestement inadaptée. Par exemple, une critique à l'encontre de LSA, formulée notamment par Hofmann [25], est que la minimisation de la distance euclidienne repose sur des hypothèses sous-jacentes de normalité sur les données qui ne semblent pas forcément justifiées dans le cadre de matrices de comptes d'occurrences de mots. C'est pourquoi il lui préfère la divergence de Kullbaek-Leibler.

Les théorèmes fondamentaux de l'article donnent des règles multiplicatives desquelles se déduisent des algorithmes itératifs pour minimiser localement chacune des distances.

- La distance euclidienne IIC -WHII décroît si l'on applique itérativement les règles de mise à jour :

$$H_{td} \leftarrow H_{td} \frac{W^T C_{\ td}}{W^T W H_{\ td}}$$

$$W_{wt} \leftarrow W_{wt} \frac{C H^T_{\ wt}}{W H H^T_{\ wt}}$$

- La divergence D(CIIWH) décroît si l'on applique itérativement les règles de mise à jour :

$$H_{td} \leftarrow H_{td} \frac{\sum_{w=1}^{n_W} W_{wt} C_{wd} \quad WH_{\ wd}}{\sum_{w=1}^{n_W} W_{wt}}$$

$$H_{wt} \leftarrow H_{wt} \frac{\sum_{d=1}^{n_D} H_{td} C_{wd} \quad WH_{\ wd}}{\sum_{d=1}^{n_D} H_{td}}$$

Lee et Seung remarquent que ces règles de mise à jour, qui en apparence ne sont pas liées à l'algorithme de descente de gradient, consistent en fait à suivre le gradient selon un pas garantissant les contraintes de positivité et de croissance de la fonction d'objectif.

Le reste de l'article établit les preuves de convergence. On remarque d'abord que l'on peut minimiser itérativement en *H*, à *W* constant, et en *W*, à *H* constant. On définit ensuite une fonction auxiliaire à deux variables *G* telle que, si F est la distance à minimiser,

$$\forall h, h' \quad \begin{matrix} G\ h, h' & \geq F\ h \\ G\ h, h & = F\ h \end{matrix}$$

avec *G* telle que *G(h,h') - F(h)* se réduit à une fonction quadratique positive de *(h –h').*

On peut constater que, dans la multiplication matricielle de *W* par *H*, pour $t = 1,...,n_T$, les éléments de la *t-ième* colonne de *W* sont toujours multipliés par ceux de la *t-ième* ligne de *H.*

On peut multiplier la ligne *t* de *H* par n'importe quel réel non nul λ_t sans changer le produit final, pour peu que l'on multiplie la colonne correspondante de *W* par $\frac{1}{\lambda_t}$ Il est possible de normaliser soit les lignes de *H* soit les colonnes de *W* pour qu'elles somment à 1. Dans le premier cas, on peut supposer que les éléments de *H* représentent la probabilité d'appartenance à un thème pour un document donné. Dans le second cas, il s'agit de la probabilité pour chaque mot de représenter un thème. D'où une interprétation assez intuitive des résultats dans le cadre de la fouille de textes: la matrice *H* définit un partitionnement probabiliste des documents sur n_T dimensions.

[45] reste peu disert sur les aspects pratiques de mise en œuvre de l'algorithme, que ce soit sur les spécificités des divers domaines d'application, les stratégies d'initialisation, de normalisation et de choix du nombre de dimensions latentes (ici la dimension commune de W et H).

[73] aborde directement l'application de NMF à la classification non supervisée de textes, en mettant l'accent sur une différence importante avec LSA : les sous-espaces latents ne sont pas forcément orthogonaux et sont susceptibles de recouvrements, ce qui est souhaitable pour de nombreux corpus. Les performances de l'algorithme sont comparées à des méthodes de regroupement spectral sur deux corpus au moyen d'un score de cooccurrences (après une étape d'appariement des groupes) et d'un score d'information mutuelle. Les capacités de classification de NMF semblent plus ou moins équivalentes à celles des algorithmes de regroupement spectral retenus, les résultats étant améliorés par une normalisation analogue à celle de la matrice d'affinité.

[70] donne également des résultats pratiques des performances de NMF sur une tâche de recherche d'information, montrant sa supériorité sur LSI en termes de précision/rappel de documents pertinents. Une des raisons avancées par Vinokourov en est que NMF encourage les représentations creuses (c'est-à-dire contenant beaucoup de zéros) et cette représentation est plus fidèle à la matrice de comptes, qui est également creuse. Le même article propose une façon probabiliste de retrouver les équations de mise à jour de NMF, avec l'hypothèse d'un bruit poissonien et de distributions a priori laplaciennes pour *W* et *H*, par une approximation variationnelle de la distribution initiale. Cette première tentative d'analogie entre NMF et d'autres algorithmes de classification non supervisée sera suivie de preuves de similarités fortes avec, notamment, le regroupement spectral [16] et PLSA [21].

NMF a fait l'objet d'une attention considérable depuis son introduction en 1999, en

particulier dans le domaine du traitement de l'image, où la décomposition d'un objet en plusieurs «parties» est intuitivement attractive. Un problème reste néanmoins préoccupant, il s'agit de l'absence théorique d'unicité de la décomposition. Il est facile de vérifier que toute permutation identique des colonnes de *W* et des lignes de *H* laisse le produit inchangé, tout comme la multiplication simultanée de la colonne *t* de *W* par une constante $\alpha_t > 0$ et de la ligne *t* de *H* par $\frac{1}{\alpha_t}$ Cette dernière remarque offre en fait une marge de manœuvre pour normaliser *H,* ou *W,* suivant l'application et l'interprétation souhaitées [73]. Mais, plus généralement, à partir d'une solution *W et H,* il suffit de trouver une matrice *A* (n_T X n_T) inversible et telle que $W = WA$ et $H = A^{-1}H$ soient encore positives pour obtenir une autre solution, potentiellement non trivialement différente, et telle que $WH = WH$. [18] s'interroge sur ce problème d'unicité de la décomposition (aux permutations et à l'échelle près) d'un point de vue théorique et propose une preuve dans un cas particulier de décomposition « par morceaux» qui semble très orienté vers le traitement de l'image. Un autre résultat d'unicité concerne une extension de l'algorithme NMF pour l'obtention de résultats dans le cas de matrice creuse *(sparse NMF* [26]). Les conditions requises sont plus faibles que la restriction à une famille mais restent relativement limitées (n_T = 2 ou $W^{-1}W \geq 0$ par exemple) [66]. Cette non-unicité de la décomposition reste donc un problème partiellement ouvert: dès lors que la solution préconisée est celle d'un algorithme itératif convergeant vers un minimum local et qu'il est difficile de proposer des stratégies d'initialisation exemptes de tout reproche, la présence de différents points de convergence possibles pose de sérieuses questions de stabilité et d'interopérabilité des résultats.

2.2.6 Goulot d'information

La méthode du « goulot d'information» *(information bottleneck)* [63] tire son inspiration de la théorie de l'information et semble donner de bons résultats pour l'analyse exploratoire. Le goulot d'information suit deux idées principales:

- les problèmes de sélection d'attributs sur le vocabulaire et de classification non supervisée de documents sont fondamentalement similaires: il s'agit de réduction de dimensionnalité dans les deux cas, selon les deux directions de la matrice mots/documents;
- l'information mutuelle permet de définir un critère mesurant la perte d'information et, donc, la qualité d'un partitionnement (sur les documents comme sur les mots).

Le principe est d'effectuer un regroupement sur la base d'un critère de maximisation

d'information mutuelle d'une variable-cible. Il s'agit ainsi de conserver autant d'information que possible sur la distribution des mots en remplaçant la classification triviale consistant à placer un seul document par groupe (n_D groupes au total) par un regroupement plus restreint de thèmes. La méthode permet, notamment, d'estimer les probabilités des thèmes conditionnellement aux documents qui définissent une classification probabilisée des textes parmi n_T groupes.

Notons W, D et T des variables aléatoires indicatrices respectivement des mots, des documents ou des thèmes (définis au niveau des documents) et à valeurs dans {l, ... , n_w}, {l, ... , n_D} et {l, ... , n_T}. Dans un souci de cohérence avec les modèles probabilistes, nous introduisons également des notations spécifiques pour certaines probabilités:

$$\alpha_t = P\ T = t$$

$$\beta_{tw} = P\ W = w | T = t$$

$$\mu_{dt} = P\ D = d | T = t$$

D'autre part, la loi jointe *P(W = w, D = d)* est supposée connue (ce qui donne aussi, de fait, l'expression des lois marginales et conditionnelles concernant *W* et *D).* Elle sera en pratique estimée par échantillonnage sur la matrice de comptes.

L'information mutuelle entre *T* et *W est* une quantité symétrique mesurant l'information que la connaissance de la variable aléatoire « thème» apporte sur la Variable « mot». Elle est définie par :

$$I\ T; W\ = \sum_{w=1}^{n_W} \sum_{t=1}^{n_T} P\ T = t\ P\ W = w | T = t\ \log \frac{P\ W = w | T = t}{P\ T = t}$$

Le problème est donc de trouver le regroupement thématique défini par les probabilités relatives à $T: \alpha, \beta\ et\ \mu$ telles que l'information mutuelle I(T; W) soit maximisée, tout en contrôlant l'information mutuelle par rapport à *D, I(T; D)* Plus précisément, la quantité minimisée par rapport aux distributions conditionnelles de *T* dans [67] est définie par:

$$\varrho = I\ T; D\ - \lambda I\ T; W\ ,$$

λ étant un paramètre analogue à un «taux de compression» opérant un compromis entre un regroupement des documents le plus synthétique possible *(I(T; D)* petit) et la conservation d'un maximum d'information concernant la distribution des mots *(I(T; W)* grand).

Cette minimisation est accomplie par un algorithme itératif. A λ_t et β_{tw} fixés, μ_{dt} se déduit par l'expression :

$$\mu_{dt} = \frac{\alpha_t}{Z\ \lambda, d}\, e^{-\lambda D_{KL}\ P\ W=w|D=d\ ||P\ W=w|T=t}$$

avec $Z\ \lambda, d\ =\ \sum_{t=1}^{n_T} \alpha_t\, e^{-\lambda D_{KL}\ P\ W=w|D=d\ ||P\ W=w|T=t}$ constante de normalisation et $D_{KL}\ P\ W = w|D = d\ ||P\ W = w|T = t$ divergence de Kullback-Leibler :

$$\sum_{w=1}^{n_W} P\ W = w|D = d\ \log \frac{P\ W=w|D=d}{P\ W=w|T=t} \qquad (1)$$

A μ_{dt} fixé, α_t et β_{tw} sont donnés par :

$$\alpha_t = \sum_{d=1}^{n_D} \mu_{dt} P\ D = d \qquad (2)$$

$$\beta_{tw} = \frac{1}{\alpha_t} \sum_{d=1}^{n_D} P\ W = w, D = d\ \mu_{dt} \qquad (3)$$

On obtient donc un ensemble de mises à jour (1), (2) et (3) qui convergent vers un minimum local de la fonction objectif. Le lecteur familier de l'algorithme Espérance-Maximisation (EM) aura noté une certaine similarité. Concernant l'application pratique de l'algorithme, une hypothèse contestable est l'estimation des probabilités relatives à *W* et *D* par échantillonnage sur la matrice de comptes et, plus précisément,

$$P\ W = w, D = d\ = \frac{C_{wd}}{l}$$

Cette estimation grossière, sans aucune forme de lissage, est probablement mauvaise pour les paramètres associés aux mots rares, pour lesquels on ne dispose que d'un nombre réduit d'observations.

La deuxième remarque importante concernant la méthode du goulot d'information est que les rôles joués par *W* et *D* sont absolument symétriques d'un point de vue formel et qu'il est donc possible d'utiliser un algorithme itératif analogue pour former des groupes sur les mots. C'est ainsi que [63] introduit la méthode du *double partitionnement (double clustering)* : dans une première étape de l'algorithme, il s'agit d'établir des groupes de mots préservant le plus d'information possible (par rapport au vocabulaire initial) à propos des documents et, par la suite, de former des partitions de documents qui minimisent la perte d'information par rapport à ces groupes de mots. Intuitivement, il s'agit de réduire alternativement les deux dimensions de la matrice de comptes en préservant à chaque étape

le plus d'information mutuelle possible vis-à-vis de l'autre dimension.

Dans la mesure où les optimums trouvés ne sont que locaux, l'initialisation joue un rôle crucial, ce que confirment [63, 62]. Les auteurs ont surtout travaillé dans le cas de la classification «déterministe» et proposent différentes stratégies, reposant sur le tirage de partitions aléatoires et les transferts d'éléments améliorant la fonction objective. Le caractère « ad hoc» de ces heuristiques d'initialisation rend l'application de la méthode moins convaincante. Par ailleurs, le problème du choix de nombre de thèmes n'est pas abordé dans l'application de la méthode et les outils de théorie de l'information ne donnent pas, a priori, de moyen naturel de le régler.

Sur le plan pratique, les performances sont évaluées par similarité avec une classification non supervisée, l'appariement thème/catégorie étant effectué« manuellement» (ce qui n'est possible que lorsque le nombre de thèmes est réduit, et est de toute façon moins satisfaisant que la méthode hongroise). La méthode du goulot d'information produit les meilleurs résultats mais les algorithmes (fondés sur des distances 1:1 et 1:2) auxquels elle est comparée semblent de toute façon peu performants sur les applications textuelles, en témoignent les bons classements de méthodes simples telles que l'algorithme des K-means et une heuristique fondée sur la représentation TFIDF [63]. L'idée du double partitionnement n'en est pas moins digne d'intérêt et les auteurs le démontrent en l'appliquant à tous les algorithmes utilisés. Il en résulte systématiquement un gain en qualité de classification.

[74] teste les bénéfices d'une étape de classification non supervisée des termes en vue d'un objectif final de classification supervisée de documents. L'amélioration de performances sur le c1assifieur bayésien naïf est manifeste mais il est légitime de se demander si l'apport de l'étape de regroupement serait aussi fort sur d'autres méthodes de classification. En effet, les faiblesses du c1assifieur bayésien naïf résident précisément dans son hypothèse simplificatrice d'indépendance entre les mots. En ce sens, l'étape de regroupement permet de moduler cette indépendance en rajoutant à moindre coût un lien entre les termes. Pour un autre algorithme qui exploiterait de façon constructive les liens entre les attributs, il n'est pas évident qu'une réduction de dimensionnalité par le biais de la classification non supervisée serait aussi bénéfique.

L'un des grands atouts de la méthode du goulot d'information est de présenter un cadre unificateur aux deux problèmes a priori éloignés que sont la classification non supervisée et la sélection d'attributs (c'est-à-dire, ici, de vocabulaire). En voyant les mots et les documents comme deux dimensions d'une même matrice de comptes, et en cherchant à réduire chaque dimension en minimisant la perte d'information mutuelle par rapport à l'autre, le goulot

d'information propose une solution combinée et originale à deux problèmes classiques d'apprentissage statistique. Néanmoins, outre le problème d'estimation des probabilités relatives à *W* et *D* par échantillonnage, les méthodes d'inférence présentées restent peu satisfaisantes car très sensibles aux optimums locaux. Ce type de problème n'est pas spécifique à la méthode du goulot d'information. Les similarités fortes entre l'EM pour le mélange de multinomiales et l'algorithme du goulot d'information itératif laissent penser qu'ils sont probablement sujets à des difficultés de même nature.

2.2.7 Autres modèles de classification non supervisée

2.2.7.1 Cartes auto-organisatrices

Les cartes auto-organisatrices *(Self-Organising Maps)* de Kohonen [37] peuvent être utilisées pour l'analyse exploratoire [41]. Il s'agit de produire en dimension 2 (ou 3) un maillage représentatif des données. Cette grille contient un nombre d'unités restreint, typiquement de l'ordre de quelques centaines, et, dans tous les cas, très inférieur à la dimensionnalité de l'espace de départ. Une carte auto-organisatrice est donc un réseau de neurones particulier dans lequel chaque élément de la grille est associé à l'espace de départ par un vecteur de poids. Ces poids sont modifiés itérativement en présentant les exemples d'apprentissage les uns après les autres : les vecteurs de poids des cellules sont ainsi ajustés pour être «rapprochés» des exemples d'apprentissage. Les modifications appliquées sont progressivement diminuées pour converger vers une topologie particulière.

Le résultat obtenu est facile à visualiser puisqu'il correspond à une carte. Pour l'application qui nous intéresse, chaque texte est placé en un point de l'espace et l'on peut identifier rapidement des groupes thématiques. Les critiques essentielles à l'encontre de cette méthode sont qu'elle est très exigeante en temps de calcul et que le domaine de l'analyse exploratoire, intrinsèquement multidimensionnel et lacunaire, ne se prête pas aussi bien que d'autres à une représentation graphique en dimension 2.

2.2.7.2 Classification en hypergraphes

L'article [24] se place dans le contexte des hypergraphes pour proposer un partitionnement de l'ensemble des documents. Un hypergraphe est l'extension d'un graphe dans laquelle une arête peut relier plus de deux sommets. Il s'agit donc plus généralement de la donnée d'un ensemble de points et d'un ensemble de groupements de ces sommets. La

classification recherchée est un hypergraphe particulier où chaque point apparaît dans un groupement et un seul.

Le regroupement d'hypergraphe consiste dans un premier temps à faire émerger des associations au sein des données pour définir les hyper-arêtes. Dans le cas du texte, il pourrait s'agir de déterminer les ensembles de termes le plus souvent en cooccurrences au sein du corpus. Une fois obtenu ce premier hypergraphe, faire de la classification consiste simplement à réduire le nombre d'hyper-arêtes. [24] utilise à cet effet un algorithme de coupe d'hypergraphes, HMETIS, fondé sur une approche de type «diviser-pour-régner». Cependant les premières applications de cette méthode ne sont pas textuelles et il n'est pas évident que les heuristiques utilisées donneraient des résultats satisfaisants dans l'espace de très grande dimension que constitue le vocabulaire.

2.2.7.3 Reconstruction localement linéaire

La reconstruction localement linéaire *(Locally Linear Embedding, LLE)* [59] repose sur une décomposition en valeurs propres comme un certain nombre de méthodes d'analyse de données. L'idée spécifique ici est de préserver au mieux les distances locales entre un point et ses voisins. Ce principe de reconstruction locale est en cela assez proche d'une autre technique classique d'analyse de données: l'échelonnement multidimensionnel *(multidimensional scaling, MDS)* [40], qui vise à préserver les distances entre couples d'observations.

Dans LLE, un plus grand nombre de voisins est considéré, grâce à deux étapes d'analyse: la première vise à construire une matrice carrée *W* de dimensions le nombre d'observations et de diagonale nulle, approximant chaque individu par une somme pondérée d'un nombre limité d'autres individus; une fois obtenu cette matrice *W,* il s'agit de résoudre un autre problème d'optimisation: reconstruire les données dans un espace de dimension nettement inférieure, en respectant au mieux les liens entre les plus proches voisins codés dans *W* (c'est cette deuxième étape qui fait intervenir un problème de détermination de vecteurs propres).

L'efficacité de LLE repose grandement sur la confiance dont on dispose envers l'estimation des plus proches voisins, et, donc, envers la distance sur l'espace de départ utilisée à cet effet.

2.2.7.4 Regroupement discriminant

L'intuition du *regroupement discriminant (discriminative clustering)* [51] est que la tâche de classification non supervisée est subjective dans une large mesure et qu'il est parfois possible de l'orienter au moyen de données externes donnant des informations sur la similarité entre les données. Dans l'exemple traité, la variable auxiliaire est une liste de mots-clés associée à chacun des documents à classer. L'intérêt de la méthode est qu'il est possible d'étendre le regroupement à de nouveaux textes, même si ceux-ci ne sont pas accompagnés de mots-clés. Pour le reste, le regroupement discriminant consiste à minimiser une fonction de distorsion entre la distribution conditionnelle des mots-clé", connaissant les documents et des profils associés à chaque thème. En cela, la méthode présente des ressemblances avec le goulot d'information notamment dans la résolution par un algorithme itératif d'un problème de minimisation d'une moyenne de divergences de Kullback-Leibler.

