Introduction aux modèles probabilistes utilisés en Fouille de Données

Théo FRANCESIAZ
Raphaël GRAILLE
Brahim METAHRI

11 Juin 2015

Table des matières

1 *Latent Dirichlet Allocation*
1.1 Vectorisation de documents
1.2 Préliminaires théoriques au modèle
1.2.1 Echangeabilité et Théorème de De Finetti
1.2.2 Loi de Dirichlet
1.3 Présentation du *LDA*
1.3.1 Principe
1.3.2 Intérêt des variables latentes
1.3.3 Comparaison à d'autres modèles

2 Estimation des paramètres du *LDA*
2.1 EM variationnel
2.1.1 Principe de l'algorithme
2.1.2 Evaluation des paramètres du *LDA*
2.2 Echantillonnage de Gibbs
2.2.1 Théorie
2.2.2 Implémentation

3 Résultats
3.1 Classification non supervisée
3.1.1 Classification par estimation des paramètres
3.1.2 Classification sur l'espace des topics
3.2 Analyse des résultats
3.2.1 Utilisation de la F-Mesure
3.2.2 Résultats par estimation des paramètres
3.2.3 Comparaison des résultats entre espace des topics et espace des mots
3.2.4 Réduction du nombre de classes expertes

Table des figures

1.1 Loi de Dirichlet - Découpe de ficelles . . . .
1.2 Latent Dirichlet Allocation - Lien entre paramètres . . . .

2.1 Schéma illustratif de la *mean-field assumption* . . . .
2.2 Visualisation des estimateurs γ et ϕ . . . .
2.3 EM variationnel appliqué à un corpus de documents . . . .
2.4 Implémentation personnelle optimisée de l'E-step . . . .
2.5 Algorithme *Gibbs sampling* appliqué à un corpus de documents . . . .

3.1 F-mesure pour le clustering du corpus 20NewsGroups par l'EM variationnel, 24 itérations . . . .
3.2 Top words pour les classes 6, 7, 8 et 14 . . . .
3.3 Clustering du corpus 20NewsGroups par l'algorithme CEM dans différents espaces
3.4 Evolution de la F-mesure en fonction du nombre de classes expertes . . . .
3.5 Clustering expert initial du corpus 20NewsGroups . . . .

Introduction

Le sujet abordé au cours de ce projet s'inscrit dans le cadre d'une thématique plus large concernant la manipulation, l'exploitation et l'interprétation de très grandes quantités de données. Cette étude se concentre essentiellement sur un modèle probabiliste en particulier, développé dans les années 1990 : le modèle LDA (*Latent Dirichlet Allocation*). Il est très utilisé pour des problèmatiques de modélisation de données discrètes telles que des corpus de documents, l'objectif étant d'en obtenir une représentation aussi concise que possible en terme de quantité d'information.

La base de travail initiale du projet consiste en une base de 19997 documents textuels issus de la base de données 20NewsGroups, de taille et de nature diverses (e-mail, article, dépêche, ...). A ce corpus est associé un vocabulaire de 59809 mots qui regroupe les "termes-clés" de celui-ci. Ne sont pas considérés comme termes-clés les entités linguistiques suivantes : prépositions, articles définis/indéfinis, pronoms, conjonctions. Autrement dit, seuls les mots réellement "significatifs" figurent dans le vocabulaire. Enfin, on suppose connues un nombre fixé de *classes* (ou *topics*) permettant de regrouper des groupes de documents sous un même thème. Cette décomposition du corpus en sous-catégories s'appelle le *clustering*.

L'objectif global du projet consiste en l'application du modèle LDA au corpus sus-cité pour en dégager une classification sous forme de classes, avec une hypothèse d'importance majeure : **les documents sont supposés indépendants des autres**.

Ce document se compose de trois chapitres principaux.

Le premier est une introduction au modèle LDA. On y introduit en premier lieu des résultats théoriques de Probabilités à la base du modèle, ainsi que les techniques traditionnelles utilisées en Fouille de Données pour représenter synthétiquement de l'information. Le principe du modèle en lui-même est ensuite présentée de manière détaillée.

Le second présente les algorithmes *EM variationnel* et d'échantillonage de Gibbs (*Gibbs sampling*) choisis pour l'implémentation du modèle.

Enfin le dernier chapitre montre les résultats obtenus après exécution des algorithmes implémentant le *LDA*. On y interprétera notamment leur efficacité relative en confrontant leurs performances respectives.

Chapitre 1

Latent Dirichlet Allocation

1.1 Vectorisation de documents

En Fouille de Données, les documents sont généralement représentés sous forme de vecteurs d'entiers, où les composantes réfèrent à un mot indexé dans le vocabulaire associé au corpus. Ici, on utilise la représentation par *(indice :valeur)* : les composantes sont des couples d'entiers séparés par deux points, l'entier de gauche symbolisant l'indice du mot représenté, l'indice de droite son nombre d'occurrences dans le document considéré.

Exemple :

$$\vec{v} = (1:5, 2:0, 3:1)$$

Ici, le document est composé de 6 mots au total, mais seulement de 2 mots distincts :
- le mot 1 apparaît 5 fois
- le mot 2 n'apparaît pas
- le mot 3 apparaît 1 fois

Par souci de simplification, on ne fera pas apparaître les mots du vocabulaire n'apparaissant pas dans un document. Ainsi, $\vec{v}$ devient $\vec{v} = (1:5, 3:1)$.
Lorsque la classe des documents est connue, on la fait figurer en première composante du vecteur correspondant. Un corpus dont on connaît la répartition en classes se présente ainsi sous la forme suivante :

```
1 1:3 2:1 4:1 5:1 7:9 10:1
1 1:3 4:1 5:2 6:1 7:6
4 3:5 6:3 8:1 9:1 11:1 12:1 13:1 14:2 15:1
```

Chaque ligne correspondant à un document, on a ici :
- le 1er document appartient à la classe 1
- le second appartient à la classe 1 également
- le 3^{me} et dernier appartient à la classe 4

N.B : Le corpus dont on dispose pour l'application du modèle LDA se présente sous cette forme. Un corpus peut donc être vu comme un vecteur de documents.