L'algorithme du regroupement discriminant s'appuie sur une idée originale dans un contexte entre classification supervisée et non supervisée. Néanmoins, sur des cas d'application réels, il est courant d'avoir au moins quelques indications sur la classification à obtenir et de ne pouvoir les représenter concrètement sous la forme de classes de documents. Dans ce cas, des modèles plus souples, comme celui du regroupement discriminant, peuvent constituer un compromis intéressant.

2.2.7.5 Analyse en composantes indépendantes

L'analyse en composantes indépendantes [29] est également appelée *séparation de sources*, d'après son domaine d'application classique : il s'agit dans ce cas de décomposer les voix mélangées de plusieurs locuteurs au sein de signaux provenant d'un certain nombre de microphones placés dans une pièce. Alors que dans l'analyse en composantes principales l'accent est mis sur l'orthogonalité des composantes et l'explication de la variance au sein des données, l'analyse en composantes indépendantes choisit comme critère privilégié l'indépendance des sources.

L'application aux données textuelles, que décrit [38], consiste formellement à décomposer la matrice de comptes en trois matrices: $C = AS + U$, où A de dimensions n_w X n_T est appelée matrice de mélange, S de dimensions n_T X n_v est la matrices des sources (en nombre n_T) et U est une matrice de bruit, avec une distribution à préciser. L'application d'un algorithme particulier dépend ensuite des choix particuliers des distributions sous-jacentes.

Celles qui sont introduites dans [38] sont dérivées d'autres domaines et ne semblent pas être particulièrement pertinentes pour la décomposition de l'analyse en composantes indépendantes, qui s'avère être la même que celle de NMF.

2.3 Modèles probabilistes

Nous abordons maintenant la fouille de textes dans le cadre des statistiques paramétriques, en présentant des modèles de mélange qui diffèrent sur le choix et l'organisation de la structure sémantique au moyen de variables cachées.

En guise d'introduction, nous présentons le modèle unigramme, probablement la modélisation la plus simple pour des données textuelles. Nous abordons ensuite les modèles de classification thématique proprement dits, en commençant par le modèle de mélange de multinomiales qui n'autorise qu'une variable latente de thème par document. L'analyse sémantique latente probabiliste (PLSA) propose plus de flexibilité en modélisant les paires (mot, document) mais ne parvient pas à décrire un modèle complètement génératif. Les modèles d'allocation Dirichlet latente (LDA) et Gamma-Poisson (GaP) corrigent ce défaut par l'introduction de modèles théoriquement mieux justifiés mais peut-être moins intuitifs.

2.3.1 Modèle unigramme

Nous considérons à présent le corpus comme un ensemble de n_D observations multi-variées sur l'espace du vocabulaire et nous cherchons une loi adéquate pour modéliser ces vecteurs de comptes C_d. Deux choix courants sous l'hypothèse du «sac de mots» sont:

- la loi multinomiale de paramètres $l_d,\ \beta_1, \dots, \beta_{nw}$:

$$P\ \ C_d; l_d,\ \beta_1, \dots, \beta_{n_w}\ \ = \frac{l_d!}{\prod_{w=1}^{n_W} C_{wd!}} \prod_{w=1}^{n_W} \beta_w^{C_{wd}}$$

avec $\forall w \in 1, \dots, n_W\ , \beta_w \in 0,1\ \ et\ \ \sum_{w=1}^{n_W} \beta_w = 1$;

- n_W lois de Poisson indépendantes de paramètres $\lambda_1, \dots, \lambda_{nw}$

$$P\ C_d; \lambda_1, \dots, \lambda_{n_w} = \prod_{w=1}^{n_W} \frac{e^{-\lambda_w} \lambda_w^{C_{wd}}}{C_{wd}!}$$

Malgré leur différence apparente, ces représentations sont étroitement liées asymptotiquement, si nous examinons le phénomène mot par mot. En effet, dans l'hypothèse multinomiale, le compte d'un mot particulier de probabilité β_w suit une loi binomiale. Or, sous les hypothèses que le nombre de tirages l tende vers l'infini, que β_w tende vers 0 et que le produit $l\beta_w$ tende vers une constante. λ_w la loi binomiale tend vers une loi de Poisson: pour $K \ll l$,

$$P_{binomiale}\ k = \binom{l}{k} \beta_w^k\ 1 - \beta_w\ ^{l-k}$$

$$= \frac{l\ l-1\ \dots\ l-k-1}{k!} \frac{l^k}{l^k} \beta_w^k\ \ \frac{1}{1-\beta_w}\ ^k e^{l \log\ 1-\beta_w}$$

$$= \frac{l}{l} \frac{l-1}{l} \dots \frac{l-k-1}{l}\ \ \frac{1}{1-\beta_w}\ ^k \frac{l\beta_w\ ^k}{k!} e^{l \log\ 1-\beta_w}$$

$$\sim 11 \dots 11 \frac{\lambda_w^k}{k!} e^{-l\beta_w}$$

$$\sim \frac{\lambda_w^k}{k!} e^{-l\beta_w}$$

$$\sim P_{Poisson}\ k$$

Vue la similarité asymptotique de ces deux modélisations, nous pouvons supposer que ce qui est vrai pour l'une l'est également pour l'autre et choisir la représentation sur d'autres critères que l'adaptation aux données.

Si nous supposons que le corpus est généré par une loi multinomiale sur le vocabulaire, de paramètres $l_d,\ \beta_1, \dots, \beta_{nw}$ et que nous souhaitons estimer les β par maximum de vraisemblance, nous obtenons la formule suivante:

$$\forall w \in\ 1, \dots, n_W\ , \beta_w = \frac{\sum_{d=1}^{n_D} C_{wd}}{l}$$

Ce modèle de tirage indépendant et identiquement distribué des mots selon la même distribution multinomiale est connu sous le nom de *modèle unigramme.* Avec cette méthode, le nombre d'observations qui contribuent à l'estimation d'un coefficient β_w est le nombre d'apparitions du mot d'indice *w* dans le corpus.

Or le vocabulaire a une distribution très particulière qui fait que les paramètres liés aux premiers mots sont très bien estimés (les formes les plus fréquentes apparaissent plusieurs dizaines de milliers de fois dans un corpus de 5000 documents) alors que ceux qui sont associés aux hapax sont tous égaux et considérablement mal estimés à partir d'une unique observation pour chaque terme. Le recours à des techniques de *lissage*, indispensable en pratique, permet d'atténuer en partie ce problème. La technique la plus simple, dite lissage de Laplace, consiste à ajouter une constante $\lambda_\beta > 0$ à chaque compte, l'estimateur devenant alors:

$$\forall w \in 1, \dots, n_W \ , \beta_w = \frac{\sum_{d=1}^{n_D} C_{wd} + \lambda_\beta}{l + n_w \lambda_\beta}$$

2.3.2 Mélange de multinomiales

Il s'agit d'un des modèles probabilistes les plus simples pour générer un corpus multithématique. Il a été présenté dans [50] dans un contexte un peu différent, dans le but d'améliorer les performances d'un classifieur bayésien par l'ajout massif de données non-étiquetées. Ici, nous considérons, a priori, que nous ne disposons d'aucune information de classes et nous estimons les paramètres du modèle génératif pour grouper ensuite les documents suivant les différentes composantes du mélange.

Nous décrivons le modèle dans sa forme la plus simple et détaillons l'application de l'algorithme *Espérance-Maximisation (EM)* [15].

On suppose que les textes sont générés indépendamment. Chaque document (numéroté $D \in 1, \dots, n_D$) résulte de l_d tirages indépendants sur le vocabulaire (hypothèse du « sac de mots») suivant une distribution liée au thème, lequel est une variable cachée tirée une fois par texte. D'où le modèle génératif pour un document :

1. Tirer un thème $T_d \sim MULT \ 1, \ \alpha_1, \dots, \alpha_{n_T}$ où les α_t sont des paramètres tels que $\sum_{t=1}^{n_T} \alpha_t = 1$.
2. Conditionnellement au thème T_d, tirer un vecteur de l_d mots

$$C_d \sim MULT \ l_d, \ \beta_{T_d 1}, \dots, \beta_{T_d n_W}$$

β étant une matrice n_T X n_w de paramètres telle que

$$\forall t \in 1, \dots, n_T \ , \ \sum_{w=1}^{n_W} \beta_{tw} = 1.$$

Les paramètres du modèle sont donc:

$$\Theta = \quad \alpha_t \;_{t=1,\dots,n_T},\ \beta_{tw} \;_{t=1,\dots,n_T, w=1,\dots,n_W}$$

La probabilité d'un document est alors:

$$P\ C_d;\Theta\ = \quad \sum_{t=1}^{n_T} P\ T_d;\Theta\ \ P\ C_{1d},\dots,C_{n_W d}|T_d = t;\Theta$$

$$= \sum_{t=1}^{n_T} P\ T_d = t;\Theta\ \ l_d\,! \prod_{w=1}^{n_W} \frac{P\ w|T_d = t;\Theta\ ^{C_{wd}}}{C_{wd}!}$$

$$= \frac{l_d!}{\prod_{w=1}^{n_W} C_{wd}!} \sum_{t=1}^{n_T} \alpha_t \prod_{w=1}^{n_W} \beta_{tw}^{C_{wd}}$$

Ainsi chaque thème contribue à la probabilité globale du document par sa probabilité a priori α_t et, pour chaque occurrence du texte, par la probabilité β_{tw} d'émission du mot *w* dans le thème en question.

La probabilité du corpus, ou vraisemblance des observations, est obtenue en réalisant le produit de l'expression ci-dessus pour l'ensemble des documents étudiés, par hypothèse d'indépendance. Cependant, il n'est pas possible d'établir directement une expression d'un estimateur de maximum de vraisemblance.

Il est alors d'usage de faire appel à l'algorithme EM. Il s'agit d'un algorithme itératif qui consiste à estimer en alternance les probabilités a posteriori des documents d'appartenir aux thèmes et les paramètres α, β du modèle. Pour cela, on s'intéresse à l'espérance conditionnellement aux observations de la log-vraisemblance *complète* $\mathcal{L}^C$, c'est-à-dire la log-vraisemblance des couples vecteur de comptes C_d et thème T_d, définie par:

$$\mathcal{L}^c = \quad \sum_{d=1}^{n_D} \log P\ C_d, T_d$$

$$= \quad \sum_{d=1}^{n_D} \log P\ T_d\ + \log P\ C_d|T_d$$

$$= \quad \sum_{d=1}^{n_D}\ \log \alpha_t + \quad \sum_{w=1}^{n_W} \log \beta_{T_{dw}}^{C_{wd}}\ + K$$

$$= \quad \sum_{d=1}^{n_D} \quad \sum_{t=1}^{n_T} 1_{\ T_d = t}\ \ \log \alpha_t + \quad \sum_{w=1}^{n_W} C_{wd} \log \beta_{tw}\ \ + K$$

Où *K* est une constante indépendante des paramètres. La notation 1 désigne la fonction indicatrice définie par :

$$1_A = \begin{matrix} 1\ si\ A\ est\ vrai \\ 0\ sinon \end{matrix}$$

Par définition, son espérance est la probabilité de A. En calculant l'espérance conditionnellement aux observations et en supposant que les paramètres sont fixés à des valeurs Θ' issues de l'itération précédente, on obtient donc:

$$E\ \mathcal{L}^c|C;\Theta' = \sum_{d=1}^{n_D}\sum_{t=1}^{n_T} P\ T_d = t|C_d;\Theta' \quad \log\alpha_t + \sum_{w=1}^{n_w} C_{wd}\log\beta_{tw}$$

Les probabilités a posteriori sont données par la formule de Bayes, conduisant, pour

$t \in 1,\dots,n_T$, $d \in 1,\dots,n_D$ à:

$$P\ T_d = t|C_d;\Theta' = \frac{P\ C_d|T_d = t;\Theta'\ P\ T_d = t;\Theta'}{P\ C_d;\Theta'}$$

$$= \frac{P\ C_d|T_d = t;\Theta'\ P\ T_d = t;\Theta'}{\sum_{t'=1}^{n_T} P\ C_d|T_d = t';\Theta'\ P\ T_d = t';\Theta'}$$

$$= \frac{\alpha'_t \prod_{w=1}^{n_W} \beta'^{C_{wd}}_{tw}}{\sum_{t'=1}^{n_T} \alpha'_{t'} \prod_{w=1}^{n_W} \beta'^{C_{wd}}_{tw}} \qquad (4)$$

Nous avons déjà vu l'expression du numérateur dans le calcul: il s'agit de la probabilité jointe du document C_d et du thème *t,* qui se décompose intuitivement en produit de la probabilité a priori du thème *t* et de 1'« importance» dans le thème de chaque occurrence du mot *w* dans le texte considéré. Le dénominateur vient de l'opération de normalisation correspondant à

$$\sum_{t=1}^{n_T} P\ T_d = t|C_d;\alpha',\beta'$$

Il est alors possible d'établir les équations de ré-estimation des paramètres en maximisant $E\ \mathcal{L}^c|C;\Theta'$, avec la technique des multiplicateurs de Lagrange pour normaliser de façon appropriée les paramètres α (le vecteur somme à 1) et β (chaque colonne somme à 1). On obtient, pour $t \in 1,\dots,n_T$, $w \in 1,\dots,n_W$:

$$\alpha_t = \frac{1}{n_D} \sum_{d=1}^{n_D} P\ T_d = t|C_d;\Theta' \qquad (5)$$

$$= \frac{1}{n_D} \sum_{d=1}^{n_D} \frac{\alpha'_t \prod_{w=1}^{n_W} \beta'^{C_{wd}}_{tw}}{\sum_{t'=1}^{n_T} \alpha'_{t'} \prod_{w=1}^{n_W} \beta'^{C_{wd}}_{tw}}$$

$$\beta_{tw} = \frac{\sum_{d=1}^{n_D} C_{wd} P\ T_d = t | C_d; \Theta'}{\sum_{d=1}^{n_D} l_d P\ T_d = t | C_d; \Theta'} \qquad (6)$$

$$= \frac{\sum_{d=1}^{n_D} C_{wd} \frac{\alpha'_t \prod_{w=1}^{n_W} \beta'^{C_{wd}}_{tw}}{\sum_{t'=1}^{n_T} \alpha'_{t'} \prod_{w=1}^{n_W} \beta'^{C_{wd}}_{tw}}}{\sum_{d=1}^{n_D} l_d \frac{\alpha'_t \prod_{w=1}^{n_W} \beta'^{C_{wd}}_{tw}}{\sum_{t'=1}^{n_T} \alpha'_{t'} \prod_{w=1}^{n_W} \beta'^{C_{wd}}_{tw}}}$$

Ces formules ont elles aussi une interprétation intuitive simple si les probabilités d'appartenance $P\ T_d = t | C_d; \Theta'$ sont exactement 0 ou 1 (classification «déterministe », chaque texte « appartient» alors à un thème et un seul) :

- Nous obtenons α_t en comptant le nombre de documents dans le thème *t* puis en normalisant.
- Nous déterminons la nouvelle valeur de β_{tw} en dénombrant le nombre d'occurrences du mot *w* dans les textes correspondant au thème *t,* puis nous normalisons sur l'ensemble des mots.

Cette interprétation peut être étendue au cas où les probabilités d'appartenance ne sont pas binaires (classification «probabiliste»). Chaque texte contribue alors au renouvellement des paramètres en proportion de son « implication» dans le thème.

L'algorithme EM consiste à appliquer, à partir d'une valeur initiale des paramètres, les formules (4), (5) et (6) de façon itérative jusqu'à la vérification d'un critère de convergence.

Lorsqu'un mot *w* n'est jamais observé dans un thème *t*, ces formules conduisent à une estimation nulle pour β_{tw}. De façon similaire au traitement du modèle unigramme, il est alors nécessaire de recourir à des techniques de lissage des estimateurs, pour rendre compte du fait que, même si un mot n'a jamais été vu en conjonction avec un thème donné dans l'ensemble d'apprentissage, son apparition dans ce thème n'est pas totalement impossible (elle est néanmoins de probabilité très faible). Il est d'usage de traiter ce problème par un lissage de Laplace, consistant à augmenter tous les comptes d'une constante (souvent égale à 1). Ce lissage peut être interprété comme correspondant à l'algorithme EM associé au

maximum *a posteriori* (et non plus au maximum de vraisemblance) lorsque les paramètres β sont munis d'une distribution *a priori* de type Dirichlet.

Au final, nous obtenons pour chaque document des probabilités d'appartenance à chaque thème. Même si le modèle est dit « monothématique », il est important de souligner que, dans le cas général, le partitionnement induit n'est pas « déterministe» mais « probabiliste» : chaque texte a une probabilité d'appartenance plus ou moins marquée à chaque thème.

Un des avantages du modèle est qu'il s'adapte de façon très naturelle aux cas supervisé et semi-supervisé [50]. Les variables latentes thématiques de certains documents ne sont alors plus cachées mais observables et ne sont donc pas remises à jour, l'équation (4) n'est appliquée que pour les textes dont l'étiquette est inconnue.

2.3.3 Analyse sémantique latente probabiliste (PLSA)

Nous présentons maintenant le modèle de l'analyse sémantique latente probabiliste (PLSA) [25]. La notation présentée ici est évoquée brièvement par Hofmann comme une paramétrisation équivalente.

Dans le modèle PLSA, chacune des l occurrences est une observation indépendante des autres. Le corpus est donc vu comme un ensemble de couples mot/document *(W, D)*, chaque paire apparaissant C_{wd} fois. On suppose maintenant, pour chaque couple, l'existence d'une variable cachée liée au *thème* et à valeurs dans {1, ... , n_T}. Il s'agit ensuite d'un modèle de mélange classique en supposant que les tirages du mot et du document sont indépendants, conditionnellement au thème. Le modèle génératif consiste à tirer l observations de la façon suivante:

1. Tirer un thème $T \sim Mult\ \ 1,\ \ \alpha_1, \dots, \alpha_{n_T}$ où les α_t sont des paramètres tels que $\sum_{t=1}^{n_T} \alpha_t = 1$.
2. Conditionnellement à *T*, tirer un document $D \sim Mult\ \ 1,\ \ \mu_{T_1}, \dots, \mu_{T_{n_D}}$, μ étant une matrice n_T Xn_D de paramètres telle que $\forall t \in\ \ 1, \dots, n_T\ ,\ \ \sum_{d=1}^{n_D} \mu_{td} = 1$.
3. Conditionnellement à *T*, tirer un mot $T \sim Mult\ \ 1,\ \ \beta_{T_1}, \dots, \beta_{T n_W}$, β étant une matrice n_T Xn_w de paramètres telle que $\forall t \in\ \ 1, \dots, n_T\ ,\ \ \sum_{w=1}^{n_W} \beta_{tw} = 1$.

Les paramètres du modèle sont donc:

$$\Theta = \quad \alpha_{t\ \ t=1,\dots,n_T},\ \mu_{td\ \ t=1,\dots,n_T,d=1,\dots,n_D},\ \beta_{tw\ \ t=1,\dots,n_T,w=1,\dots,n_W}$$

Si l'on note W_i, D_i les variables aléatoires associées aux numéros du mot et du document du tirage $i = 1, \dots, l$ *et* T_i la variable cachée correspondante, la vraisemblance des données (W_1, D_1 , … , W_i, D_i) est alors:

$$v = P\left(W_i, D_i\right)_{i=1,\dots,l}; \Theta$$

$$= \prod_{i=1}^{l} P\left(W_i, D_i\right); \Theta$$

$$= \prod_{i=1}^{l} \sum_{t=1}^{n_T} P\left(T_i = t; \Theta\right) P\left(W_i, D_i \middle| T_i = t; \Theta\right)$$

$$= \prod_{i=1}^{l} \sum_{t=1}^{n_T} \alpha_t \beta_{tW_i} \mu_{tD_i}$$

La matrice de comptes permet de déterminer le nombre d'occurrences de chaque couple, ce qui donne par suite une expression de la log-vraisemblance :

$$\mathcal{L} = \sum_{w=1}^{n_W} \sum_{d=1}^{n_D} C_{wd} \log \prod_{i=1}^{l} \sum_{t=1}^{n_T} \alpha_t \beta_{tw} \mu_{td}$$

Mais, cette expression n'étant pas maximisable directement en fonction des paramètres, nous considérons l'espérance conditionnellement aux observations de la log-vraisemblance complète $\mathcal{L}^c$, c'est-à-dire la log-vraisemblance des triplets (W_1, D_1, T_1 , … , W_l, D_l, T_l), en supposant que le thème correspondant à 1'observation (Wi, Di) est Ti.

$$\mathcal{L}^c = \sum_{i=1}^{l} \log P\left(W_i, D_i, T_i\right); \Theta$$

$$= \sum_{i=1}^{l} \log P\left(T_i; \Theta\right) + \log P\left(W_i \middle| T_i; \Theta\right) + \log P\left(D_i \middle| T_i; \Theta\right)$$

$$= \sum_{i=1}^{l} \log \alpha_{T_i} + \log \beta_{T_i W_i} + \log \mu_{T_i D_i}$$

$$= \sum_{i=1}^{l} \sum_{t=1}^{n_T} 1_{T_i=1} \left(\log \alpha_t + \log \beta_{tW_i} + \log \mu_{tD_i}\right)$$

L'espérance conditionnelle par rapport aux paramètres Θ' issus de l'itération précédente se calcule de façon similaire au modèle simple de mélange de multinomiales :

$$E\ \mathcal{L}^c|C;\Theta'\ = \sum_{t=1}^{n_T}\sum_{i=1}^{l}\sum_{w=1}^{n_W}\sum_{d=1}^{n_D} 1_{W_i=w,D_i=d}\ P\ T_i=t|W_i=w,D_i=d;\Theta' \times\ \log\alpha_t+\log\beta_{tw}+\log\mu_{td}$$

$$= \sum_{t=1}^{n_T}\sum_{w=1}^{n_W}\sum_{d=1}^{n_D} C_{wd}\,P\ T=t|W=w,D=d;\Theta'\ \times\ \log\alpha_t+\log\beta_{tw}+\log\mu_{td}$$

La maximisation de cette expression donne, en ajoutant les conditions de normalisation adéquates avec la technique des multiplicateurs de Lagrange, pour $t=1,\dots,n_T\ ;\ W=1,\dots,n_W\ ;d=1,\dots,n_D$:

$$\alpha_t=\frac{1}{l}\sum_{w=1}^{n_W}\sum_{d=1}^{n_D} C_{wd}\,P\ T=t|W=w,D=d;\Theta'$$

$$\beta_{tw}=\frac{\sum_{d=1}^{n_D} C_{wd}\,P\ T=t|W=w,D=d;\Theta'}{\sum_{w=1}^{n_W}\sum_{d=1}^{n_D} C_{wd}\,P\ T=t|W=w,D=d;\Theta'}$$

$$=\frac{\sum_{d=1}^{n_D} C_{wd}\,P\ T=t|W=w,D=d;\Theta'}{l\alpha_t}$$

$$\mu_{td}=\frac{\sum_{w=1}^{n_W} C_{wd}\,P\ T=t|W=w,D=d;\Theta'}{\sum_{w=1}^{n_W}\sum_{d=1}^{n_D} C_{wd}\,P\ T=t|W=w,D=d;\Theta'}$$

$$=\frac{\sum_{w=1}^{n_W} C_{wd}\,P\ T=t|W=w,D=d;\Theta'}{l\alpha_t}$$

Pour calculer les probabilités a posteriori, il faut utiliser la formule de Bayes, ainsi, pour $t=1,\dots,n_T\ ;\ W=1,\dots,n_W\ ;d=1,\dots,n_D$:

$$P\ T=t|W=w,D=d;\Theta'\ =\frac{P\ W=w,D=d|T=t;\Theta'\ P\ T=t;\Theta'}{P\ W=w,D=d;\Theta'}$$

$$=\frac{P\ W=w,D=d|T=t;\Theta'\ P\ T=t;\Theta'}{\sum_{t'=1}^{n_T} P\ W=w,D=d|T=t';\Theta'\ \ P\ T=\mathrm{t}'''''''';\Theta'}$$

$$= \frac{\alpha'_t \beta'_{tw} \mu'_{td}}{\sum_{t'=1}^{n_T} \alpha'_{t'} \beta'_{t'w} \mu'_{t'd}}$$

Sous cette formule, le modèle est symétrique dans les rôles joués par les mots et les documents, ce qui semble relativement contre-intuitif. C'est pourquoi Hofmann présente le modèle génératif d'une autre façon lorsqu'il l'introduit dans [25], en inversant les tirages du thème et du document:

1. Tirer un document $\mathrm{D} \sim \mathrm{Mult}(1, \mathrm{P}(\mathrm{D}))$.
2. Conditionnellement au document D, tirer un thème $\mathrm{T} \sim \mathrm{Mult}(\mathrm{l}, \mathrm{P}(\mathrm{T}|\mathrm{D}))$.
3. Conditionnellement au thème T, tirer un mot $\mathrm{W} \sim \mathrm{Mult}(\mathrm{l}, \mathrm{P}(\mathrm{W}|\mathrm{T}))$.

Alors que le nombre de paramètres semble différent, il suffit d'appliquer la formule de Bayes pour vérifier que la probabilité jointe d'un document et d'un terme particulier obéit à la même équation, ainsi que, par conséquent, la log-vraisemblance et les formules de ré-estimation. Il est intéressant de remarquer que la probabilité de tirer un document n'est en fait pas ré-estimée car la valeur maximisant la vraisemblance est le nombre de mots dans le document divisé par le nombre de mots dans le corpus, qui est identique d'une itération sur l'autre. Il n'est pas difficile de prouver que cette probabilité est égale à la même constante avec l'autre paramétrage, en décomposant les probabilités a posteriori en fonction des probabilités à l'étape précédente, à l'aide des formules de ré-estimation:

$$\forall d \in 1, \ldots, n_D \; , P\ D = d; \Theta \ = \sum_{t=1}^{n_T} P\ D = d|T = t; \Theta \ P\ T = t; \Theta$$

$$= \sum_{t=1}^{n_T} \mu_{td} \, \alpha_t$$

$$= \sum_{t=1}^{n_T} \frac{\sum_{w=1}^{n_W} C_{wd} \, P\ T = t|W = w, D = d; \Theta'}{l}$$

$$= \frac{1}{l} \sum_{t=1}^{n_T} C_{wd}$$

$$= \frac{l_d}{l}$$

Et une formule équivalente est calculable pour les mots, ce qui est cohérent avec la nature d'un estimateur du maximum de vraisemblance.

Les autres formules de ré-estimation sont exactement identiques pour β et équivalentes pour p et α par la formule de Bayes. La présentation de l'article permet, en pratique, de ré-estimer moins de paramètres mais, étant donné que les formules sont plus compliquées, la complexité est sensiblement la même.

Le lien entre LSA et PLSA est ensuite explicité, la formule clé étant la décomposition convexe de la probabilité d'un mot pour un document précis donné dans «l'espace latent» des thèmes, c'est-à-dire sur les probabilités conditionnellement à un thème donné pour $W = 1, \dots, n_W \; ; d = 1, \dots, n_D$:

$$P\ W = w|D = d; \Theta\ = \sum_{t=1}^{n_T} P\ W = w|T = t; \Theta\ P\ T = t|D = d; \Theta$$

$$= \sum_{t=1}^{n_T} P\ W = w|T = t; \Theta\ P\ T = t; \Theta\ \frac{P\ D = d|T = t; \Theta}{P\ D = d; \Theta}$$

$$= \sum_{t=1}^{n_T} \beta_{tw} \alpha_t \frac{\mu_{td}}{\frac{l_d}{l}}$$

Soit :

$$l_d P\ W = w|D = d; \Theta\ = \sum_{t=1}^{n_T} \beta_{tw} (l\alpha_t)\, \mu_{td}$$

Sous cette forme, nous reconnaissons en effet une analogie avec la décomposition en valeurs singulières de la matrice de comptes, que l'on cherche à approximer par $l_d P\ W = w|D = d; \Theta$ avec U correspondant à β, V à p et $l\alpha_t$ aux n_T valeurs singulières. Dans les deux cas, les valeurs singulières et proportions du mélange représentent 1'« importance» de chaque thème dans le corpus alors que les matrices de passage font le lien entre mots/documents et thèmes.

Toutefois, la différence fondamentale réside dans la quantité optimisée lors du processus d'approximation: pour LSA, il s'agit de minimiser la distance euclidienne alors que PLSA cherche le maximum de vraisemblance. Hofmann énumère également plusieurs avantages de PLSA sur LSA : l'interprétation plus facile des éléments du mélange et la possibilité de trouver le nombre de thèmes optimal grâce à la théorie de sélection de modèles et de contrôle de complexité. Sur le plan pratique, en revanche, il semble que LSA soit plus facile à mettre en œuvre que PLSA. En effet, la décomposition en valeurs singulières est effectuée de manière exacte et permet de trouver un optimum global alors que l'algorithme EM ne

converge qu'itérativement et vers un maximum dont rien ne garantit qu'il soit global. Hofmann nuance sérieusement ces inconvénients en signalant que, dans les expériences conduites, les extrema globaux sont souvent «sur-adaptés» aux données d'apprentissage (on parle de sur-apprentissage ou overfitting) et qu'il est donc bon de ne pas conduire trop d'itérations. D'autre part, la complexité d'une itération de l'algorithme est du même ordre de grandeur $(O(ln_T))$ que LSA et donc les temps de calcul globaux sont comparables si le nombre d'itérations est réduit.

PLSA est ensuite comparé avec des modèles de classification non supervisée qui associent une variable latente thématique à chaque document. Il met l'accent sur la différence notable que, lorsque la formule de Bayes est utilisée pour calculer les probabilités d'appartenance a posteriori d'un document à un groupe, l'appartenance d'un texte à plusieurs thèmes ne réside que dans l'incertitude sur la valeur du thème. Dans le modèle PLSA, en revanche, une variable latente (un thème) différente est associée à chaque observation (mot, document), deux occurrences d'un même mot dans un même document pouvant même être issues de différents «aspects» ou thèmes. Tous les mots d'un document sont tout de même liés, au sens où ils partagent les probabilités a priori $P\ T = t|D = d; \Theta$.

L'évaluation des modèles est faite selon la perplexité sur des données de test. Les résultats montrent que PLSA est toujours plus fidèle aux données que LSA mais l'obtention des probabilités prédictives est sujette à discussion dans les deux modèles. Il est, par conséquent, difficile de dresser des conclusions, comme le souligne Hofmann. S'intéressant au phénomène de sur-apprentissage, il propose une version modifiée de l'algorithme EM avec un paramètre de température qui contrôle la convergence et que l'on estime sur un autre jeu de données.

L'évaluation est prolongée sur une tâche de recherche d'information sur des corpus annotés. PLSA est comparé en termes de précision/rappel à une technique de base de similarité documents/requêtes et à LSA. Cependant, la similarité utilisée n'est pas exactement celle qui serait prédite par PLSA, mais, apparemment, une moyenne des similarités prédites sur chacun des thèmes cachés. La raison pour laquelle cette technique est utilisée au détriment de l'importance relative des différents thèmes dans le corpus n'est pas très claire dans l'article. Hofmann évoque un problème de robustesse.

2.3.4 Allocation Dirichlet latente (LDA)

L'allocation Dirichlet latente (Latent Dirichlet Allocation, LDA) [5] a été élaborée à partir des limites de PLSA. Les deux critiques essentielles de B1ei, Ng et Jordan à l'égard de PLSA sont:

- Le modèle n'est pas un modèle génératif des documents, car la variable aléatoire des textes est une multinomiale qui ne peut prendre comme valeurs que les numéros des textes de l'ensemble d'apprentissage. Si une telle stratégie d'indiçage ne semble pas poser de problème sur les mots, elle est plus discutable sur les documents car ne permet pas d'estimer simplement la probabilité d'un document non vu.
- La complexité du modèle est linéaire en fonction du nombre de documents utilisés pour l'estimation des paramètres. B1ei, Ng et Jordan voient ce problème comme une cause du phénomène de sur-apprentissage observé par Hofmann (qu'ils confirment d'ailleurs dans leurs expériences).

Le modèle génératif proposé doit donc contenir une variable de valeur commune à tous les mots d'un document mais sans pour autant perdre la pluri-thématicité garantie par PLSA. C'est pourquoi il a été suggérer de tirer chaque mot d'un thème, lui-même tiré dans un mélange pour chaque document $d = 1, \dots, n_D$:

1. Tirer les paramètres du mélange $\mu_d \sim Dir(\alpha)$. μ_d est donc un vecteur dont les n_T composantes somment à 1. α est un vecteur de même longueur mais dont les composantes, strictement positives, sont les mêmes pour tous les textes du corpus.
2. Pour chaque mot $i = 1, \dots, l_D$:
   (a) Conditionnellement à μ_d, tirer un thème $T_i \sim Mult(1, (\mu_1, \dots, \mu_{n_T}))$
   (b) Conditionnellement à T_i, tirer un mot $W_i \sim Mult(1, (\beta_{T_i 1}, \dots, \beta_{T_i n_W}))$, β étant une matrice n_T X n_w de paramètres telle que $\forall t \in 1, \dots, n_T$, $\sum_{w=1}^{n_W} \beta_{tw} = 1$.

Les paramètres du modèle sont donc:

$$\Theta = \alpha_{t \ t=1,\dots,n_T}, \ \beta_{t \ t=1,\dots,n_T, w=1,\dots,n_W}$$

On peut également voir ce modèle comme un modèle de mélange de lois discrètes, Comme le note [8], dont les coefficients de mélange μ_d sont tirés indépendamment pour chaque document. Chaque document résulte alors de l_d tirages indépendants selon une loi discrète

de paramètres $\sum_{t=1}^{n_T} \mu_{dt}\beta_{t1}, \ldots, \sum_{t=1}^{n_T} \mu_{dt}\beta_{tn_W}$, c'est à dire que le profil d'occurrences caractéristique de chaque document s'obtient comme une combinaison barycentrique, contrôlée par la variable latente μ_d, des n_T profils de base $\beta_1, \ldots, \beta_{nt}$.

Dans tous les cas, la probabilité d'un document est

$$P(C_d; \Theta) = \int_{\mu} P(C_d|\mu; \Theta)\, P(\mu; \Theta)\, d\mu$$

$$= \int_{\mu} \prod_{i=1}^{l_d} P(W_i|\mu; \beta)\, P(\mu; \alpha)\, d\mu$$

$$= \int_{\mu} \prod_{i=1}^{l_d} \sum_{t=1}^{n_T} P(W_i|T_i = t; \beta)\, P(T_i = t; \mu)\, P(\mu; \alpha)\, d\mu$$

Or, pour μ et α dans le simplexe de dimension n_T -1, pour $w = 1, \ldots, n_w$ et, pour $t = 1, \ldots, n_T$,

$$P(\mu; \alpha) = \frac{\Gamma\left(\sum_{t=1}^{n_T} \alpha_t\right)}{\prod_{t=1}^{n_T} \Gamma(\alpha_t)} \prod_{t=1}^{n_T} \mu_t^{\alpha_t - 1}$$

$$P(T = t|\mu) = \mu_t$$

$$P(W = w|T = t; \beta) = \beta_{tw}$$

Donc :

$$P(C_d; \Theta) = \frac{\Gamma\left(\sum_{t=1}^{n_T} \alpha_t\right)}{\prod_{t=1}^{n_T} \Gamma(\alpha_t)} \int_{\mu} \prod_{i=1}^{l_d} \sum_{t=1}^{n_T} \beta_{tW_i}\mu_t \prod_{t=1}^{n_T} \mu_t^{\alpha_t - 1}\, d\mu$$

$$= \frac{\Gamma\left(\sum_{t=1}^{n_T} \alpha_t\right)}{\prod_{t=1}^{n_T} \Gamma(\alpha_t)} \int_{\mu} \prod_{w=1}^{n_W} \sum_{t=1}^{n_T} \beta_{tw}\mu_t^{c_{wd}} \prod_{t=1}^{n_T} \mu_t^{\alpha_t - 1}\, d\mu$$

Il s'agit d'une fonction hypergéométrique qui n'est pas maximisable directement. Blei, Ng et Jordan utilisent une technique d'inférence variationnelle pour trouver une bonne approximation. Cette méthode, fondée sur l'inégalité de Jensen, consiste à trouver une distribution calculable, plus simple, et d'ajuster ses paramètres afin de minimiser la divergence de Kullback-Leibler avec la distribution visée. Les auteurs proposent ici de simplifier le modèle en enlevant le lien entre μ et T. Il est alors possible de minimiser la distance entre le modèle approché et la distribution initiale par la méthode de Newton-Raphson. Il s'agit de l'étape E de cet algorithme EM « variationnel». Dans l'étape M, on peut ensuite maximiser en α et β la borne inférieure de la vraisemblance ainsi obtenue.

LDA est d'abord évalué formellement par une mesure de perplexité. La méthode est comparée au modèle unigramme simple, à un mélange de multinomiales (un thème par document) et à PLSA sur deux corpus (AP et CRAN). PLSA sans contrôle de convergence est pire que toutes les autres techniques. Avec contrôle de convergence, elle fait mieux que les modèles unigramme et de mélange de multinomiales, qui présentent des perplexités comparables. LDA est encore nettement meilleur et d'autant plus que le nombre de dimensions augmente.

Sur des applications plus concrètes, comme de la classification supervisée ou du filtrage collaboratif, LDA est meilleur que les deux modèles les plus basiques, bien que de façon moins marquée. Malheureusement, le modèle n'est cette fois pas comparé à PLSA.

En conclusion, bien que LDA soit un modèle théoriquement mieux justifié que PLSA, le fait de tirer un thème pour chaque mot semble en contradiction avec une notion intuitive de continuité dans le discours. Comment interprète-t-on une phrase qui contient 10 thèmes différents par exemple, une situation probable si l'on suit la recommandation des auteurs de considérer un grand nombre n_T de thèmes possibles? La complexité du modèle induit aussi des difficultés de calcul, qui obligent à effectuer plusieurs approximations dans l'estimation des paramètres. Cependant, au final, la qualité des performances obtenues est un argument de poids en faveur de LDA, qui en fait un algorithme à inclure dans une série de tests comparatifs sur la classification textuelle non supervisée.

2.4 Discussion

Parmi les modèles probabilistes, PLSA est l'une des méthodes les plus utilisées dans la classification de documents. De nombreuses autres méthodes fondées sur des modèles réellement génératifs ont été développées (GaP, LDA et extensions hiérarchiques) et le sont encore. Cependant, il nous semble qu'elles peuvent toutes être vues comme des extensions du modèle de mélange de multinomiales de base, dans la mesure où elles consistent à ajouter des variables latentes de thèmes.

Les modèles de langage tentent d'introduire une meilleure estimation de la distribution des mots dans les documents la modélisant par une loi multinomiale. Toutefois, Church et Gale (1995) prouvent que ce type de distribution ne reflète pas complètement la réalité. En général, le cadre du maximum de vraisemblance est utilisé pour estimer la probabilité pour un modèle de document de générer la requête. Les idées du modèle algébrique LSI/LSA ont été reprises par pLSI/pLSA, une approche probabiliste de l'analyse sémantique latente. Cette approche définit un modèle génératif de la matrice de comptes selon lequel les mots proviennent de k thèmes mis en relation par des variables latentes de mélange. Pour lever ces problèmes, Blei et al. (2003) proposent Latent Dirichlet Allocation (LDA), un modèle où les mots des documents sont toujours générés par une distribution multinomiale. Néanmoins, les thèmes latents sont considérés comme des variables régissant le comportement des paramètres de la multinomiale à travers une distribution de Dirichlet. Ce modèle implique une estimation coûteuse de ses paramètres sans toutefois apporter les gains en performance escomptés sur une tâche de recherche documentaire […]. Bien que LDA lève certain problème de plSA , néanmoins pLSA reste plus simple à appliquer , sur du LDA, la combinaison des paramètres est plus difficile, car ça n'est pas un algorithme EM normal, l'avantage du LDA est que le résultat final ne dépend pas d'une initialisation, comme pLSA, mais seulement si on utilise une méthode d'estimation du type « échantillonnage de Gibbs », chose qui coûte trop cher en calcul, et chose qu'on ne veut pas faire, donc notre choix c'est porté sur la méthode pLSA.