1.2 Préliminaires théoriques au modèle

1.2.1 Echangeabilité et Théorème de De Finetti

Echangeabilité

Définition 1 *Un ensemble fini $\{X_1, ..., X_n\}$ est dit échangeable si sa loi jointe est invariante par permutation, i.e. :*

$$\forall \sigma \in S_n, \mathcal{L}(X_{\sigma(1)}, ..., X_{\sigma(n)}) = \mathcal{L}(X_1, ..., X_n)$$

où S_n est le groupe des permutations de $\{1, ..., n\}$.
Un ensemble infni de variables aléatoires est dit infiniment échangeable si tous ses sous-ensembles finis sont échangeables.

Lien avec le LDA : pour un document de N mots $D = (w_1, ..., w_N)$ donné, l'ensemble $w_1, ..., w_N$ est échangeable. Autrement dit, l'ordre des mots au sein des documents n'a pas d'importance : un document est un "sac de mots" plutôt qu'une liste ordonnée (*hypothèse bag-of-words*).

Théorème de De Finetti

Théorème 1 *Soit $\{X_i | i \in \mathbb{N}^*\}$ un ensemble infiniment échangeable.*
Alors il existe une variable aléatoire Y telle que, sachant Y, les X_i sont indépendantes et identiquement distribuées (i.i.d.), c'est-à-dire :

$$\mathcal{L}(X_1, ..., X_n | Y) = \mathcal{L}(X_1|Y) \otimes ... \otimes \mathcal{L}(X_n|Y)$$

pour tout $n \in \mathbb{N}^*$, avec $\mathcal{L}(X_i|Y) = \mathcal{L}(X_j|Y)$ pour tout (i,j) $\in \mathbb{N}^* \times \mathbb{N}^*$.

Lien avec le LDA : sachant que les mots $w_1, ..., w_N$ d'un document D de N mots sont échangeables, De Finetti assure que les w_i sont *i.i.d* conditionnellement à une variable aléatoire z que l'on introduira en section 1.3.

1.2.2 Loi de Dirichlet

Résultats essentiels

La loi de dirichlet, notée généralement $Dir(\alpha)$, est une loi de probabilité continue pour des variables aléatoires multinomiales. Elle est paramétrée par le vecteur α de réels positifs.

Pour un échantillon $X = (X_1, ..., X_K) \sim Dir(\alpha)$, on a :

- Densité de probabilités : pour une loi de Dirichlet d'ordre $K \geq 2$, de paramètre $\alpha = (\alpha_1, ..., \alpha_K)$, on a

  $$f(x_1, ..., x_K, \alpha_1, ..., \alpha_K) = \frac{1}{B(\alpha)} \prod_{i=1}^{K} x_i^{\alpha_i - 1}$$

  où le facteur de normalisation $B(\alpha)$ est la fonction *beta* s'exprimant en fonction de la fonction Γ :

  $$B(\alpha) = \frac{\prod_{i=1}^{K} \Gamma \alpha_i}{\Gamma(\sum_{i=1}^{K} \alpha_i)}$$

- Espérance :

  $$\mathbb{E}[X_i] = \frac{\alpha_i}{\alpha_0} \ , \qquad \text{où } \alpha_0 = \sum_{i=1}^{K} \alpha_i$$

- Variance :

  $$Var(X_i) = \frac{\alpha_i(\alpha_0 - \alpha_i)}{\alpha_0^2(\alpha_0 + 1)}$$

Interprétation intuitive

On donne l'intuition sous-jacente à la loi de Dirichlet à travers l'exemple suivant.

Exemple : *Découpe de ficelles*

On consière un nombre M de ficelles, de longueur initiale 1.0, que l'on souhaite découper en K pièces de différentes longueurs. Chaque pièce a une longueur moyenne désignée, mais on autorise néanmoins cette longueur à varier. Les longueurs $\frac{\alpha}{\alpha_0}$ spécifient les longueurs moyennes des découpes résultant de la distribution.

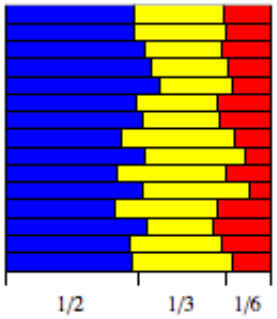


FIG. 1.1: Loi de Dirichlet - Découpe de ficelles

Cet exemple se transpose ***en tout point*** au problème de classification de corpus : les documents se substituent aux ficelles, et les *topics* aux découpes de ces-derniers. Ainsi, l'idée sous-jacente au paramètre α est qu'il donne **les proportions moyennes de chaque *topic* pour chacun des documents**.

1.3 Présentation du *LDA*

1.3.1 Principe

Le *Latent Dirichlet Allocation* est un modèle probabiliste *génératif* de corpus.
L'idée consiste à considérer les documents comme des mélanges aléatoires sur des *topics* sous-jacents, où chaque *topic* est caractérisé par une une distribution sur les mots. Le processus de génération des documents se décline ainsi :

Pour un corpus $C = (d_1, ..., d_M)$ de M documents :
Pour chaque document $d_i = (w_1, ..., w_{N_i})$ de N_i mots,

1. On choisit $N_i \sim Poisson(\xi)$
2. On choisit $\theta \sim Dir(\alpha)$
3. Pour chacun des N mots w_n de d_i :
   (a) On choisit un $z_n \sim Multinomiale(\theta)$
   (b) On choisit un mot w_n de $p(w_n|z_n, \beta)$, une probabilité multinômiale conditionnée sur le *topic* z_n

Le schéma génératif précédent fait ainsi ressortir cinq paramètres, définis comme suit :