2.5 Conclusion

Dans ce chapitre, nous avons donné un panorama des différentes méthodes de classification non supervisée, allons des modèles algébriques aux modèles probabilistes, suite à ça, nous avons pu diriger notre choix. Dans le chapitre prochain, nous allons exposer

notre approche, par rapport à la classification automatique dans le cas des données distribuées, nous exposerons en détail la solution proposée, et nous justifierons le chois du protocole Gossip pour le calcul distribué.

Chapitre 3

Approche proposée

3.1 Introduction

Ce chapitre, décrit l'algorithme proposé, pour la classification distribuée. Dans notre travail, nous allons nous intéresser à la classification non supervisée de données distribuées. Concrètement, cela peut se présenter comme suit : supposons qu'on a un ensemble de données distribuées sur différents sites formant ainsi des corpus. Le but est de grouper dans une même classe, les données considérées comme similaires, de telle sorte que les données d'un même groupe soient le plus similaires entre eux, et les données de deux groupes distincts présentent le moins de similarité possible.

Notre solution est basée sur l'estimation statistique distribuée. Dans ce cadre, l'agrégation de multinomial d'une même classe, est estimée à plusieurs nœuds sur différents corpus, est une nécessité à laquelle nous nous intéressons. L'approches proposées pour la fusion de mélanges multinomial exigent uniquement un calcul au niveau de chaque nœud et peu de données transit entre les nœuds. Cette propriété est obtenue en agrégeant le modèle via ces paramètres plutôt que par les données en eux même. Dans notre approche, en suppose que les multinomiales sont estimés indépendamment, nous propageons leurs paramètres de façon décentralisée à l'aide de l'algorithme Gossip, dans un réseau, et agrégeons les multinomiales à partir des nœuds reliés entre eux, pour améliorer l'estimation. Une méthode puissante pour trouver la solution du maximum de vraisemblance pour les modèles avec les variables latentes est la méthode pLSA, qui utilise l'algorithme d'espérance-maximisation (EM) [15, 20, 51]. EM est la méthode la plus utilisée pour estimer les paramètres d'un modèle de mélange.

Dans notre cas nous supposons que les données sont distribuées sur plusieurs sites formant ainsi plusieurs corpus. Le but étant de faire une classification automatique (non-supervisée) et distribuée pour cela notre choix c'est porté sur la combinaison de deux solution : l'apprentissage statistique pLSA pour la classification et l'Algorithme Gossip pour le calcul distribué.

3.2 Apprentissage statistique

Les méthodes statistiques sont souvent employées pour décrire un ensemble de données. Nous pouvons reconnaitre la nature probabiliste de l'information que nous cherchons à traiter, et la forme sous laquelle nous devrions exprimer le résultat. Avec un modèle statistique, on applique la théorie des probabilités et la théorie de la décision pour obtenir un algorithme. Ceci est contraire à l'emploi des données de l'apprentissage simplement pour choisir parmi les différents algorithmes ou utiliser des algorithmes heuristiques, qui ont un sens commun, pour concevoir un algorithme.

L'objectif principal de l'apprentissage statistique est de fournir un Framework pour étudier le problème de l'inférence, qui permet d'acquérir des connaissances, de prédire, de prendre des décisions ou de construire des modèles à partir d'un ensemble de données [68]. Le but de cette théorie est d'offrir une introduction à une partie de la méthodologie de classification moderne et de l'apprentissage statistique.

Différentes stratégies sont utilisées pour concevoir un classificateur en reconnaissance de formes statistique, selon le type d'information disponible sur les densités de la classe conditionnelle. Certains systèmes importants de l'apprentissage statistique sont : le classificateur linéaire [52, 4], la machine à vecteur de support (SVM) [9], la machine à vecteur de pertinence (RVM) [178], les modèles de mélange, le mélange des experts [30] et l'arbre de décision [7].

Nous présentons dans ce qui suit certains thèmes qui sont importants dans les approches statistiques:

- Estimation de densité de probabilité: l'estimation de la fonction de densité de probabilité est un concept fondamental en statistiques. Les estimateurs de densité sont les outils fondamentaux pour l'extraction de l'information incluse dans les données brutes. L'estimation de densité est la construction d'une estimation, basée sur les données observées, d'une fonction de densité de probabilité inobservable. La tâche d'estimer les paramètres s'appelle l'estimation de paramètre.

- Modèles paramétriques: les modèles paramétriques sont gouvernés par un petit nombre de paramètres adaptatifs. L'une des approches les plus simples à l'estimation de densité est de représenter la densité de probabilité $p(x)$ en termes d'une fonction spécifique à partir de laquelle contient un certain nombre de paramètres réglables. Pour appliquer ces modèles au problème de l'estimation de densité, nous avons besoin d'une procédure de détermination des valeurs appropriées des paramètres, pour un ensemble de données observées. Les valeurs des paramètres peuvent ensuite être optimisées afin de donner la meilleure adéquation aux données. Le maximum de vraisemblance, le maximum a posteriori, l'estimation prédictive, la validation croisée et l'espérance maximisation (EM) sont les exemples pour optimiser les modèles paramétriques.
- Modèles génératifs : Les modèles génératifs sont les approches pour manipuler les modèles non-déterministes en décrivant ou en estimant une densité de probabilité sur les variables en question. Un modèle génératif est un modèle pour la génération aléatoire des données observées, le plus souvent quelques paramètres cachés sont donnés. Une distribution de probabilité commune sur l'observation et les séquences d'étiquette est définie par ce modèle. Les modèles génératifs sont utilisés en apprentissage automatique pour la modélisation des données, directement, ou comme une étape intermédiaire à la formation d'une fonction de densité de probabilité conditionnelle. Ceci est connu comme un modèle génératif car ayant cette distribution de probabilité, nous pouvons générer les échantillons de différentes configurations du système.

3.2.1 Principe de l'estimation par maximum de vraisemblance

Rappelons le principe de l'estimation par maximum de vraisemblance (MV).

On dispose de n observations, considérées comme les réalisations de n variables aléatoires indépendantes et identiquement distribuées $X_1, \dots, X_n$ **[58]**.

On se place dans le cadre d'un modèle statistique paramétrique $E_X, \varepsilon, P_\theta \ _{\theta \in \Theta}$: chaque $X_i, i = 1, \dots, n$, suit une loi gouvernée par le vecteur de paramètres $\theta \in \mathbb{R}^d$. Par exemple, les X_i peuvent être des multinomiales, toutes de même loi $N\ \mu, \sigma^2$ inconnue : on a alors $\theta = \ \mu, \sigma^2$, et $\Theta = \mathbb{R}^2$ sauf indication contraire.

On note $L_\theta\ x_i$, $L\ x_i;\theta$ ou plus souvent $L\ x_i|\theta$ dans la littérature anglophone, la densité de X_i. On appelle vraisemblance de l'échantillon, la densité jointe de $X_1, \dots, X_n$:

$$L_n\ x_1, \dots, x_n;\theta\ = \ \prod_{i=1}^{n} L\ x_i;\theta \qquad (1)$$

Dans le cas particulier d'une loi discrète, cela se ramène à :

$$L_n\ x_1, \dots, x_n;\theta\ = P_\theta\ X_1 = x_1, \dots, X_n = x_n\ = \ \prod_{i=1}^{n} P_\theta\ X_i = x_i \qquad (2)$$

Définition 1 (EMV). — L'estimateur au sens MV de θ est :

$$\theta = \arg max_\theta\ L_n\ x_1, \dots, x_n;\theta \qquad (3)$$

C'est donc, à échantillon fixé, la valeur du vecteur de paramètres θ qui rend aussi vraisemblables que possible les observations obtenues **[58]**.

Pour des raisons de commodité analytique, on préfère souvent maximiser $\log(L_n)$ plutôt que L_n. La fonction log étant strictement croissante, cela revient au même, tout en facilitant considérablement les calculs puisque :

$$\log L_n\ x_1, \dots, x_n;\theta\ = \ \sum_{i=1}^{n} \log\ L\ x_i;\theta \qquad (4)$$

et qu'il est généralement bien plus simple de travailler sur une somme plutôt que sur un produit **[58]**.

3.2.2 Analyse sémantique latente probabiliste (PLSA)

La tâche de clustering de documents est un problème important en fouille de données et en apprentissage. La méthode PLSA (pour Probabilistic Latent Semantic Analysis) (Hofmann, 1999), est à présent l'algorithme standard largement employé pour cette tâche de partitionnement dans la communauté de Recherche d'Information. Avec le modèle (PLSA), chaque document d'une collection D est représenté par une distribution de probabilité sur les K valeurs de la variable thématique latente $\alpha \in A = \ \alpha_1, \dots, \alpha_k$, où chaque valeur de α correspond à une distribution de probabilité sur l'ensemble des mots de la collection. Dans le processus génératif correspondant à ce modèle, un document est d'abord choisi suivant la probabilité $P(d)$, ensuite une thématique α est générée avec une probabilité $P(\alpha|d)$, et finalement un mot w est émis suivant la probabilité $P(w|\alpha)$. Dans le cas où les partitions de

documents sont confondues aux thématiques, l'algorithme PLSA peut être utilisé comme un algorithme de clustering de documents.

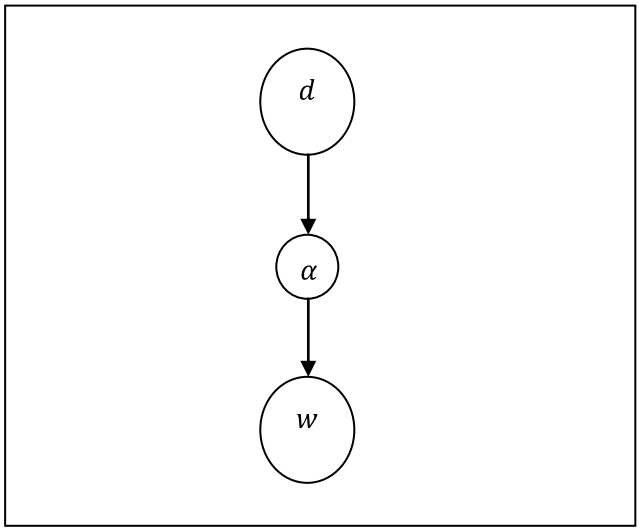


Figure 2. Modèles graphiques correspondants à PLSA.

L'algorithme EM sert à déterminer les paramètres d'une loi de probabilité connaissant des valeurs prises par une variable aléatoire. Dans notre travail, nous adapterons le modèle PLSA au distribué.

3.2.2 Algorithme EM

L'algorithme EM pour Expectation-Maximisation est un algorithme itératif proposé par Dempster et al. (1977). Il s'agit d'une méthode d'estimation paramétrique s'inscrivant dans le cadre général du maximum de vraisemblance **[58]**.

Lorsque les seules données dont on dispose ne permettent pas l'estimation des paramètres, et/ou que l'expression de la vraisemblance est analytiquement impossible à maximiser, l'algorithme EM peut être une solution. De manière grossière et vague, il vise à fournir un estimateur lorsque cette impossibilité provient de la présence de données cachées ou manquantes ou plutôt, lorsque la connaissance de ces données rendrait possible l'estimation des paramètres **[58]**.

L'algorithme EM tire son nom du fait qu'à chaque itération il opère deux étapes distinctes :

- la phase « Expectation », souvent désignée comme « l'étape E », procède comme son nom le laisse supposer à l'estimation des données inconnues, sachant les données observées et la valeur des paramètres déterminée à l'itération précédente ;

- la phase « Maximisation », ou « étape M », procède donc à la maximisation de la vraisemblance, rendue désormais possible en utilisant l'estimation des données inconnues effectuée à l'étape précédente, et met à jour la valeur du ou des paramètre(s) pour la prochaine itération.

En bref, l'algorithme EM procède selon un mécanisme extrêmement naturel : s'il existe un obstacle pour appliquer la méthode MV, on fait simplement sauter cet obstacle puis on applique effectivement cette méthode **[58]**.

Le côté itératif de l'algorithme pourra peut-être paraître un peu mystérieux pour l'instant, mais comme nous le verrons, l'algorithme garantit que la vraisemblance augmente à chaque itération, ce qui conduit donc à des estimateurs de plus en plus corrects.

3.2.2.1 Formalisation d'une itération de l'algorithme EM

Pour formaliser un peu ce qui est exposé en section (3.2.2) :

- nous disposons d'observations i.i.d. $X = X_1, \dots, X_n$ de vraisemblance notée $P(X|\theta)$;
- maximiser $log\ P(X|\theta)$est impossible ;
- on considère des données cachées $Z = (Z_1, \dots, Z_n)$dont la connaissance rendrait possible la maximisation de la « vraisemblance des données complètes», $log\ P\ X, Z\ \theta$;
- comme on ne connaît pas ces données Z, on estime la vraisemblance des données complètes en prenant en compte toutes les informations connues : l'estimateur est naturellement $E_{z|X,\theta_m}\ [log\ P\ X, Z\ \theta\ ]$ (« étape E » de l'algorithme) ;
- et on maximise enfin cette vraisemblance estimée pour déterminer la nouvelle valeur du paramètre (« étape M » de l'algorithme).

Ainsi, le passage de l'itération m à l'itération $m+1$ de l'algorithme consiste à déterminer :

$$\theta_{m+1} = \arg max_\theta\ \ E_{z|X,\theta_m}\ [log\ P\ X, Z\ \theta\ ] \qquad (5)$$

La mécanique itérative de l'algorithme est très astucieuse, et débouche sur une amélioration progressive et réciproque des données cachées Z et de la valeur du vecteur de paramètres θ.

En effet, on démarre l'algorithme avec une ignorance absolue des données cachées Z et en initialisant θ à une valeur θ_0 totalement arbitraire, potentiellement très loin de la

réalité. L'algorithme se sert de θ_0 pour estimer Z, puis se sert de Z pour ré-estimer les paramètres en une valeur θ_1 plus pertinente **[58]**.

À l'itération suivante, on améliore donc l'estimation des données cachées Z puisque cette nouvelle estimation se base cette fois sur θ_1. Et cette meilleure précision sur Z conduit à son tour à une meilleure précision sur θ_2, etc **[58]**.

Au final, l'algorithme EM fournit donc non seulement une estimation de plus en plus pertinente de θ, mais aussi une estimation de plus en plus pertinente de Z. Si l'algorithme est classiquement utilisé pour l'estimation paramétrique, rien n'empêche de le considérer dualement comme une façon d'estimer les données cachées, si tel est notre but. Une autre utilisation de l'algorithme EM peut donc être la complétion de données manquantes **[58]**.
On peut donc définir l'algorithme EM de la manière suivante:

- Initialisation au hasard de θ_0
- m=0
- Tant que l'algorithme n'a pas convergé, faire
- Evaluation de l'espérance (étape E) : $E_{z|X,\theta_m}\ [log\ P\ \ X,Z\ \theta\ ]$
- Maximisation (étape M) : $\theta_{m+1} = \arg max_\theta\ \ E_{z|X,\theta_m}\ [log\ P\ \ X,Z\ \theta\ ]$
- m=m+1
- Fin

En pratique, pour s'affranchir du caractère local du maximum atteint, on fait tourner l'algorithme EM un grand nombre de fois à partir de valeurs initiales différentes de manière à avoir de plus grandes chances d'atteindre le maximum global de vraisemblance.

3.3 Calcul distribué

Le calcul distribué ou répartie est un type de calcul segmenté ou parallèle. Le calcul parallèle traite le développement des programmes où plusieurs processus simultanés coopèrent à l'accomplissement d'une tâche commune. Il est le plus utilisé généralement pour se rapporter au traitement dans lequel les différentes parties d'un programme exécutent simultanément sur deux ou plusieurs processeurs qui font partie du même ordinateur. Le calcul distribué est une méthode de traitement informatique dont les différents composantes et objets comportant une application peuvent être trouvés sur différents ordinateurs connectés à un réseau de communication les uns avec les autres. Les processus séparés ne partagent pas une mémoire commune et communiquent de façon asynchrone les uns avec les autres en passant des messages sur les canaux de communication. Les algorithmes pour tels

réseaux doivent être robustes contre les changements de la topologie. Les exemples sont les données financières rapportées sur Internet, les données de temps observées par un ensemble de capteurs, etc. Un des problèmes avec l'augmentation de données distribuées est de trouver les modèles pour ces données distribuées. Le but dans cet arrangement est de concevoir les algorithmes de telle sorte que les calculs et les communications désirés soient faits aussi rapidement et efficacement que possible. Une approche est de recueillir toutes les données dans un nœud et de trouver le modèle de données entier. Ceci est passible pour les petites données sur un petit réseau mais pour les données qui sont énormes et distribuées sur un grand réseau ne peut pas être fait facilement. En particulier, dans beaucoup d'applications d'exploration de données (data mining) nous sommes intéressés par l'apprentissage d'un modèle global de ces données, comme une distribution de probabilité ou un clustering des données, sans transférer toutes les données à un dépôt central. Idéalement, nous aimerions avoir un algorithme entièrement décentralisé qui calcule et dissémine des agrégats des données, avec un minimum de traitement et de communication et un bon comportement tolérance aux pannes. En fait nous avons besoin d'un system de "l'apprentissage distribué" qui apprend un modèle en utilisant les données distribuées sur le réseau sans rassembler les données entières sur un nœud central.

3.3.1 Algorithmes de rumeur (Gossip)

Les progrès de la technologie de réseau ad hoc, comme les réseaux P2P sur l'Internet ou les réseaux de capteurs, ont mis en évidence la nécessité des moyens efficaces de travailler sur de grands nombres de données qui sont réparties sur un ensemble de nœuds et ont nécessité la conception des algorithmes de calcul et d'échange d'information distribués et tolérante aux pannes. Une approche est d'envoyer les nouvelles données d'un nœud à d'autres nœuds jusqu'à ce que tous les nœuds aient les mêmes données. Ainsi, l'ensemble des données est mis sur tous les nœuds. Ce travail a été présenté par Demers et al. [14]. L'idée est d'utiliser les algorithmes épidémiques pour la mise à jour des objets de données dans une base de données repliée à beaucoup d'emplacements, par exemple, les pages jaunes, les serveurs de noms, ou les annuaires de serveur. Demers et al. proposent les deux concepts suivants :

- anti-entropie : chaque site choisit régulièrement un autre site au hasard et résout toutes les différences en échangeant le contenu complet de la base de données. Il

s'avère que l'anti-entropie est extrêmement fiable mais produit énormément de communication qu'il ne peut pas être utilisé trop fréquemment.

- rumeur colportage (mongering): quand un site reçoit une nouvelle mise à jour, il devient une "rumeur à chaud". Si un site possède une "rumeur chaud", il choisit périodiquement un autre site au hasard et envoie la rumeur à l'autre site. L'idée de la rumeur colportage est d'échanger seulement les mises à jour récentes, ce qui réduit significativement les coûts de communication. Dans la pratique, on peut utiliser une combinaison de ces deux concepts c.-à-d., en utilisant souvent les rumeurs colportage et très rarement l'anti-entropie afin de s'assurer que toutes les mises à jour sont identifiées par tous les sites.

L'idée originale pour la propagation de rumeur était d'envoyer les rumeurs seulement de l'appelant vers l'appelé (la transmission de poussée ou push transmission) [14]. Plusieurs mécanismes d'arrêt ont été étudiés pour décider quand une rumeur devient "froide" pour que sa transmission soit arrêtée. Une autre idée introduite dans [14] est d'envoyer les rumeurs de l'appelé à l'appelant (transmission de traction ou pull transmission). Il a été observé que le nombre de sites non informés diminue beaucoup plus rapidement en utilisant un arrangement de traction au lieu d'un arrangement de poussée. Le travail de Demers et al. a lancé une énorme quantité d'étude expérimentale et conceptuelle des algorithmes épidémiques. Par exemple, il y a une série d'issues de recherches comme la cohérence, l'exactitude, les structures de données, et l'efficacité [1, 23, 38].