- $\boldsymbol{\xi}$:
  Variable aléatoire générant le nombre de mots pour chaque document.
  En pratique, l'hypothèse d'une loi de Poisson pour les N_i n'est pas fondamentale pour la suite du modèle, et on peut donc s'en abstraire (d'autant plus que N_i n'est pas lié aux autres variables du modèle, quelque soit i).
  On supposera donc dans toute la suite de ce document que, pour tout $i \in \{1, ..., M\}$, N_i est une variable déterministe, sans aléa
- $\boldsymbol{\alpha}$:
  Vecteur de dimension K correspondant au paramètre de la loi de Dirichlet générant les mots d'un document. L'interprétation intuitive de ce paramètre est donnée dans l'exemple des ficelles de la section précédente.
  N.B : Aucune inférence n'est réalisée sur ce paramètre dans le cadre du projet car il n'impacte pas de façon significative la suite de l'étude ; α ***a été choisi de dimension égale au nombre de classes (K = 20), avec toutes ses composantes fixées à 1***
- $\boldsymbol{\beta}$:
  Matrice $K \times V$, où K représente le nombre de classes, et V la taille du vocabulaire associé au corpus. L'expression du terme général de la matrice est :

$$\beta_{ij} = p(w_j|z_i) \quad (*)$$

  Les coefficients β_{ij} correspondent ainsi à **la probabilité d'un mot w_j d'appartenir au *topic* z_i**. Contrairement à α, l'estimation de β a une importance critique dans l'application du *LDA*. On effectue donc une inférence sur ce paramètre à l'aide de l'algorithme EM (cf. chapitre suivant)
- $\boldsymbol{\theta}$ et $\boldsymbol{z}$:
  Ce sont des paramètres dits *latents* du modèle, d'où l'appellation *LDA*.
  Cette dénomination vient du fait qu'ils sont eux-mêmes générés par d'autres paramètres (α en l'occurrence dans le processus précédent).
  - $\boldsymbol{\theta}$:
    θ représente **la proportion exacte des *topics*** dans chaque document, à l'instar des α_i donnant une proportion moyenne seulement
  - $\boldsymbol{z}$:
    Les z_i, $i \in \{1, ..., N_i\}$, sont les *topics* (classes) associées à **chaque mot w_i d'un document**

1.3.2 Intérêt des variables latentes

La dépendance entre les différents paramètres introduits précédemment peut être visualisée ainsi :

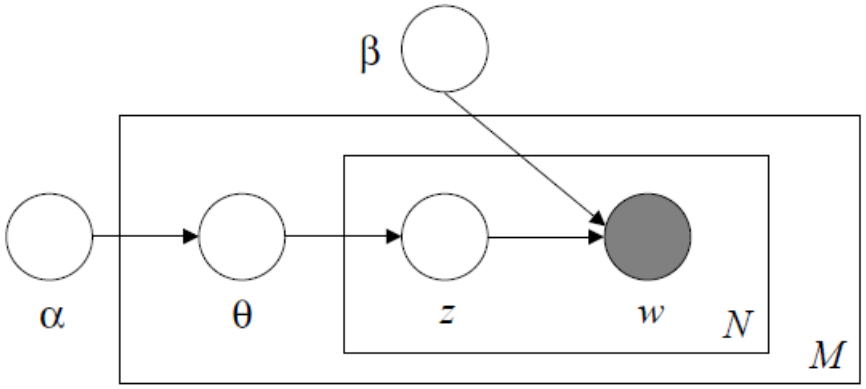


FIG. 1.2: Latent Dirichlet Allocation - Lien entre paramètres

N.B : α et β sont appelés hyperparamètres du modèle, à l'inverse de θ, z, et w qui sont des paramètres latents. N correspond à N_i pour un document d donné.

Les variables θ et z présentent un intérêt majeur en ce qu'elles **hiérarchisent le modèle et évaluent chaque "composante" du corpus** (corpus entier, document, mots). On distingue en effet trois "niveaux" distincts sur la Figure 1.3 :

- Les hyperparamètres α et β sont des paramètres globaux définis pour un corpus C tout entier ; les autres variables sont générées à partir de celles-ci
- θ est défini pour un document ; sa densité de probabilités s'exprime à partir des propropriétés de la loi de Dirichlet vues en section 1.2.2 :

$$p(\theta|\alpha) = \frac{\Gamma(\sum_{i=1}^{K}\alpha_i)}{\prod_{i=1}^{K}\Gamma(\alpha_i)}\theta_1^{\alpha_1-1}...\theta_K^{\alpha_K-1}$$

- Enfin z et w sont propres à chaque mot. On donne la probabilité jointe d'une séquence de mots w et d'une séquence de topics z (conséquence de De Finetti) :

$$p(w,z) = \int p(\theta)\left(\prod_{n=1}^{N} p(z_n|\theta)p(w_n|z_n)\right)d\theta$$

1.3.3 Comparaison à d'autres modèles

Vue sommaire des autres modèles existants

Voici un bref "état de l'art" des *Topic models*, modèles probabilites analysant la structure d'un corpus de documents pour en dégager des thèmes sous-jacents :

1. **Le modèle *unigram* :**
   Modèle très simple où :
   (a) Les documents sont générés de façon indépendante
   (b) Les mots de chaque document sont générés de façon indépendante, à partir d'une mesure de probabilité discrète.
   La distribution associée est :
   $$P_d(w_1,...,w_N) = \prod_{n=1}^{N} p(w_n)$$
2. **Le modèle *mixture of unigram* :**
   On introduit ici une variable latente "topic" Z à valeurs dans $\{1,...,K\}$, où K est le nombre de classes. une hypothèse supplémentaire par rapport au *unigram* :

– Chaque document est associé à un *unique* topic
La distribution associée est :

$$P_d = \sum_{z=1}^{K} p(z) \prod_{n=1}^{N} p(w_n|z)$$

3. **Le modèle *pLSI* :**
Le pLSI (*probabilstic Latent Semantic Indexing*) fixe dès le départ le nombre M de documents, générés de façon indépendante. La génération se fait mot par mot ; chaque mot correspond à la réalisation d'un triplet de variables aléatoires (D, W, Z) où :
– $D \in \{1, ..., M\}$ est l'indice du document dans lequel se trouve le mot
– $W \in \{1, ..., V\}$ est l'indice du mot dans le dictionnaire
– $Z \in \{1, ..., K\}$ est le topic sous-jacent à cette occurrence du mot
L'hypothèse d'échangeabilité sur les mots est vérifiée aussi dans ce modèle.
La distribution d'une observation (d, w) est :

$$p(w, d) = p(d) \sum_{z=1}^{K} p(w|z)p(z|d)$$

Avantages/Inconvénients du *LDA*

On peut rapprocher le *LDA* du *pLSI* en ce qu'ils se basent sur la même approche de génération d'un corpus, au niveau des *mots* et pas seulement des documents. Ces deux modèles fournissent beaucoup plus d'information sur le corpus que les deux modèles *unigram* qui sont relativement simplistes. Ils ont en effet l'avantage majeur de tenir compte explicitement de la polysémie des mots, attribuant ainsi un thème à un document de façon plus précise.

Le LDA se distingue du pLSI sur deux points :

1. Sa complexité n'augmente pas avec le nombre de documents ; autrement dit le nombre de paramètres n'augmente pas quand on ajoute des documents au corpus ce qui rend le modèle moins sensible au problème d'*overfitting*, récurrent en Fouille de Données
2. Il est plus complet, au sens où tous les paramètres ont une loi générative, au niveau des documents notamment ($\theta \sim Dir(\alpha)$ dans le LDA contre liste d'indices pour D dans le pLSI)

Le principal inconvénient du LDA est la difficulté d'estimation des paramètres, les distributions données en section 1.3.2 n'étant pas calculables directement en pratique à cause du couplage entre β et θ. D'où la nécéssité de déterminer des estimateurs pour ces paramètres.

Chapitre 2

Estimation des paramètres du *LDA*

L'estimation des paramètres constitue une part cruciale de la mise en application du *LDA* sur le corpus 20NewsGroups étudié . Pour cela, deux types d'algorithmes ont été implémentés, issus de deux techniques d'estimation possibles (structurelle et stochastique). Le premier algorithme, structurel, est l'EM variationnel (*Expectation & Maximisation*). Le second, stochastique, est l'échantillonage de Gibbs (*Gibbs sampling*).

2.1 EM variationnel

2.1.1 Principe de l'algorithme

L'algorithme EM est un algorithme itératif efficace pour calculer un estimateur de maximum de vraisemblance (*EMV*) dans le cas de données cachées ou manquantes. A travers l'EMV, on souhaite déterminer les paramètres du modèle pour lesquels les données observées sont les plus probables.

Chaque itération de l'EM se divise en deux étapes :

1. La ***E-step*** (*Expectation step*)
   On estime les données cachées/manquantes à partir des données observées et de l'estimation courante des paramètres du modèle
2. La ***M-step*** (*Maximisation step*)
   On maximise la fonction de vraisemblance sous l'hypothèse que les données sont connues. L'estimation des données cachées est utilisée en lieu et place de celles-ci

La convergence est assurée par l'augmentation de la vraisemblance à chaque itération, elle-même garantie par l'algorithme.

2.1.2 Evaluation des paramètres du *LDA*

Initialisation de β

On rappelle qu'on a fixé toutes les composantes de l'hyperparamètre α à 1 en section 1.3.1. Reste donc à initialiser la matrice β. Cette étape a une grande importance pour le résultat de l'algorithme puisqu'on a vu en Figure 2.1 que β est directement liée aux séquences de mots w. Une bonne initialisation de ce paramètre doit se traduire par un taux de prédiction satisfaisant quant aux classes associées à chaque document.

Pour ce faire, on sélectionne un "échantillon représentatif" de documents du corpus.

Par exemple, on considère 50 documents du corpus 20NewsGroups, répartis de façon suffisamment équilibrée entre les 20 classes expertes, et pour chacun d'eux, on peut encore choisir le nombre de mots que l'on retient pour entraîner le paramètre $(5, 10, 100, ..., N)$. Puis on calcule les β_{ij} à l'aide de la formule $(*)$

Estimateurs de θ et z

Afin de découpler les variables β et θ, on passe par deux variables intermédiaires γ et ϕ qui vérifient une hypothèse classique de Probabilités appelée *mean-field assumption*. L'idée générale derrière cette hypothèse est qu'il existe un couple (γ^*, ϕ^*) qui minimise une divergence de Kullback entre deux espaces de distribution. Une illustration de cette idée figure sur le schéma 2.1 .

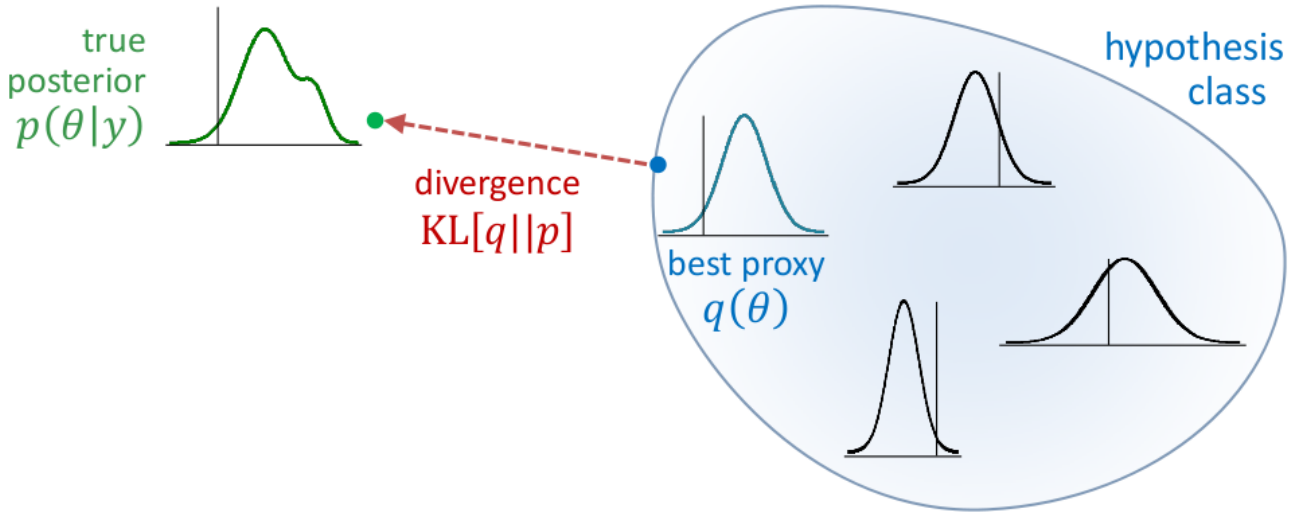


FIG. 2.1: Schéma illustratif de la *mean-field assumption*

Le découplage peut se visualiser ainsi, en association avec la Figure 1.2 :