3.3.2 L'agrégation de nœud en utilisant l'algorithme Gossip

Dans de nombreux grands systèmes, il est plus important d'avoir les valeurs agrégées du réseau entier que les données à chaque nœud [22, 47, 69] ou en d'autres termes nous souhaitons calculer et répandre les agrégats des données. Par agrégation nous entendons trouver les statistiques sur un ensemble de valeurs numériques qui sont distribuées, comme les valeurs, la moyenne, la somme, le compte, la variance, etc. Ceci est particulièrement nécessaire lorsque la taille des réseaux augmente ce qui motive pour trouver un algorithme entièrement décentralisé avec un minimum de traitement et de communication et un bon comportement de tolérance aux pannes. Rassembler toutes les données en locales dans un nœud spécifique crée un problème de communication ou même un problème de stockage dans ce nœud.

Une solution est d'employer les modèles basé sur des rumeurs pour le calcul [32, 36] qui fournit la robustesse, l'évolutivité et la simplicité. Dans [36] les auteurs montrent comment calculer les échantillons aléatoires en utilisant une solution basée sur la rumeur et décentralisée complètement. Leurs approches répondent également à beaucoup d'autres questions d'agrégation du mode décentralisée. Elles prolongent l'étude de l'agrégation aux sommes et aux moyennes. Cette solution peut être employée seulement pour les petites données et n'est pas appropriée pour les données énormes. Comme la taille des données augmente, il n'est pas facile d'utiliser de tels algorithmes, car la taille des données à transférer sur le réseau est énorme.

3.3.3 Algorithme Gossip de base

Les deux fils suivants s'exécutent en parallèle sur chaque nœud :

Algorithme 1 Fil actif Gossip générique (au nœud n)

Require : une information locale m
Loop
Attendre un temps Δt
Choisir un voisin v au hasard
Envoyer au nœud v l'information m
Recevoir du nœud v l'information s
Incorporer s dans l'information locale
End loop

Algorithme 2 fil passif Gossip générique (au nœud v)

Require : une information locale s
Loop
Recevoir du nœud n l'information m
Envoyer au nœud n l'information s
Incorporer m dans l'information locale
end loop

Périodiquement chaque nœud échange de l'information avec un de ses voisins logiques

- Sélection de la cible
- Communication
- Gestion de l'information reçue

Figure 3. Echange d'information entre deux nœuds.

un nœud n, contenant une information m locale, attend un temps Δt et choisit un voisin v au hasard, lui envoi l'information m, et le nœud v contenant une information locale s, reçoit celle-ci et envoi à son tour son information locale, et chacun des nœuds incorpore l'information reçu à l'information locale.

3.4 Approche proposée

Dans notre cas, nous supposons que les données sont distribuées sur plusieurs sites formant ainsi plusieurs corpus. Le but étant de faire une classification automatique est distribuée, autrement dit même si deux documents n'appartenant pas à un même corpus, peuvent être attribué au même cluster. Pour aboutir à ce résultat notre choix c'est porté sur les méthodes d'apprentissage statistique et plus exactement sur la méthode pLSA (Hoffman .all 1977), et pour le calcul distribué notre choix c'est porté sur l'algorithme Gossip, qui est un algorithme bien adapté pour l'apprentissage distribué asynchrone des modèles de thèmes. Les algorithmes asynchrones offrent plusieurs avantages sur leurs homologues de calcul synchrone :

1- aucune étapes de synchronisation globale n'est nécessaire ;
2- le système est extrêmement tolérant aux panes en raison de son caractère décentralisé ;
3- des machines hétérogènes avec différentes vitesses de processus et de capacités de mémoire peuvent être utilisés ;
4- de nouveaux processeurs et de nouvelles données peuvent être incorporés dans le système à tout moment.

3.4.1 Architecture de l'approche

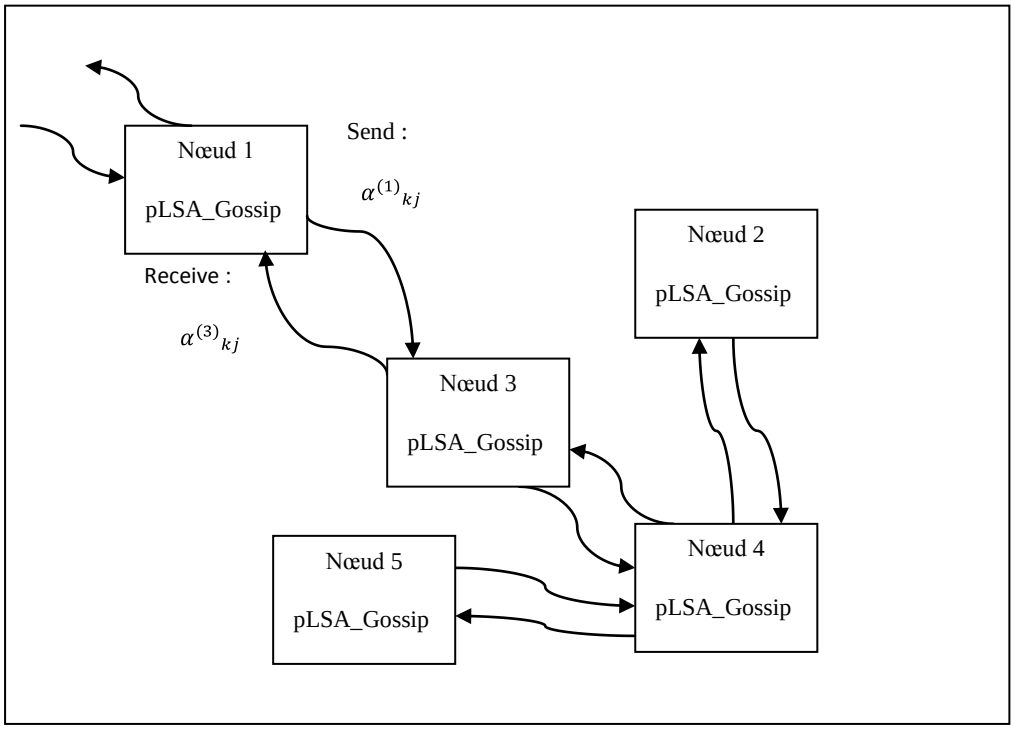


Figure 4. Echange de multinomiales entre les nœuds.

La figure. 4 représente l'échange de paramètre entre les différents nœuds du réseau, dans notre cas un nœud représente un corpus situé sur site donné, chaque nœud calcule ses multinomiales, à travers l'algorithme pLSA_Gossip, en locales, envoie à un de ses voisins choisit au hasard par Gossip sa variable locale, et reçoit en retour la multinomiale complète pour agréger sa variable locale.

3.4.2 Explication de l'Algorithme pLSA-Gossip

Supposons que nous avons deux nœuds, un nœud actif n et un nœud passif v, cet algorithme tourne sur chaque nœud de manière asynchrone, nous aurons comme paramètres d'entrée la matrice termes/ documents pour chaque nœud, qui contient le nombre d'occurrence des termes contenu dans chaque document, les documents sont censée être des colonnes de cette matrice. En sortie on aura comme résultat le produit de deux matrices, une matrice terme/ thème, qui après traitement sera globale à tout les nœuds du réseau, et qui représente la densité de probabilité du terme sachant le thème et cela pour chaque terme et thème du réseau, ainsi qu'une matrice thème/ document pour chaque nœud contenant, la

densité de probabilité du thème sachant le document et cela pour chaque thème et document d'un corpus donné.

Le travail demandé pour pLSA_Gossip estimation répartie asynchrone, est de voir :

- comment réaliser les étapes EM,
- quelles sont les informations qu'il faut transporter.

Algorithme général

- Paramètre d'entrée :

X : matrice (terme, document), les documents sont les colonnes de cette matrice

- Paramètre de sortie :

W : matrice (terme, topic), contient les entrées W (terme,topic)=P(terme|topic) pour chaque terme et chaque topic.

S : matrice (topic,document), contient les entrées S(topic,document)=P(topic|document)

- Initialisation :

Initialisation de S : supposer que les valeurs sont équiprobables

$P\ Z_j|d_{local}, W\ = \frac{1}{K}$, $\forall j$ sachant que K est le nombre de topic

Initialisation de W : supposer que les valeurs de la matrice sont équiprobables.

Pour chaque nœud (sac-de-mot) effectuer :

Tant que l'algorithme n'a pas convergé faire :

Etape E :

- Estimer des probabilités a posteriori d'appartenance du terme W au document d_i
- Calculer

$P\ d|Z_j\ =\ P\ W_{i\ du\ noeud}|\text{multinomial de } Z_j$

- Calculer

$$P\ Z_j|d_i, W\ = \frac{P\ Z_j\ P\ d|Z\ P\ W|Z}{\sum_{z' \in Z} P\ z'_j\ P\ d|z'\ P\ W|z'}$$

Etape M :

- Recevoir les multinomiales des voisins choisis avec le protocole GOSSIP
- La mise à jour des multinomiales

$$P\ Z_j|d_i, W\ = n\ d_i, w\ P\ Z|d_i, W\ + \sum_{j \neq i} n\ d_j, w\ P\ Z|d_j, W$$

- Envoi des multinomiales locales aux voisins choisis par le protocole GOSSIP

Fin

3.4.2 Algorithme pLSA en gossip

L'algorithme 3 représente l'exécution de la solution proposée sur chaque document, pour notre simulation, nous avons supposés que chaque document est un corpus qui est représenté en sac de mots.

Algorithm 3 Algorithme pLSA /document

Require:
$D = d_1, \dots, d_N$: Collection de documents
$W = w_1, \dots, w_M$: Vocabulaire
$n(d_i, w_m)$: Nombre d'occurrences du terme m dans le document i
$n(d_i)$: Nombre de termes dans le document i
Initialiser $p(w_j|z_k) =$ une multinomiale tirée au hasard, pour tout k
Initialiser $p(z_k|d_i) = 1/K$ pour tout i, pour tout k
repeat
Etape E
Calculer, pour tout k, pour tout i, pour tout j :

$$p(z_k|d_i, w_j) = \frac{p(w_j|z_k)p(z_k|d_i)}{\sum_{l=1}^{K} p(w_j|z_l)p(z_l|d_i)}$$

Etape M
Calculer pour tout k, pour tout i, la composition des documents en thèmes

$$p(z_k|d_i) = \frac{\sum_{l=1}^{M} n(d_i, w_j)p(z_k|d_i, w_j)}{n(d_i)}$$

Calculer pour tout k, pour tout j, les nouvelles multinomiales

$$p(z_k|d_i) = \frac{\sum_{i=1}^{N} n(d_i, w_j)p(z_k|d_i, w_j)}{\sum_{m=1}^{M} \sum_{i=1}^{N} n(d_i, w_m)p(z_k|d_i, w_m)}$$

until convergence

Ci-dessous, le schéma représentatif de notre solution, qui est une illustration d'échange de données entre les nœuds, ainsi que le calcule effectué au niveau de chaque nœud, sachant que l'algorithme pLSA_Gossip s'exécute sur chaque nœud individuellement, et effectue l'agrégation des valeurs reçu par les différents nœuds voisins.

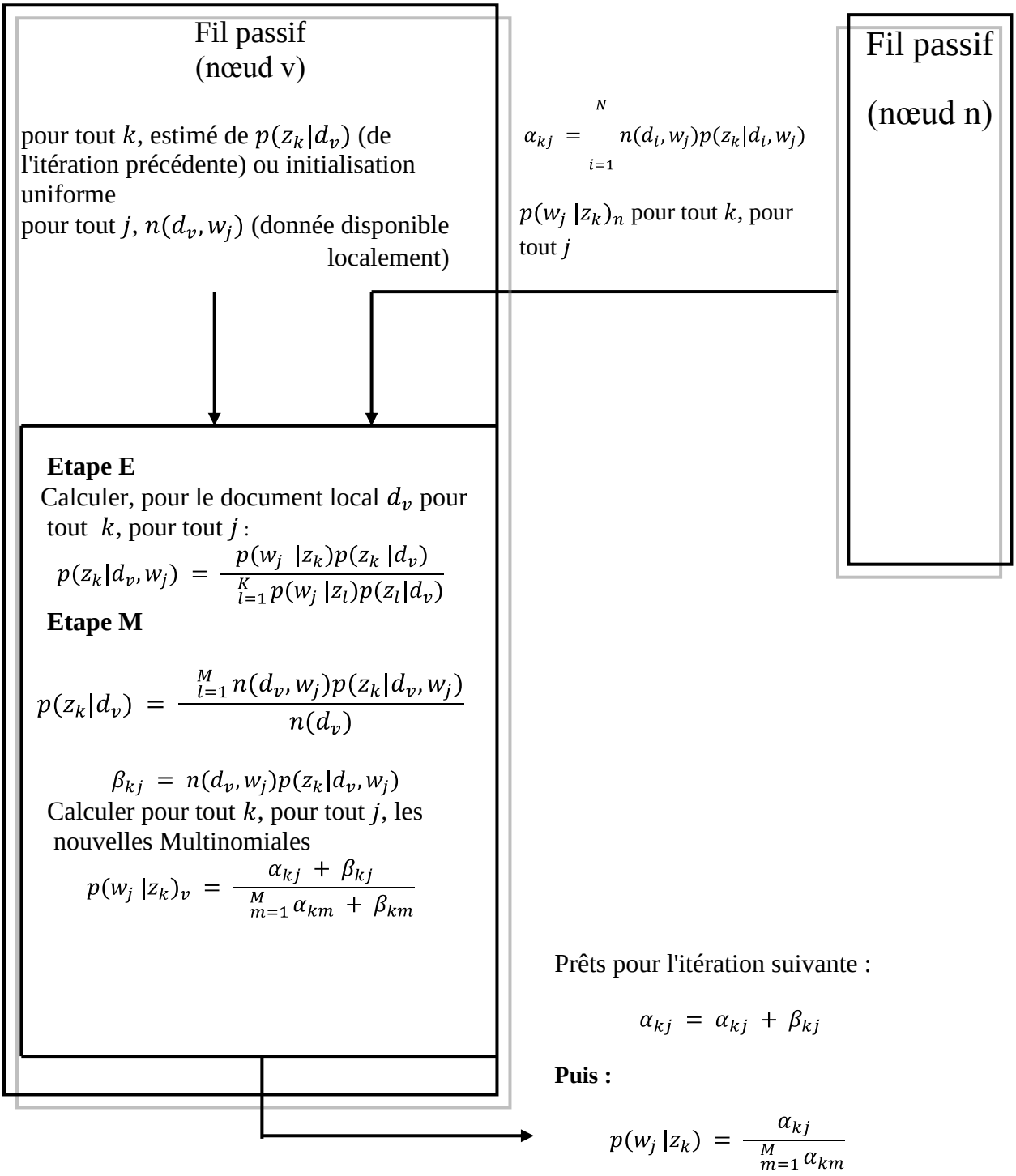


Figure 5. Echanges et calculs à réaliser pour plsa-gossip

Condition

La condition principale est de garder en locale les sac-de-mots et la distribution des topics, et de distribuer les multinomiales, pour l'exécution de la solution les étapes suivantes sont effectuées :

- Application de l'algorithme pLSA_Gossip pour chaque nœud ;
- Choix des nœuds voisins avec le protocole GOSSIP ;
- Echange de multinomial entre les nœuds ;

Dans notre modèle PLSA_Gossip, chacune des n occurrences est une observation indépendante des autres. Le corpus est donc vu comme un ensemble de couples mot/document *(W, D),* chaque paire apparaissant $n(d_i, w_m)$ fois. On suppose maintenant, pour chaque couple, l'existence d'une variable cachée liée au *thème* et à valeurs dans {1, ... , n_T*}*. Il s'agit ensuite d'un modèle de mélange classique en supposant que les tirages du mot et du document sont indépendants, conditionnellement au thème.

Nous avons comme entrée la matrice de comptes, qui permet de déterminer le nombre d'occurrences de chaque couple, ce qui donne par suite une expression de la log-vraisemblance.

Nous initialisons la matrice de probabilité $p(w_j \,|z_k)$ d'un terme w_j sachant le thème z_k qui est une multinomial tirée au hasard pour tout thème et ceux pour la première itération, et nous initialisons aussi la matrice de probabilité $p(z_k|d_i)$ d'un thème z_k sachant le document d_i qui est une multinomial avec des valeurs uniforme pour la première itération, sinon la valeur de l'itération précédente. On utilise l'algorithme Gossip pour le choix des voisins.

Pour l'application de notre algorithme nous supposons avoir deux fils, une fil actif n et une fil passif v, dans notre algorithme pLSA-Gossip on effectue l'étape-E qui consiste à calculer, en local pour chaque nœud pour tout thème z_k et pour tout document d_i du corpus, la probabilité à posteriori pour les variables latentes (thèmes), donc la probabilité du thème sachant le document et le terme,

$$p(z_k|d_i, w_j) = \frac{p(w_j \,|z_k)p(z_k \,|d_i)}{\sum_{l=1}^{K} p(w_j \,|z_l)p(z_l|d_i)}$$

Puis vient l'étape-M qui consiste à maximiser les données attendues complètes avec le log-vraisemblance,

$$p\ z_k\ d_v\ = \frac{\sum_{l=1}^{M} n\ d_v, w_j\ p\ z_k\ d_v, w_j}{n\ d_v}$$

et qui calcule la composition des données en thème, ainsi que de mettre à jour les nouvelles multinomial qui est la composition en terme sachant le thème.

$$p(w_j\ |z_k)_v\ = \frac{\alpha_{kj}\ +\ \beta_{kj}}{\sum_{m=1}^{M} \alpha_{km}\ +\ \beta_{km}}$$

A travers Gossip, choisir un nœud *n* au hasard, lui envoyer les nouvelles multinomiales, d'où la deuxième fil qui est la fil passif, qui reçoit la valeur du nœud n, et l'incorpore dans sa valeurs locale.

$$p(w_j\ |z_k)\ = \frac{\alpha_{kj}}{\sum_{m=1}^{M} \alpha_{km}}$$

Au final nous avons comme résultats une classification globale à tous les nœuds du réseau.

3.5 Conclusion

Nous avons présenté dans ce chapitre, notre approche proposée, pour la classification distribuée. Nous nous sommes concentrer sur le modèle pLSA pour l'Analyse sémantique latente probabiliste, qui se base sur le modèle de mélange multinomiales, dans les techniques d'apprentissage automatique que nous utilisons dans ce mémoire. Ensuite, nous justifions notre choix, qui c'est porté sur la combinaison de deux solution : l'apprentissage statistique pLSA pour la classification et l'Algorithme Gossip pour le calcul distribué. Dans l'approche proposée, nous définissons l'architecture de notre solution ainsi que l'explication de l'algorithme proposé pLSA_Gossip. Dans le chapitre prochain, nous allons expérimenter notre algorithme pLSA_Gossip sur un corpus de test, et en fonction des résultats, nous allons évaluer notre approche.

Chapitre 4

Résultats expérimentaux

4.1 Introduction

Les expérimentations sont organisées de la manière suivante. Nous évaluons dans un premier temps la classification d'un corpus de 348 documents avec la version centralisé standard de la méthode pLSA, suite à ça, nous évaluerons le même corpus en distribué avec la version de notre algorithme pLSA_Gossip. L'objectif est alors d'évaluer le comportement de notre Algorithme et les résultats obtenus par rapport à la version centralisée. Cet apprentissage, aussi bien pour le modèle pLSA et le modèle pLSA_Gossip, a été réalisé à partir d'un corpus contenant environ 348 documents. Pour ces deux modèles, le nombre de thèmes a été fixé à 4.

4.2Environnement de développement

4.2.1 Outils utilisés

Pour implémenter notre algorithme, nous avons utilisé Matlab, beaucoup de méthodes d'apprentissage machine et algorithmes sont facilement mises en œuvre et téléchargeables sur MATLAB. Comme de nouvelles solutions sont en permanence en cours d'élaboration, la plupart de ces méthodes sont écrit en MatLab. MATLAB est un langage de haute performance pour le calcul scientifique. Il est facile à utiliser. Le nom de MATLAB est synonyme de *MATrix LABoratory*. C'est parce que dans MatLab le type de base est la matrice. Vous pouvez effectuer des opérations avec des matrices dans une seule ligne de code.