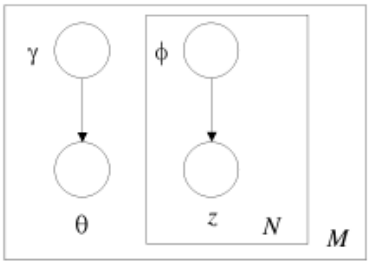


FIG. 2.2: Visualisation des estimateurs γ et ϕ

La détermination explicite de γ et ϕ passe par la résolution du problème d'optimisation suivant :

$$(\gamma^*, \phi^*) = arg\,min_{(\gamma,\phi)}\, D(q(\theta, z|\gamma, \phi)||p(\theta, z|w, \alpha, \beta))$$

où q est une famille de distributions sur les variables latentes θ, z et w.
On obtient les formules suivantes :

$$\phi_{ni} = \beta_{iw_n} exp\left(\Psi\big(\gamma_i\big) - \Psi\Big(\sum_{j=1}^{N} \gamma_j\Big)\right)$$

$$\gamma_i = \alpha_i + \sum_{n=1}^{N} \phi_{ni}$$

où $\Psi(x) = \frac{d}{dx} log(\Gamma(x))$, $x \geq 0$ (fonction *digamma*)

Code de l'algorithme

Commentaire :

- Dans la E-step, le critère de convergence choisi est un critère empirique tiré de [1] (cf. Réf. bibliographiques) : pour éviter d'avoir à recalculer la vraismblance à chaque fois, on itère sur le nombre de mots pour chaque document (moins coûteux en temps de calcul et en mémoire).
- Dans la M-step, l'initialisation de β à η, choisi arbitrairement à 1 est une opération de lissage (*smoothing*) qui permet d'éviter d'avoir une probabilité d'occurrence nulle pour un mot, notamment dans le cas de l'ajout d'un nouveau document au corpus contenant des mots nouveaux.

FIG. 2.3: EM variationnel appliqué à un corpus de documents

initialization of ϕ :
for $i = 1$ to K **do**
  for $n = 1$ to N **do**
    $\phi_{ni} = \frac{1}{K}$
  end for
end for

initialization of γ :
for $i = 1$ to K **do**
  $\gamma_i = \alpha_i + \frac{N}{K}$
end for
E-step :
while count $\leq$ wordCount **do**
  for $n = 1$ to N **do**
    for $i = 1$ to K **do**
      $\phi_{ni} = \beta_{iw_n} exp(\Psi(\gamma_i))$
    end for
    normalization of ϕ_n to sum to 1
  end for
  $\gamma = \alpha + \sum_{n=1}^{N} \phi_n$
  count += 1
end while

M-step :
Initialisation de tous les β_{ij} à η :
$\eta = 1$
for $i = 1$ to K **do**
  for $j = 1$ to V **do**
    $\beta_{ij} = \eta$
    sumBeta$[i] += \eta$
  end for
end for

Pour chaque mot de chaque document, on attribue le topic z correspondant :
for each document **do**
  for $n = 1$ to N **do**
    $\beta_{iw_n} += \phi_{ni} \times freq(w_n)$
    sumBeta$[i] += \phi_{ni} \times freq(w_n)$
  end for
end for

On normalise β :
for $i = 1$ to K **do**
  for $j = 1$ to V **do**
    $\beta_{ij} += \frac{\beta_{ij}}{sumBeta[i]}$
  end for
end for

Optimisation des performances

Un des défauts de l'algorithme précédent est son temps d'exécution relativement long (près de 120 min pour une E-step + une M-step). Cela était dû à un manque de parallélisation des calculs dans l'algorithme proposé (tiré de [1]). Nous avons ainsi pris l'initiative d'optimiser ce code (l'E-step en particulier) afin de le rendre plus rapide. L'implémentation proposée en Figure 2.4 a permis un gain de temps considérable : une itération de l'algorithme ne prend désormais plus que 10 min !

FIG. 2.4: Implémentation personnelle optimisée de l'E-step

E-step :
while count<wordCount **do**
  for $i = 1$ to K **do**
    psi_gamma[i] = $\Psi(\gamma_i)$
  end for
  for $n = 1$ to N **do**
    for $i = 1$ to K **do**
      $\phi_{ni} = \beta_{iw_n} exp(psi\_gamma[i])$
      norm_phi += ϕ_{ni}
    end for
  end for
  for $i = 1$ to K **do**
    $\phi_{ni} = \phi_{ni} \div norm_phi$
    if $n > 0$ **then**
      $\gamma_i += \phi_{ni} \times freq(w_n)$
    else
      $\gamma_i = \alpha + \phi_{ni} \times freq(w_n)$
    end if
    norm_phi = 0
  end for
  count += 1
end while

2.2 Echantillonnage de Gibbs

2.2.1 Théorie

Par opposition à l'EM variationnel qui génère une suite de paramètres qui converge vers les paramètres estimés, on va simuler la loi générant ces paramètres, particulièrement adapté car on a un modèle génératif complet. Pour simuler ces lois, on va implémenter un algorithme d'échantillonnage de Gibbs. Cet algorithme utilise une méthode de Monte-Carlo par chaînes de Markov. Une chaîne de Markov est un processus de Markov à temps discret. Plus précisement, un processus de Markov est un processus stochastique qui possède la propriété suivante : la prédiction du futur, sachant le présent, ne va pas être enrichie par la connaissance des évènements précedent le présent.

En Mathématiques, on appelle méthodes de Monte-Carlo les techniques permettant d'évaluer une quantité déterministe à l'aide de l'utilisation de tirages aléatoires. Plus précisement, les méthodes de Monte-Carlo par chaîne de Markov permettent de générer un vecteur X_{t+1} à partir du vecteur X_t. Le premier vecteur est généré aléatoirement et par propriété des processus de Markov, on peut calculer la distribution en t+1 seulement à partir de celle calculée en t. Dans le cadre de l'échantillonneur de Gibbs, pour générer les vecteurs, on procède ainsi :
Pour $X_t = x_1^t, ..., x_n^t$,

$$\begin{aligned} x_1^{t+1} &\sim p(x_1|x_2^t, x_3^t ..., x_n^t) \\ x_2^{t+1} &\sim p(x_2|x_1^t, x_3^t ..., x_n^t) \\ &\vdots \\ x_n^{t+1} &\sim p(x_n|x_1^t, x_2^t ..., x_{n-1}^t) \end{aligned}$$

Cet algorithme utilise une marche aléatoire sur les chaînes de Markov. Ici, nous allons créer une chaine de Markov sur les topics associés à chaque mot de chaque document. L'échantillonnage de Gibbs assure que la loi de distribution genérée aléatoirement va converger vers la loi de distribution voulue au bout d'un certain nombre d'itérations. Cette distribution est représentée par z (que l'on va définir dans la section suivante) qui s'avère être une statistique exhaustive. Grace à cette propiété, on peut, à partir des résultats obtenus suite à l'éxecution de l'algorithme, estimer les paramètres θ et β.

Un défaut notoire de cet algorithme est l'absence de critère de convergence bien que la théorie affirme que l'échantillonnage de Gibbs converge. Il n'existe pas d'outil permettant de construire un critère de convergence pour cet algorithme. Toutefois, l'échantillonnage de Gibbs reste un algorithme puissant et efficace, et donc très utilisé en pratique. Pour remédier au problème d'absence de critère de convergence, nous avons extrait d'un article un nombre d'itérations choisi empiriquement.

2.2.2 Implémentation

Ici, on introduit un nouvel hyperparamètre μ. C'est le paramètre de la loi de Dirichlet qui génère β.

Pour l'implémentation, nous avons crée une classe Gibbs_sampling qui nous permet de stocker les compteur et les variables utiles pour calculer ensuite θ et β .Les attributs de cette classe sont les suivants :

- Des entiers K, M, V qui représentent respectivement le nombre de classes, le nombre de documents, et taille du vocabulaire
- Les paramètres α et μ

- Une matrice *freq* de taille $V * K$ qui représente le nombre de fois que chaque mot apparait dans chaque classe
- Un vecteur *freqTotal* de taille K qui représente le nombre de mots associés à chaque classe
- Une matrice *top* de taille $M * K$ qui représente le nombre de mots dans chaque document assignés à chaque classe
- Une matrice z qui, pour chaque mot de chaque document lui assigne une classe.
- Un vecteur *sumtop* de taille M qui représente la somme des mots associés à toutes les classes dans chaque document

Ces données suffisent pour obtenir les estimations des paramètres θ et β.
L'implémentation de l'algorithme est donnée en Figure 2.1 ci-dessous.

N.B : Les initialisations de μ et α sont extraites de la littérature.

---

initialization of z random $[\![1, K]\!]$
initialization of top, freq, freqTotal, sumtop.
$\alpha = 50/K$, $\mu = 0,1$
for $iter = 1, iter \leq nbIter$ **do**
  for $d = 1, d \leq M$ **do**
    for $i = 1, i \leq N$ **do**
      $w = currentWordIndex$
      $t = z_{d,i}$
      $top_{d,t} - = 1$
      $freq_{w,t} - = 1$
      $freqTotal_t - = 1$
      $sumtop_d - = 1$
      for $k = 1, k \leq K$ **do**
        $p(z = k|.) = \frac{top_{d,k}+\alpha}{sumtop_d+K*\alpha} \frac{freq_{w,k}+\mu}{freqTotal_k+\mu*V}$
      end for
      topic = sample from p(z|.)
      $z_{d,i} = topic$
      $top_{d,j} + = 1$
      $freq_{w,j} + = 1$
      $freqTotal_j + = 1$
      $sumtop_d + = 1$
    end for
  end for
end for

FIG. 2.5: Algorithme *Gibbs sampling* appliqué à un corpus de documents

Explication du sample :

Nous avons crée une variable aléatoire r qui suit une loi uniforme sur $\left[0, \sum_{k=0}^{K} p(z = k|.)\right]$. On considère la quantité $q_k = \sum_{i=0}^{k} p(z = i|.)$.
Comme r suit une loi uniforme, p($r \in [q_{k-1}, q_k]$)= p(z=k|.). En ce qui concerne l'implémentation, on commence par générer r. Ensuite, tant que $r \leq q_k$, on incrémente k. Enfin, quand la condition est satisfaite, l'algorithme renvoie le k courant.

L'algorithme ci-dessus modifie des attributs de la classe qui vont ensuite être utilisés pour calculer une estimation des paramètres β et θ. Les formules sont les suivantes :

$$\beta_{k,w} = \frac{freq_{w,k}+\mu}{freqTotal_k+\mu*V} \ , \ \theta_d = \frac{top_{d,k}+\alpha}{sumtop_d+\alpha*K}$$

Contrairement à l'EM variationnel, il n'y a pas d'initialisation de β grâce à l'hyperparamètre μ qui lui, est initialisé plus simplement que β. L'exécution d'une itération de cet algorithme est très rapide, beaucoup plus que pour l'EM variationnel : cela prend en moyenne entre 2 et 2,5 secondes par itération.