4.2.2 Corpus

Pour nos résultats expérimentaux, nous avons utilisé une matrice termes/documents qui a été extraite d'un échantillon de la collection de texte de *20Newsgroups*[1], qui est une collection d'environ 20.000 documents, partagée uniformément et répartis dans 20 groupes de discussion différents. Ils ont été recueillis par Ken Lang. Cette collection est devenue un jeu de données populaire pour des expériences dans les applications de techniques d'apprentissage machine, tels que la classification de texte et le regroupement de texte. Notre échantillon contient un dictionnaire de termes associés: fichier de données (100 termes x 348 docs); fichier de dictionnaire (100 termes énumérés). Notre corpus qui contient 348 documents est composé de 4 thèmes, et on sait quels documents sont "en vrai" correspondant à quel thème, les 87 premiers sont pour le premier thème, 87 suivant le second thème ainsi de suite. C'est sur cette base que nous allons évaluer notre algorithme.

4.2.3 Conditions expérimentales

Dans nos expériences, nous allons d'abord évaluer le corpus sur pLSA standard, suite à ces résultats, et sachant bien sûr quel documents correspondant à quel thème, nous allons évaluer notre algorithme pLSA_Gossip sur le même corpus, qu'on distribuera en plusieurs sous-corpus et qu'on partagera. Notre expérience va s'organisé comme suit :

1. application de l'algorithme pLSA standard sur le corpus (centralisé) ;
2. division du corpus en plusieurs sous corpus : comme premier test, nous allons considérer chaque document comme étant un corpus ;
3. application de l'algorithme pLSA_Gossip, sur les 348 corpus (distribués) ;
4. comparaison des résultats obtenus par les deux algorithmes, centralisé et distribué.

4.3 Implémentation plsa_Gossip

Notre programme contient deux fonctions, une fonction initialisation représentée par la figure 6, qui consiste à récupéré toutes les données d'entrée, dans notre cas, en entrée on a la matrice de cooccurrence termes/documents, ainsi que l'initialisation de la matrice de densité

[1] http://people.csail.mit.edu/jrennie/20Newsgroups/

de probabilité termes/topics, qui est une répartition au hasard pour les valeurs initiales, ainsi que la matrice topics / documents, qui est initialisé à 1/le nombre de topics, initialement le nombre de topics a été fixer à 4. Ainsi que la fonction *PLSA_Gossip* représentée par la figure 7, qui est le cœur de notre implémentation :

1. calcule, pour chaque document local dv, pour tout topic, pour tout terme, en considérant un document comme étant un sac de mots qu'on traitera individuellement.
2. Incorpore la multinomiale reçu par le nœud voisin dans l'information locale $p(w_j | z_k)_v = \frac{\alpha_{kj} + \beta_{kj}}{\sum_{m=1}^{M} \alpha_{km} + \beta_{km}}$.
3. Met à jour les nouvelles multinomiales qu'on transmettra au document suivant, et qui à son tour mettra à jour ces multinomiales ainsi de suite $\alpha_{kj} = \alpha_{kj} + \beta_{kj}$

```
function  Initialisation
[m,nd] = size(x);
Pw_z  =  rand(m,K);
Pw_z = Pw_z./repmat(sum(Pw_z),m,1);
for j = 1:nd
   for j = 1:K
    Pz_d(i,j)  = 1/K;
    end
end
Pz_d = Pz_d./repmat(sum(Pz_d),K,1);
Pz   = ones(K,1)/K;
Pz_dw  =  zeros(m,nd,K);
```

Figure 6. Fonction d'initialisation des matrices

```
Fonction PLSA_Gossip
som=0;
Aw_z = zeros(m,K);
%100 iterations par default
if nargin<3, iter=100; end
for i=1:iter
  for j=1:nd
 (1) Etape E  la probabilité à posteriori pour les variables latentes z,
    for k = 1:K
      Pz_dw(:,j,k) = Pw_z(:,k) * Pz_d(k,j)' * Pz(k);
    end
    for k = 1:K
       som = som + Pw_z(:,k) * Pz_d(k,j);
    end
    for k = 1:K
    Pz_dw(:,j,k) = Pz_dw(:,j,k) ./som+eps;
    end
 (2) Etape M, maximisation du log-vraisemblance
     for k = 1:K
      Pz_d(k,j) = sum(x(:,j).* Pz_dw(:,j,k),1)';
     end
    Pz_d(:,j) = Pz_d(:,j)./sum(x(:,j));
    Pz_d=Pz_d./repmat(sum(Pz_d),K,1);
     C = sum(Pz,2);
     Pz = Pz .* 1./C;
     for k = 1:K
          beta(:,k) = (x(:,j)+eps) .*Pz_dw(:,j,k);
     end
     for k = 1:K
       Pw_z(:,k) = Aw_z(:,k) + beta(:,k) ./sum((Aw_z(:,k)+eps),1);
     end
     Pw_z=Pw_z./repmat(sum(Pw_z),m,1);
     Aw_z = Aw_z + beta;
  end
end;
```

Figure 7. Fonction PLSA_Gossip

4.3 Résultat obtenus

Dans ce qui suit, nous allons discuter des résultats obtenus :

- La figure.8 représente la matrice d'entrée termes/documents qui est l'appartenance des termes aux documents avec des densités différentes, selon le niveau de gris, un terme peu appartenir à plusieurs documents avec des densités de probabilité différente.

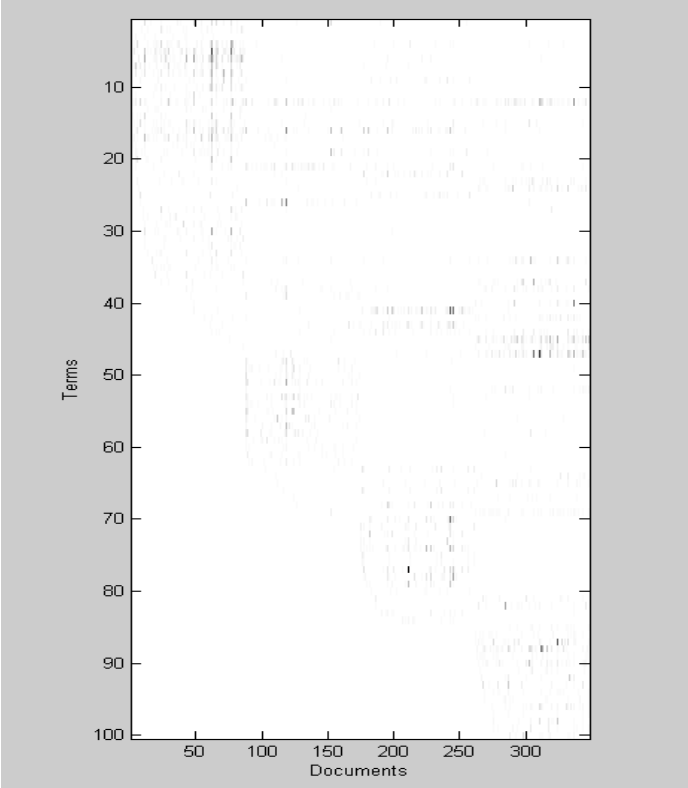


Figure 8. Représentation graphique de la matrice termes/documents.

- La figure.9 représente le résultat de la matrice termes/topics, qui est la classification des termes par rapport aux topics avec des densités différentes, selon le niveau de gris, un terme peu appartenir à plusieurs topics avec des densités de probabilité différentes.

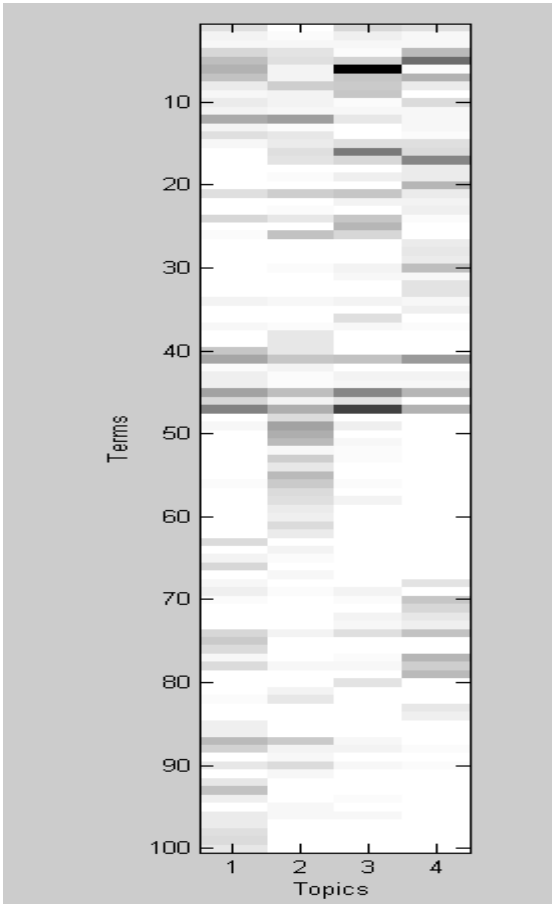


Figure 9. Représentation graphique de la matrice termes/topics

- Dans la figure.10 le résultat de la matrice topics /documents qui représente l'appartenance des documents aux topics.

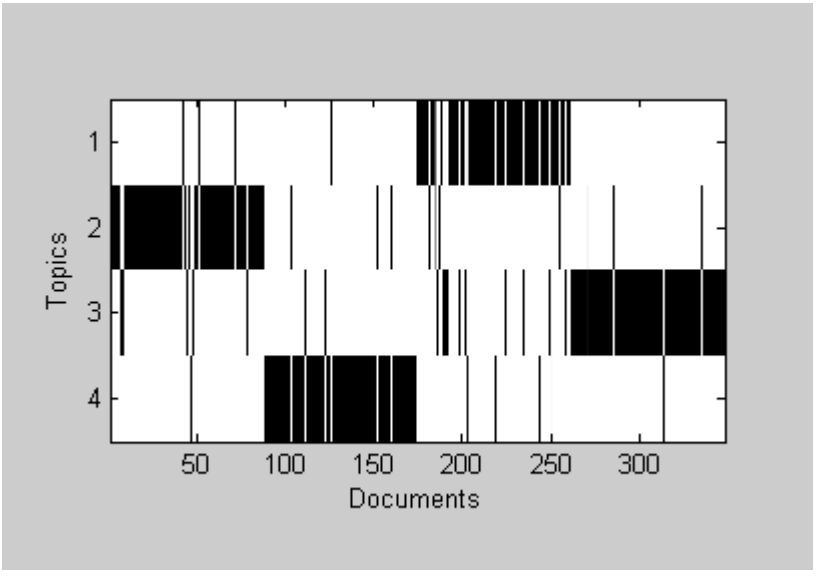


Figure 10. Représentation graphique de la matrice topics /documents.

4.3.1 Simulation de pLSA standard

Résultats obtenus par la méthode pLSA centralisé standard (Figure 2**).**

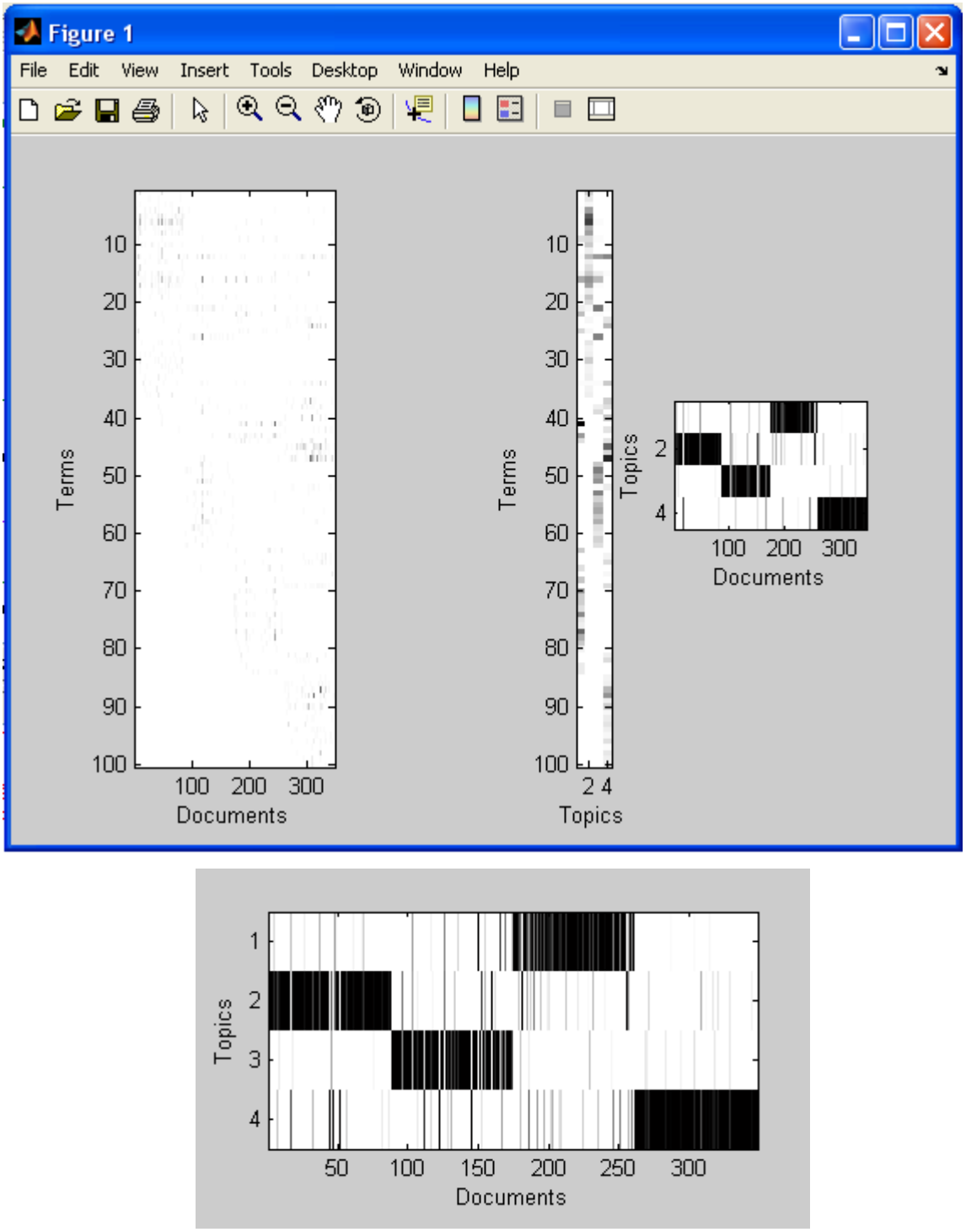


Figure 11. Résultat obtenu par l'algorithme centralisé standard pLSA.

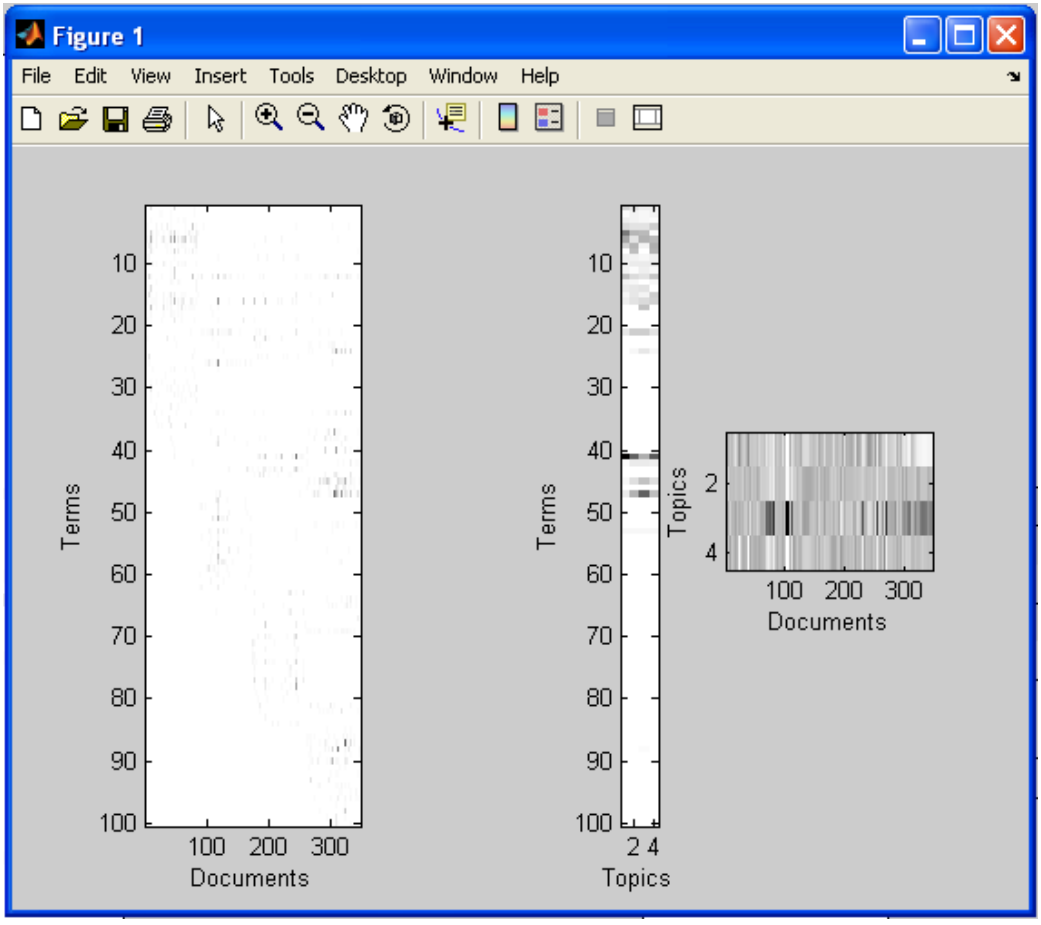
La Figure 11. Représente les résultats obtenus sur notre corpus, qui serons utilisés comme résultats de comparaison pour la solution proposée.

4.3.2 Simulation de l'algorithme pLSA_Gossip

Les figures suivantes représentent les résultats obtenus, en exécutant notre implémentation qui illustre le fonctionnement de l'algorithme après plusieurs itérations. Il montre comment la matrice initiale termes x documents est décomposée en produit de deux matrices, matrice termes x topics et une matrice topics x documents.

Le côté itératif de l'algorithme pourra peut-être paraître un peu mystérieux pour l'instant, mais comme nous le verrons, l'algorithme garantit que la vraisemblance augmente à chaque itération, ce qui conduit donc à des estimateurs de plus en plus corrects.

Nous allons évaluer la capacité de notre algorithme à retrouver les 4 classes réelles, ou, tout au moins, semblable. Nous initialisons l'algorithme à 4 classes à trouver au hasard. Les résultats présentés dans la figure. 12 sont obtenus en tant que moyenne de la première itération de l'exécution de notre algorithme.