Néanmoins le Gibbs sampling fournit moins d'information que l'algorithme EM car z ne donne que la classe prédominante associée à un mot alors que dans l'algorithme EM, le ϕ nous donne les probabilités d'un mot d'appartenir à chacune des classes. Cependant, dans notre étude nous n'avons jamais eu à utiliser cette information.

Chapitre 3

Résultats

Le modèle LDA peut être appliqué dans de nombreux domaines de la Fouille de Données comme l'extraction des thèmes abordés dans un corpus, les systèmes de recomandation, ou la classification de documents de diverses natures. Dans le cadre de cette étude, nous nous sommes penchés sur les performances de ce modèle dans le cadre de la classification de documents textuels. Pour ce faire, nous avons utilisé le corpus de documents 20NewsGroups constitué de 19997 documents répartis équitablement en 20 classes. Cette classification a priori permet d'évaluer les performances de notre modèle. L'essentiel de notre étude s'est portée sur la classification non supervisée, c'est à dire sans connaissance a priori de la classe de chaque document. Nous avons également abordé, dans une moindre mesure, la classification supervisée.

Nous avons utilisés les deux algorithmes présentés au chapitre précédent pour obtenir les résultats, ce qui nous a permis de comparer leurs performances.

3.1 Classification non supervisée

La classification non supervisée a pour but de regrouper en un certain nombre de catégories des documents partageant des caractéristiques communes. Dans notre cadre, il s'agit de regrouper les documents abordant les mêmes thèmes sans connaissance préalable de ces derniers. Le modèle LDA impose de fixer à l'avance le nombre de thèmes. On peut donc parler de classification semi-supervisée dans le sens où le nombre de classes est donnée mais les thèmes de ces classes sont inconnus.

Nous avons utilisé le modèle LDA de deux manières différentes. Dans le premier cas, les documents ont été classés directement grâce à l'estimation des paramètres du modèle. Dans le second cas, nous utilisons ces paramètres pour avoir une autre représentation des documents, dans un espace de dimension inférieure, sur lequel nous appliquons un algorithme de classification non supervisé : l'algorithme CEM.

3.1.1 Classification par estimation des paramètres

Une idée naïve consiste à attribuer à chaque document la classe la plus représentée dans celui-ci. Elle peut être déterminée par l'estimations des paramètres de deux manières différentes. La première méthode consiste simplement à selectionner la classe qui maximise θ dans le document. Dans la deuxième méthode, on attribut à chaque mot du document une classe (cela se fait directement en regardant les $z_{d,i}$ dans le Gibbs sampling, et en prenant le plus grand ϕni pour chaque mot dans l'EM variationnel) et l'on compte le nombre de mot pour chaque classe. On attribut alors au document la classe qui comptabilise le plus de mots. Ces deux méthodes sont équivalentes. On obtient ainsi une classification du corpus.

3.1.2 Classification sur l'espace des topics

Ici, on n'utilise non pas le modèle LDA pour classer directement le corpus mais pour en tirer une nouvelle représentation des documents. Pour cela on commence par estimer les paramètres du modèle avec l'un des algorithmes décrit précédemment. On attribue à chaque mot de chaque document une classe (comme dans la section précédente). On compte le nombre de mots correspondant à chaque classe et l'on obtient pour tous les documents la proportion de chaque classe. Les documents sont alors symbolisés non plus par la fréquence des mots qui le constituent mais par la répartition selon chaque classe (ou topic). Par exemple, un document contenant 3 mots de la classe 1, 5 mots de la classe 4, et 1 mot de la classe 5, aura la représentation suivante : (1 : 3 4 : 5 5 : 1). Ainsi, on passe d'une représentation dans l'espace des mots de dimension V à une représentation dans l'espace des topics de dimension K. On applique alors aux documents représentés dans l'espace des topics un algorithme de classification non supervisé, l'algorithme CEM, pour obtenir la classification des documents selon le nombre de classes voulu. Il faut ici bien distinguer le nombre de classes K utilisées dans le modèle LDA, qui représente le nombre de topics selon lesquels les documents seront représentés, et le nombre de classes K' issue de la classification de l'algorithme CEM, correspondant au 20 classes dans la base de donnée 20NewsGroups. Pour commencer, nous prendrons $K = K'$.

3.2 Analyse des résultats

3.2.1 Utilisation de la F-Mesure

Le principal problème pour évaluer une classification non supervisée est de pouvoir mesurer la pertinence du clustering. Dans notre cas, nous disposons d'une base de données bénéficiant d'un clustering expert. Nous pouvons donc évaluer notre clustering en fonction de celui-ci, à condition de pouvoir identifier clusters expert et clusters produit par l'algorithme. Pour cela, on associe à chaque cluster réalisé le cluster expert qui y est le plus représenté. On introduit alors les quantités suivantes :
On note i le cluster produit par l'algorithme et j le cluster expert associé

- $C(i,j) =$ Nombre de document de classe j dans le cluster i
- $T_{clu}(i) =$ Nombre de document dans le cluster i
- $T_{exp}(j) =$ Nombre de document dans le cluster expert j
- $P_i = \frac{C(i,j)}{T_{clu}(i)}$ mesure de précision du cluster i
- $R_i = \frac{C(i,j)}{T_{exp}(j)}$ mesure de rappel du cluster i
- $F_i = 2\frac{P_i * R_i}{P_i + R_i}$ F-mesure du cluster i

  On obtient donc $F_i = \frac{2*C(i,j)}{T_{clu}(i)+T_{exp}(j)}$

La F-Mesure permet donc de mesurer l'adéquation du clustering produit avec le clustering expert. Cependant, dans le cas où un cluster ne correspond pas avec une classe prédéfinie, cela ne permet pas a priori de savoir si le cluster est tout de même pertinent, c'est-à-dire si il a réussi à capter du sens dans le corpus.