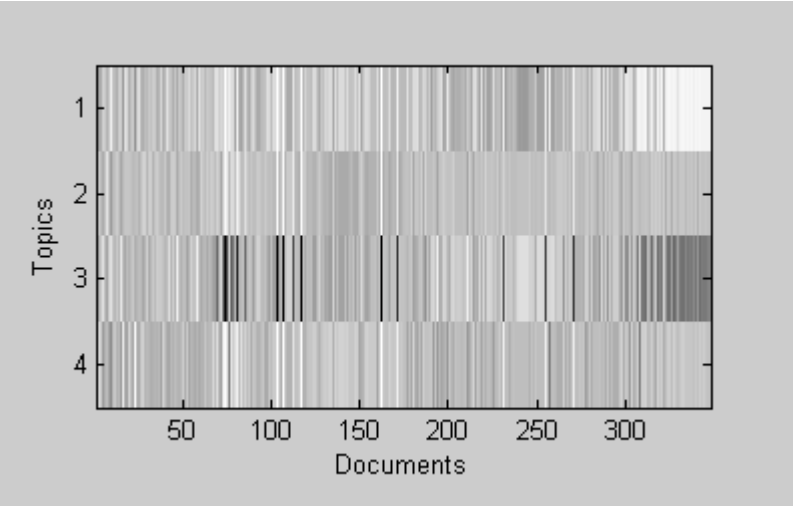
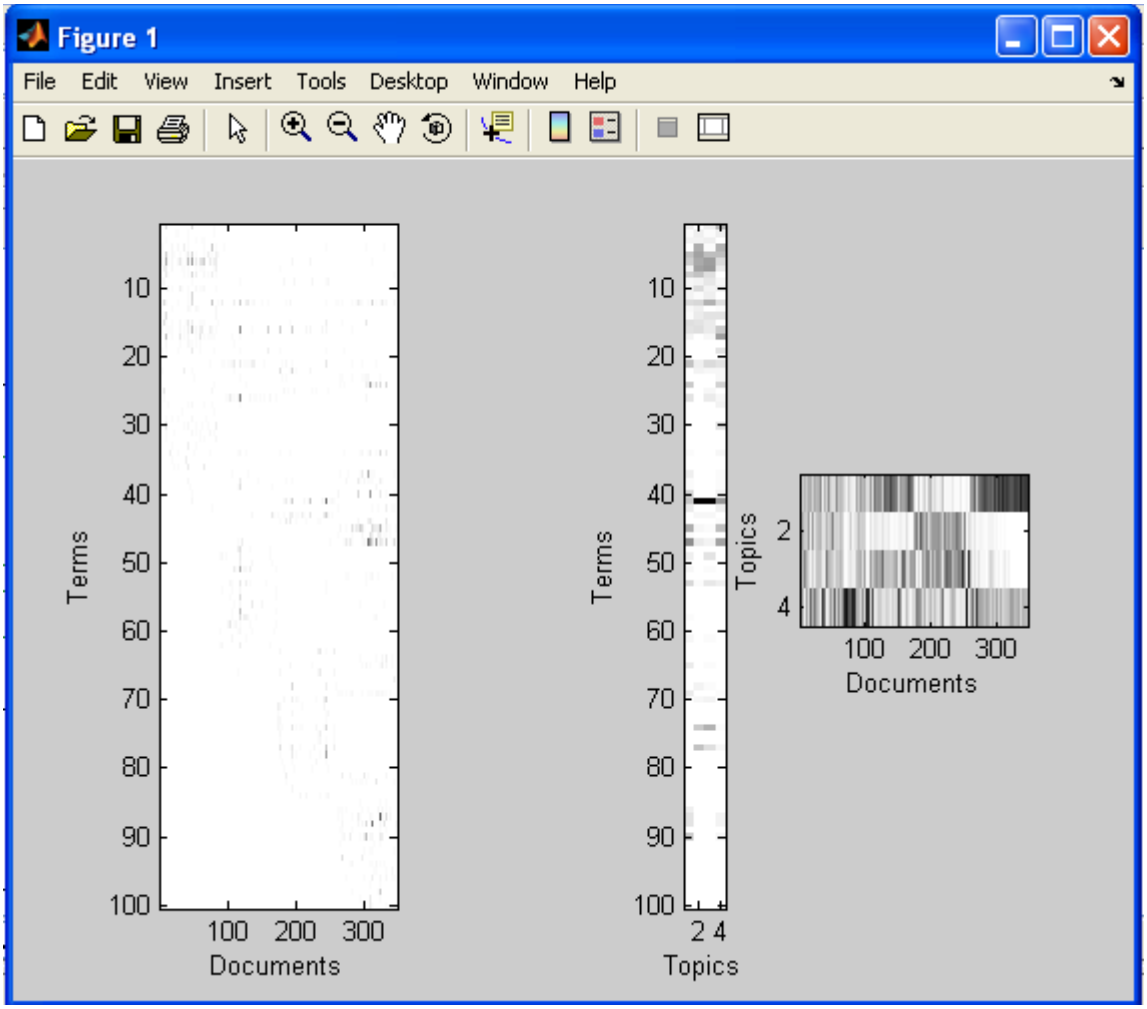


Figure 12. pLSA_Gossip après la 1ère itération.

- La figure.12 représente la classification obtenue après une itération, on peu voir que la classification ne correspond pas aux résultats voulus, donc le système de converge pas encor.



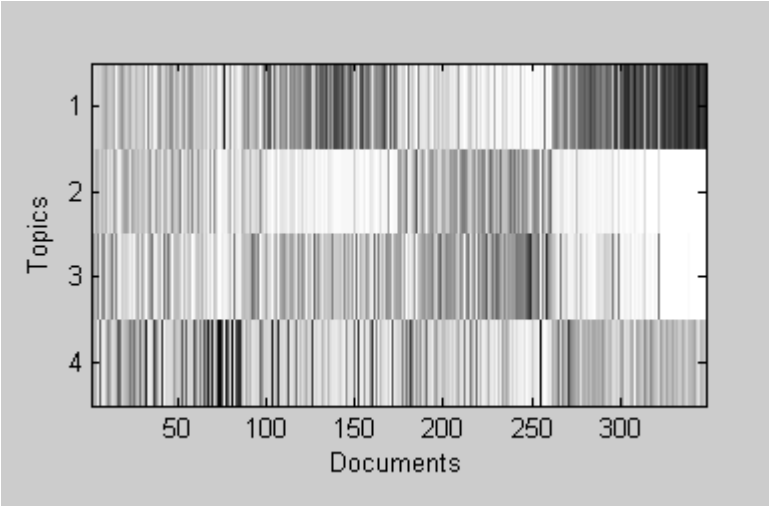
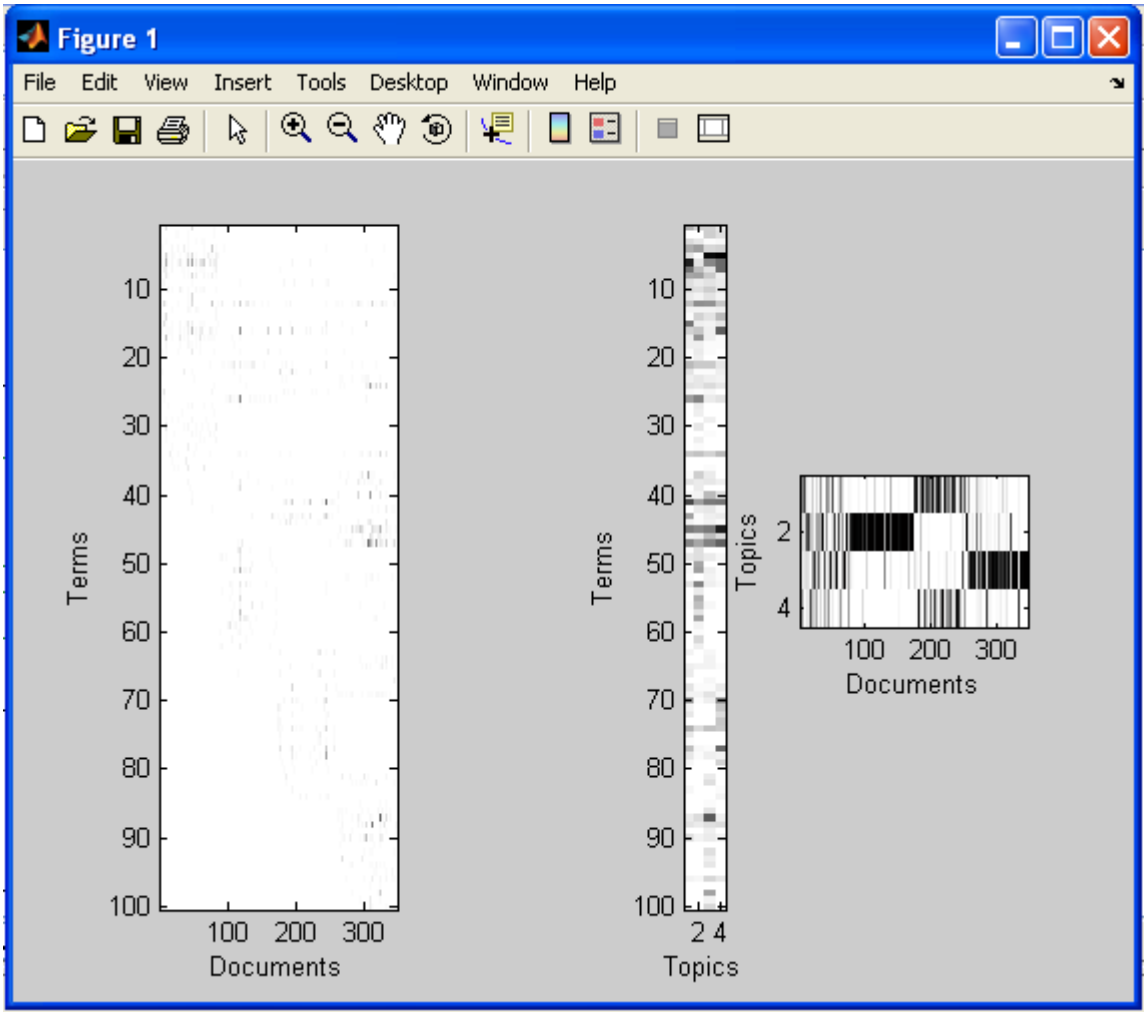


Figure 13. pLSA_Gossip après la 2eme itération.

- La figure.13 représente la classification obtenue après la deuxième itération, on peu voir que la classification ne correspond pas aux résultats voulus, mais on se rapproche de la vraisemblance.



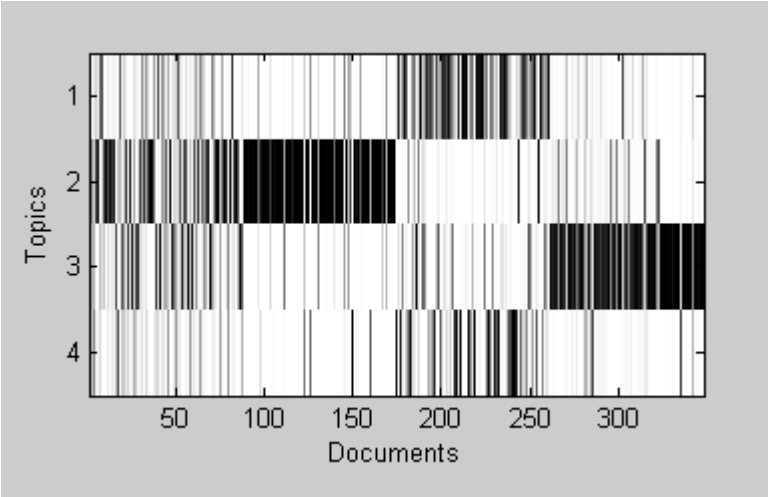
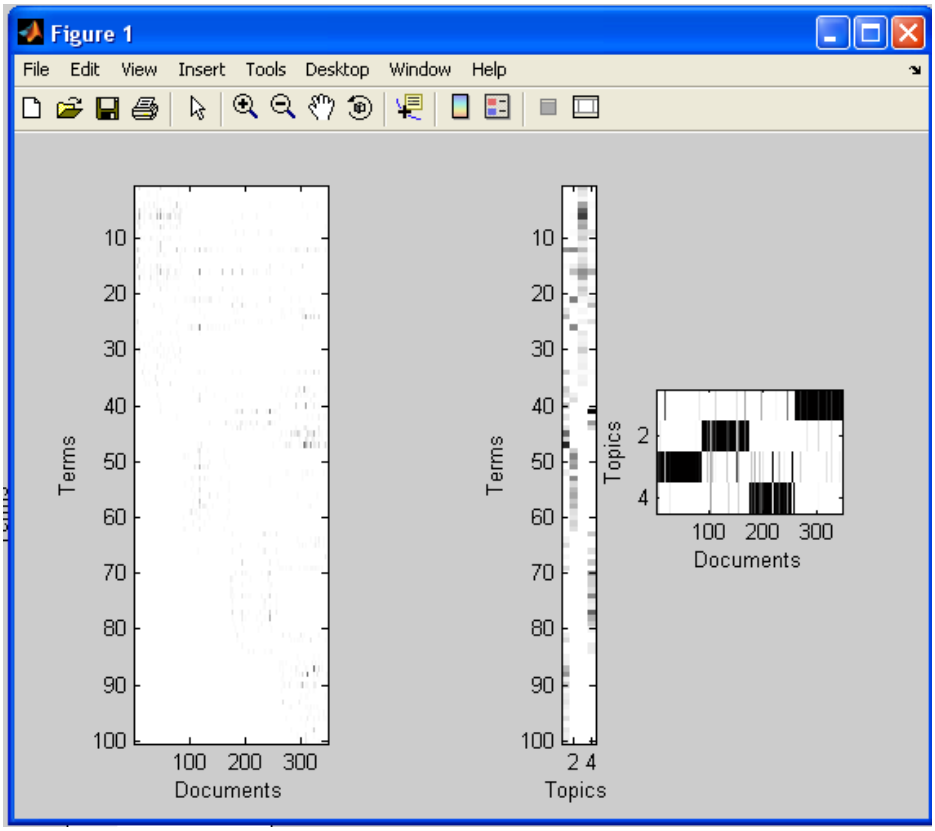


Figure 14. pLSA_Gossip après 5 itérations.

- La figure.14 représente la classification obtenue après cinq itérations, on peu voir que la classification se rapproche de plus en plus de la vraisemblance.



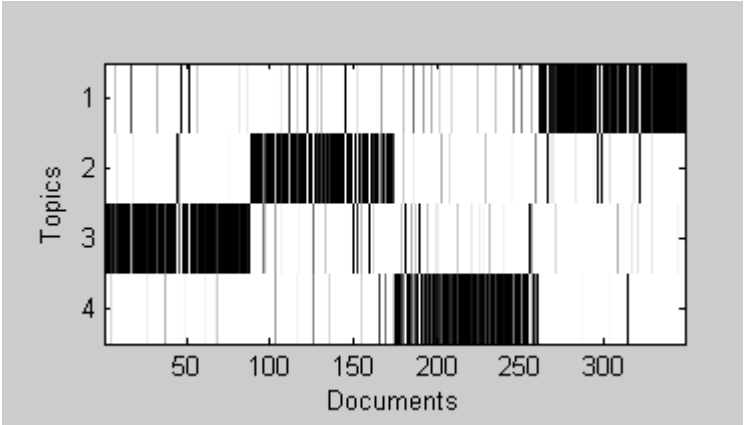


Figure 15. Résultat obtenu par l'algorithme pLSA_Gossip

- La figure.15 représente la classification obtenue après 100 itérations, on peu voir que la classification obtenue est plus au moins satisfaisante, comparativement a la classification obtenue avec pLSA standard (Figure.11).

4.4 Discussion

Dans nos expériences nous avons utilisé un sous-ensemble des collections de 20Newsgroups. Afin d'évaluer la pertinence des partitions obtenues, nous avons d'abord exécuté la méthode pLSA standard sur notre corpus, nous avons obtenu les résultats de la classification(figure.11), sachant au préalable que les documents de ce corpus appartiennent à 4 classes, les classes sont divisés comme suit : les documents de 1- 87, appartiennent à la première classe, de 88-174 à la deuxième classe, 175-261 à la troisième classe, 262-348 la quatrième classe, l'ordre des classes est aléatoire. Suite à ça, nous avons divisé notre corpus en plusieurs sous-corpus, en considérant chaque document, comme étant un corpus, donc notre unité de mesure, sur chaque nœud, est un sac de mots représentant un document. Nous avons exécuté notre algorithme pLSA_Gossip, sur chaque nœud et fait l'échange de données entre les nœuds. Chaque nœud garde en locale ses données ainsi que la distribution des probabilités des documents, et transmet ses multinomiales au nœud voisin, ce dernier agrège les multinomiales local avec ceux de son voisin, et ainsi de suite, jusqu'au traitement de tous les nœuds. Nous pouvons constater qu'après un certain nombre d'itération, nous obtenons un résultat plus ou moins proche de la réalité, rappelons que notre corpus de test, est réparti en 4 thèmes prédéfini. En effet, on démarre l'algorithme avec une ignorance absolue des variables latentes (données cachées) Z et en initialisant les multinomiales θ à une valeur θ_0 totalement arbitraire, potentiellement très loin de la réalité. L'algorithme se sert de θ_0 pour

estimer Z, puis se sert de Z pour ré-estimer les paramètres en une valeur θ_1 plus pertinente, et agrège les θ obtenus entre les nœuds.

À l'itération suivante, on améliore donc l'estimation des données cachées Z puisque cette nouvelle estimation se base cette fois sur θ_1. Et cette meilleure précision sur Z conduit à son tour à une meilleure précision sur θ_2, etc .

Sur les différentes figures, nous avons indiqué l'évolution du nombre de classes estimés au cours de l'exécution du processus.

De ces résultats, nous formons les conclusions suivantes :

Après 100 itérations le système converge lentement vers le maximum de vraisemblance. Dans le cadre expérimental utilisé (échantillon statique, initialisations aléatoires), le niveau de classes observé dans les vrais groupes a presque pu être atteint. Le processus d'estimation devrait donc être capable de fortement se rapprocher de vraies classes.

4.5 Conclusion

Nous avons présenté notre contribution dans le domaine de la classification distribuée et non-supervisée, l'agrégation des multinomiales du modèles pLSA sur différents nœuds, permet de trouvé une classification globale, de tout le réseau, au niveau de chaque nœud. Nous avons illustré notre proposition par des expériences. Nous aurions pu employer des échantillons de dimensionnalités plus élevées, mais nous avons justifié notre choix dans un but de comparaison. La prise en compte de la classe de chaque document au préalable, ouvre la voie à la validation de notre approche. Les résultats obtenus nous semblent bon, après comparaison entre la version centralisée et la version décentralisée qu'on a élaborées, sur un corpus de données réel de taille modeste, mais réaliste et « benchmark standard du domaine ».

Chapitre 5

Conclusion Générale

Le travail présenté concerne l'apprentissage statistique, plus précisément la classification statistique non supervisée. Il traite le cas des données vectorielles discrètes. Dans le cadre large de la recherche d'information, de tels vecteurs discrets sont, en particulier, largement utilisés pour décrire les fréquences d'occurrence de termes dans des documents textuels, représentés par des «sacs de mots» ou encore pour caractériser les usages et interactions homme/machine (matrices utilisateurs/objets), par exemple pour les systèmes à recommandation. Sur de telles données, le travail est orienté vers les techniques d'analyse sémantique latente, qui permettent, par exemple dans les deux exemples cités ci-dessus, d'identifier des thématiques cachées dans une masse de documents, ou d'identifier automatiquement des communauté d'intérêts dans un large ensemble d'utilisateurs ou réseaux sociaux.

L'originalité du travail réside en le caractère réparti des données et de l'algorithmique de classification non supervisée, et en le souhait que ces algorithmes de classification opèrent de façon décentralisée, pour traiter des sources de données multiples dynamiques.

Ce mémoire a abordé le problème de la classification non supervisée de documents distribués par l'étude d'un modèle particulier: le modèle pLSA, qui utilise le mélange de lois multinomiales avec variables latentes thématiques au niveau des documents.

Dans notre travail, nous avons présenté un panorama de la classification non supervisée de données textuelles, méthodes **vectorielles** (K-moyennes, analyse sémantique latente, factorisation en matrices non négatives, goulot d'information) et méthodes **probabilistes** (mélange de multinomiales, analyse sémantique latente probabiliste, allocation Dirichlet latente). Nous avons mis l'accent sur les raisons qui ont motivé notre préférence à utiliser les modèles probabilistes, à savoir les facilités de généralisation d'interprétation des résultats et

l'adaptabilité naturelle du cadre statistique à des situations variées, notamment aux cadres supervisés et semi-supervisés. Parmi ces derniers, nous avons justifié notre intérêt pour le modèle pLSA. Nous avons par la suite discuté du choix d'utilisation de l'algorithme Gossip, pour la partie distribuée, et pour les techniques de diffusion asynchrone de données décentralisées, par passage de message. Enfin, nous avons proposé un algorithme étendu de la méthode pLSA, que nous avons nommé pLSA_Gossip.

Pour remédier aux problèmes de centralisation de données, et afin de décentraliser les algorithmes de classification opérant de façon centralisée, et pour traiter des sources de données multiples dynamiques, nous avons proposé un procédé d'estimation répartie, s'appuyant sur une agrégation des modèles estimés séparément, sur des dépôts de données distincts. Une propriété importante de ce procédé d'agrégation est qu'il n'a recours qu'aux paramètres des modèles, plutôt qu'avoir accès aux données.

L'approche proposée est basée sur des méthodes probabilistes. Ayant une collection de corpus distribués sur plusieurs nœuds différents, le problème consiste à partitionner chacun de ces corpus en considérant les données locales et les classifications distantes des autres corpus distribués, sans partage de données entre les différents nœuds. Pour ce faire, nous avons proposé une approche asynchrone qui se subdivise en deux phases : Une phase locale et une phase distribuée. La phase locale reviendrait à appliquer un algorithme de classification, dans notre cas c'est la méthode pLSA, localement et indépendamment sur chacun des corpus, ce qui se soldera par l'obtention d'une partition pour chacun de ces corpus. La phase distribuée reviendrait à faire participer chacun des corpus avec toutes les classifications associées aux autres corpus lors de la phase locale. Ainsi, comme résultat on obtient sur chacun des nœuds une classification proche de la classification qu'on aurait obtenue si on avait fait centraliser tous les corpus en un seul corpus global. A l'issu des deux phases, toutes les classifications locales seront enrichies.

Comme perspectives

- Nous souhaitons vivement valider la méthode par une application réelle ou les corpus sont distribués.
- Appliquer la solution aux systèmes de recommandations pour identifier automatiquement des communautés d'intérêt dans un large ensemble d'utilisateurs ou réseaux sociaux.

- Les volumes de données et les applications à la recherche d'information de multimédia se développent très rapidement et est devenue une tâche importante dans les domaines d'application modernes (l'imagerie médicale, le commerce électronique, la recherche de copie illégale pour les raisons de droit d'auteur, etc) et les applications à usage général (par exemple Google Image et Youtube). Le terme de multimédia comprend les textes, les images, les documents audio et vidéo. Comme perspective, adapter et améliorer la solution proposée aux documents multimédia.
- Proposer une approche avec la méthode LDA, comme solution pour le cas distribué.

Bibliographies

[1]: D. Agrawal, A. El. Abbadi, and R.C. Steinke. Algorithms in replicated databases. Proceedings of the sixteenth ACM SIGACT-SIGMOD-SIGART Symposium on Principles ofDatabase Systems, pages 161{172, 1997.

[2]: F. R. Bach and M.I. Jordan. Leaming spectral clustering. In Advances in Neural Information Processing Systems (NIPS), volume 16, 2004.

[3]: J.P. Benzécri et al. *Pratique de l'analyse des données, tome* 3. *Linguistique et lexicologie.* Dunod, 1981.

[4] : M. Bishop. Neural Networks for Pattern Recognition. Oxford University Press, 1995.

[5]: D. M. Blei, A.Y. Ng, and M.I. Jordan. Latent Dirichlet allocation. In *Advances in Neural Information Processing Systems (NIPS),* volume 14, pages 601-608, 2002.

[6]: B. E. Boser, I. M. Guyon, and V. N. Vapnik. A training algorithm for optimal margin classifiers. In *Proceedings of the 5th Annual Workshop on Cornputational Leaming Theory (COLT),* pages 144-152, 1992.

[7] :L. Breiman, J.H. Friedman, R.A. Olshen, and P.J. Stone. Classi¯cation and Regression Trees. Wadsworth, 1984.

[8]: W. Buntine and A. Jakulin. Applying discrete PCA in data analysis. In *Proceedings of the 20th Conference on Uncertainty in Artificial Intelligence (UAI),* pages 59-66, 2004.

[9] : C. Cortes and Vl. Vapnik. Support-vector networks. Machine Learning, 20(3), Septembre 1995.

[10] : J.COSTA DA SILVA & M.KLUSCH (2006). Inference in distributed data clustering, *Eng. Appl. Artificial Intelligence* 19, p. 363-369

[11]: N. Cristianini, J. Shawe-Taylor, and H. Lodhi. Latent semantic kemels. In *Proceodings of the 18th International Conference on Machine Learning (ICML).* Morgan Kaufman, 2001.

[12]: D. R. Cutting, D. R. Karger, J. O. Pedersen, and J. W. Tukey. Scatter/gather: A clusterbased approach to browsing large document collections. In *Pmceedings of the 15th ACM international ConfeTence on Researd, and Development of Infomwtion RetTieval (SICIR),* pages 318-329, 1992.

[13]: S. C. Deerwester, S. T. Dumais, T. K. Landauer, G. W. Fumas, and R. A. Harshman. Indexing by latent semantic analysis. *Journal of the American Society of Information Science,* 41(6) :391-407, 1990.

[14]: A. Demers, D. Greene, C. Hauser, W. Irish, J. Larson, S. Shenker, H. Sturgis, D. Swinehart, and D. Terry. Epidemic algorithms for replicated database maintenance. Proceedings of the 6th ACM Symposium on Principles of Distributed Computing, pages 1{12, 1987.

[15]: A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society,* B 39 :1-38, 1977.

[16]: C. Ding, X. He, and H. D. Simon. On the equivalence of nonnegative matrix factorization and

spectral clustering. In *Proceedings of SIAM International Conference on Data Mining (SDM),* pages 606-610, 2005.

[17]: C. Ding, T. Li, and W. Peng. Nonnegative Matrix Factorization and Probabilistic Latent Semantic Indexing : Equivalence, chi-square statistic, and a hybrid method. In *Proceedings of the 21st National Conference on Artificial Intelligence (AAAI'06), 2006.*

[18]: D. Donoho and V. Stodden. When does non-negative matrix factorization give a correct decomposition into parts? In *Advances in Neural Information Processing Systems (NIPS),* volume 16, 2004.

[19]: S. T. Dumais. Enhancing peIforrnance in latent semantic indexing (LSI) retrieval. Technical Report TM-ARH-017527, Bellcore, 1990.

[20] : R. Fletcher. Practical methods of optimization. Whiley, second edition, 1987.

[21]: E. Gaussier and C. Goutte. Relation between PLSA and NMF and implications. In *Proceedings of the 28th A CM international Conference on Research, and Development of Information Retrieval (SICIR),* pages 601-602, 2005.

[22]: I. Gupta, R. van Renesse, and K. Birman. Scalable fault-tolerant aggregation in large process groups. Proc. Conf. on Dependable Systems and Networks, pages 433{442, 2001.

[23]: R. Guy, G. Popek, and T.w.jr. Page. Consistency algorithms for optimistic replication. Proceedings of the First International Conference on Network Protocols, IEEE, 1993.

[24]: E.-H. S. Han, G. Karypis, V. Kumar, and B. Mobasher. Hypergraph based clustering in high-dimensional data sets: A summary of results. *IEEE Bulletin of the Technical Committee on Data Engineering,* 21 (1), 1998.

[25]: T. Hofmann. Unsupervised learning by probabilistic latent semantic analysis. *Machine Leaming Joumal,* 42(1) :177-196,2001.

[26]: P. O. Hoyer. Non-negative matrix factorization with sparseness constraints. *Joumal of Machine Leaming Researc,* 5 :1457-1469, 2004.

[27] : E. J. Hunter, (2000). Do we still need classification? In Marcella, R., & Maltby, A. (eds.). The future of classification. London, UK : Gower, 1-17.

[28] : E. J. Hunter, (2002). Classification made simple. Aldershot, Hants, England ; Burlington, Vermont : Ashgate

[29]: A. Hyvarinen, J. Karhunen, and E. Oja. *Independent Component Analysis.* Wiley-Interscience, 2001.

[30] :Michael I. Jordan and Robert A. Jacobs. Hierarchical mixtures of experts and the em algorithm. Neural Computation, 6(2):181{214, March 1994.

[31]: A. K. Jain, M. N. Murphy, and P. Flynn. Data clustering : a review. *ACM Computing surveys,* 31(3) :264-323, 1999.

[32]: M. Jelasity, W. Kowalczyk, and M. van Steen. Newscast computing. Technical report, Dept. of Computer Science, Vrije Universiteit Amsterdam,IR-CS-006, 2003.

[33]: T. Joachims. Text categorization with suport vector machines : Leaming with many relevant features. In *Proceedings of the 10th European Conference on Machine Leaming (ECML),* pages 137-142, 1998.

[34]: D. Jurafsky and J. H. Martin. *Speech and Language Processing* : *An Introduction to Natuml Language Processing, Computational Linguistics, and Speech Recognition.* Prentice Hall PTR,2000.

[35]: R. Karman, S. Vempala, and A. Vetta. On clusterings : Good, bad and spectral. In *Proceedings of the 41st Annual Syrnposiurn on the Foundation of Computer Science (FOCS),* pages 367-377, 2000.

[36]: D.Kempe, Alin Dobra, and Johannes Gehrke. Gossip-based computation of aggregate information. Proceedings of the 44th Annual IEEE Symposium on Foundations ofComputer Science, pages 482{491, 2003.

[37]: T. Kohonen, editor. *Self-orqanizing maps.* Springer-Verlag, 1997.

[38]: T. Kolenda and L. K. Hansen. Independent components in text, 1999.

[39]: A. Kononov M. Rabinovich, N. Gehani. Scalable update propagation in epidemic replicated databases. Proceedings of the 5th International Conference on Extending Database Technology: Advances in Database Technology, 1057:207{222, 1996.

[40]: J. B. Kruskal. Nonmetric multidimensional scaling : a numerical method. *Psychornetrika,* 29(2) :115-129, 1964.

[41]: K. Lagus, T. Honkela, S. Kaski, and T. Kohonen. WEBSOM for textual data mining. *Artificial Intelligence Review,* 13(5/6) :345-364, 1999.

[42]: T. K. Landauer and S. T. Dumais. A solution to Plato's problem : The latent semantic analysis : Theory of the acquisition, induction, and representation of knowledge. *Psychological Review,* 104 :211-240, 1997.

[43]: T. K. Landauer, P. W. Foltz, and D. Laham. Introduction to latent semantic analysis. *Discourse Processes,* 25 :259-284, 1998.

[44]: L. Lebart and A. Salem. *Analyse Statistique des Données Textuelles.* Dunod, 1988.

[45]: D. D. Lee and H. S. Seung. Algorithms for non-negative matrix factorization. In *Advances in Neural Inforrnation Processing Systems (NIPS),* volume 13, pages 556-562, 2001.

[46]: J. B. MacQueen. Sorne rnethods for classification and analysis of rnultivariate observations. In *Proceedings of the Fifth Berkeley Syrnposiurn on Mathernatics, Statistics and Probability,* volume 1, pages 281-296, 1967.

[47]: S. Madden, M. Franklin, J. Hellerstein, and W. Hong. Tag: A tiny aggregation service for ad-hoc sensor networks. Proc. 5th Symp. on Operating Systems Design and Implementation, 2002.

[48]: A. McCallum and K. Nigam. A comparison of event models for Naive Bayes text classification. In *AAAI-g8 Workshop on Leaming for Text Categorization,* pages 41-48, 1998.

[49]: A. Y. Ng, M. 1. Jordan, and Y. Weiss. On spectral clustering : Analysis and an algorithm. In *Advances in Neural Information Processing Systerns (NIPS),* volume 14, pages 849-856, 2002.

[50]: K. Nigam, A. K. McCallum, S. Thrun, and T. M. Mitchell. Text classification from labeled and unlabeled documents using EM. *Machine Learning,* 39(2/3) :103-134,2000.

[51]: J. Nocedal and S.J. Wright. Numerical optimization. Springer, 1999.

[52] :Richard O. Duda and Peter E. Hart. Pattern classi¯cation and scene analysis. Wiley-Interscience, 1973.

[53]: C. H. Papadimitriou, H. Tamaki, P. Raghavan, and S. Vempala. Latent semantic indexing : A probabilistic analysis. In *Proceedings of the ACM Conference on Principles of Database Systerns (PODS),* pages 159-168, Seattle, 1998.

[54] : W. PEDRYCZ (2002) Collaborative fuzzy clustering, *Pattern Recognition Lett.* 23, p. 675-686.

[55] : W. PEDRYCZ & P.RAI (2008). Collaborative clustering with the use of Fuzzy C-Means and its quantification, *Fuzzy Sets and Systems,* doi: 10.1016/j.fss.2007.12.030.

[56]: J. Quesada, W. Kintsch, and E. Gomez. A computational theory of complex problem solving using latent semantic analysis. In *Proceedings of the 24th Annual Conference of the Cognitive Science Society (GOGSG1'02),* pages 750-755, 2002.

[57]: G. Salton, A. Wong, and C. S. Yang. A vector space model for automatic indexing. *Commun. A CM,* 18(11) :613-620, 1975.

[58]: F. Santos. L'algorithme EM : une courte presentation, CNRS, UMR 5199 PACEA, 28 juin 2010

[59]: L. Saul and S. Roweis. An introduction to locally linear embedding, 2001. URL http: //www.cs.toronto.edu/-roweis/lle/papers/lleintro.pdf.

[60]: J. Shawe-Taylor and N. Cristianini. *Kernel Methods for Pattern Analysis.* Cambridge University Press, 2004.

[61]: J. Shi and J. Malik. Normalized cuts and image segmentation. In *Proceodings of the IEEE Conference on Computer Vision and Pattern Recognition (GVPR),* pages 731-737,1997.

[62]: N. Slonim, N. Friedman, and N. Tishby. Unsupervised document classification using sequential information maximization. In *Pmcecdings of the 25th AGM intemational GonfeTence on Researû, and Development of InfoTmation RetTieval (SIGIR),* pages 129-136, 2002.

[63]: N. Slonim and N. Tishby. Document clustering using word clusters via the information bottleneck method. In *Research, and Development in Information Retrieval,* pages 208215,2000.

[64]: M. Steinbach, G. Karypis, and V. Kumar. A comparison of document clustering techniques. In *Proceedings of the Knowledge Discovery and Data Mining Workshop on Text Mining,* 2000.

[65] : A. STREHL & J. GHOSH (2002). Cluster ensembles: a knowledge reuse framework for combining multiple partitions. *Journal of Machine Learning* Research (JMLR), 3, p. 583 – 617.

[66]: F. J. Theis, K. Stadlthanner, and T. Tanaka. First results on uniqueness of sparse nonnegative

matrix factorization. In *Proceedings of European Signal Processing Conference (EUSIPCO)* , 2005.

[67]: N. Tishby, F. C. Pereira, and W. Bialek. The information bottleneck method. In *Proceedings of the 37th Annual Alierton Conference on Communication, Control and Computing,* pages 368-377, 1999.

[68] :V. N. Vapnik. Statistical learning theory. Wiley, New York, ISBN: 978-0-471-03003-4, pages 339{371, 1998.

[69]: R. van Renesse, K. Birman, and W. Vogels. Astrolabe: A robust and scalable technology for distributed system monitoring, management, and data mining. ACM TOCS, 2003.

[70]: A. Vinokourov. Why nonnegative matrix factorization works weil for text information retrieval. URL http://citeseer . ist. psu. edu/458322.html. Preprint, 2002.

[71]: E. M. Voorhees. The cluster hypothesis revisited. In *Proceedings of the 8th ACM international Conference on Research, and Development of Information Retrieval (SICIR),* pages 188-196, 1985.

[72]: y. Weiss. Segmentation using eigenvectors : A unifying view. In *Proceedings of the International Conference on Computer Vision (ICCV),* pages 975-982, 1999.

[73]: W. Xu, X. Liu, and Y. Gong. Document clustering based on non-negative matrix factorization. In *Proceedings of the 26th A CM international Conference on Research, and Development of Information Retrieval (SICIR),* pages 267-273, 2003.

[74]: N. Slonim and N. Tishby. The power of word clusters for text classification. In *Proceedings of the 23Td European Colloquium on Information Retrieval Research, (ECIR), 2001.*

الملخص

نناقش في هذه الأطروحة مشكلة تصنيف غير خاضعة للرقابة من الوثائق التي وزعتها وسائل احتمالي .دراستنا تركز بشكل خاص على القوانين النموذجية خليط متعدد الحدود المادية الكامنة متغير مستوى الموضوعية.

ويركز العمل على التعلم دون إشراف توزيعها على الحفاظ على سرية البيانات باستخدام نتائج تصنيف أخرى دون استخدام البيانات الأخيرة .وسوف يستند النهج الذي ينبغي دراستها على الطرق الاحتمالية .وجود مجموعة من المجاميع موزعة على مواقع مختلفة، والمشكلة هي لتقسيم كل جسم النظر في البيانات المحلية والتصنيفات للجسم بعد الأخرى ، وتوزيعها، مع عدم وجود تبادل البيانات بين المراكز المختلفة .للقيام بذلك، نقترح نهجا غير المتزامن الذي ينقسم إلى مرحلتين : مرحلة محلية ومرحلة توزيعية . سوف تكون مرحلة المحلية لتطبيق خوارزمية تصنيف في حالتنا هو الأسلوبPLSA ، محليا وبشكل مستقل على كل الجسم، الأمر الذي سيؤدي في الحصول على نقاط عن كل جسم .وتشمل المرحلة توزع كل واحد من الجسم مع جميع التصنيفات المرتبطة هيئة أخرى في المرحلة المحلية .وهكذا ، نتيجة يتم الحصول عليها في كل موقع بالقرب من تصنيف لتصنيف التي ستنجم لو كان لدينا كل جسم مركزي في هيئة عالمية واحدة، وتجاهل قيد السرية .في نهاية المرحلتين، سوف يتم تصنيف جميع التخصيب المحلية.

الكلمات المفتاحية : التعلم بدون إشراف، والتعلم الموزعة، والحوسبة الموزعة والبيانات الموزعة، وأساليب احتمالي.

RÉSUMÉ

Nous abordons dans ce mémoire le problème de la classification non supervisée de documents répartis par des méthodes probabilistes. Notre étude se concentre en particulier sur le modèle de mélange de lois multinomiales avec variables latentes thématiques au niveau des documents.

Le travail porte sur l'apprentissage non supervisé distribué permettant de préserver la confidentialité des données en utilisant d'autres résultats de classifications sans avoir recours aux données de ces dernières. L'approche qui sera étudiée sera basée sur des méthodes probabilistes. Ayant une collection de corpus distribués sur plusieurs sites différents, le problème consiste à partitionner chacun de ces corpus en considérant les données locales et les classifications distantes des autres corpus distribués, sans partage de données entre les différents centres. Pour ce faire, nous proposons une approche asynchrone qui se subdivise en deux phases : Une phase locale et une phase de distribuée. La phase locale reviendrait à appliquer un algorithme de classification, dans notre cas c'est la méthode pLSA, localement et indépendamment sur chacun des corpus, ce qui se soldera par l'obtention d'une partition pour chacun de ces corpus. La phase distribuée reviendrait à faire participer chacun des corpus avec toutes les classifications associées aux autres corpus lors de la phase locale. Ainsi, comme résultat on obtient sur chacun des sites une classification proche de la classification qu'on aurait obtenue si on avait fait centraliser tous les corpus en un seul corpus global. A l'issu des deux phases, toutes les classifications locales seront enrichies.

MOTS-CLÉS: Apprentissage non-supervisé, apprentissage distribué, calcul réparti, données distribuées, méthodes probabilistes.

ABSTRACT

We discuss in this paper the problem of unsupervised classification of documents distributed by probabilistic methods. Our study focuses particularly on the mixture model laws multinomial latent variable level thematic material.

The work focuses on unsupervised learning distributed to preserve the confidentiality of data using other classification results without using recent data. The approach to be studied will be based on probabilistic methods. Having a collection of corpora distributed on several different sites, the problem is to partition each corpus considering local data and classifications of other remote distributed corpus, with no sharing of data among different centers. To do this, we propose an asynchronous approach which is divided into two phases: a local phase and phase distribution. The local phase would be to apply a classification algorithm in our case is the PLSA method, locally and independently on each corpus, which will result in obtaining a score for each corpus. Distributed phase would involve each of the corpus with all classifications associated with the other body at the local phase. Thus, as a result is obtained at each site near a classification of the classification that would result if we had centralized all the corpus into a single global body, and ignoring the constraint of confidentiality. At the end of both phases, all the local classification will be enriched.

KEYWORDS: Unsupervised Learning, Distributed Learning, distributed computing, Distributed Data, probabilistic methods.