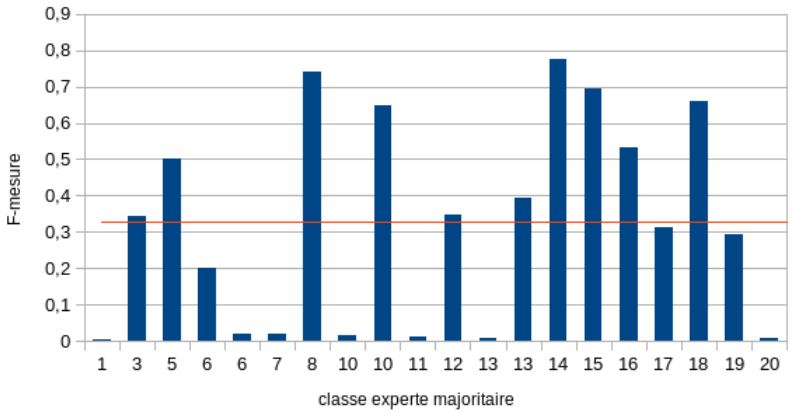


FIG. 3.1: F-mesure pour le clustering du corpus 20NewsGroups par l'EM variationnel, 24 itérations

3.2.2 Résultats par estimation des paramètres

medecine(14)	automobile(8)	thème indéfini(7)	thème indéfini(6)
medecine	car	die	heil
article	article	ihr	joachim
food	cars	proline	martillo
disease	engine	cosmoproangmar	cornerstore
medical	good	und	tonis
doctor	price	mit	santos
pain	ive	ist	ajami
patiens	oil	sie	carlo
treatment	speed	uunetbuedualpha	xnewsreader
ive	buy		fourdcom

FIG. 3.2: Top words pour les classes 6, 7, 8 et 14

Sur la Figure 3.1, on note une grande disparité entre les classes : certaines correspondent quasi-parfaitement au clustering expert (F-mesure supérieure à 0.6, voire 0.7), alors que d'autres ne semblent être liées à aucun topic a priori (F-mesure quasi-nulle). La F-mesure moyenne (*macro-F*) est d'environ 0.32.

Lorsqu'il y a des doublons, on remarque qu'on a toujours une des deux classes a une très, alors qu'elle est très faible pour l'autre. C'est le cas pour les classes 10 et 6.

Le tableau ci-dessus regroupe les "top words" d'un topic qui correspondent en fait aux termes les plus probables au sein d'une classe. Ils permettent d'évaluer la sémantique du clustering.

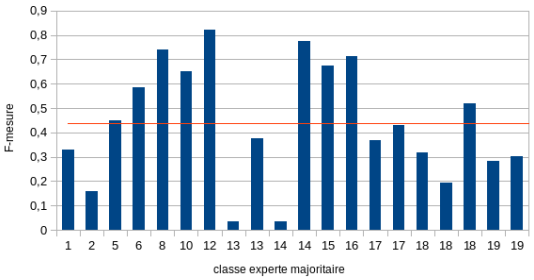
On constate que pour les deux premières colonnes, qui représentent les classes 14 et 8, que leurs thèmes sous-jacents respectifs sont mis en évidence de façon claire, ce qui n'est pas le cas pour les deux autres. Une forte F-mesure semble donc coïncider avec une bonne sémantique pour un cluster.

Enfin, quelques mots d'usage fréquent ou sans sens défini a priori ("article", "ive", "uunet-

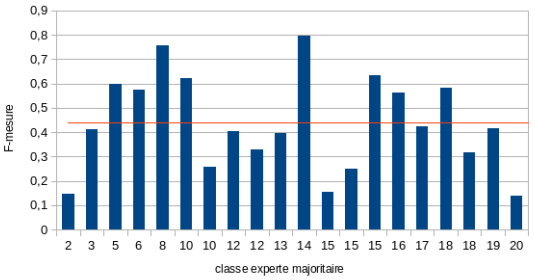
buealpha", ...) baissent la performance car ne discriminent pas suffisamment les topics. D'où l'importance d'un bon pré-traitement du corpus (cf. Section 1.1, Vectorisation des documents).

Néanmoins, on peut trouver un certain sens a posteriori pour l'une d'entre elles (classe 7 : mots en Allemand).

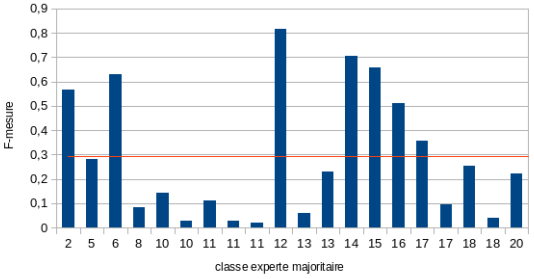
3.2.3 Comparaison des résultats entre espace des topics et espace des mots



(a) Classification issue de l'algorithme CEM à partir de l'espace des topics généré par Gibbs sampling ; **macro-F : 0,437969**



(b) Classification issue de l'algorithme CEM à partir de l'espace des topics généré par EM variationnel ; **macro-F : 0,438138**



(c) Classification issue de l'algorithme CEM à partir de l'espace des mots ; **macro-F : 0,29205**

FIG. 3.3: Clustering du corpus 20NewsGroups par l'algorithme CEM dans différents espaces

De façon générale, le clustering est meilleur dans l'espace des topics, indépendamment de l'algorithme utilisé (0.43 dans l'espace des topics pour la macro-F contre 0.32 dans l'espace des

mots).

Plus précisément, on observe que les classes 14, 15 et 16 sont bien distinguées dans les 3 cas. Elles correspondent aux topics "science" et "christianisme", ce qui explique une telle mise en relief car le vocabulaire lié à ces thèmes est très spécifique, et donc discriminant. La classe 7 n'apparaît dans aucun des espaces ; cela confirme ce qui a été vu en Figure 3.1, où la F-mesure de cette classe était très faible (< 0.05). Cela est dû à la faible sémantique rattachée à ce topic : thème divers, étiqueté *miscalleneous* dans le clustering expert.

La F-mesure obtenu par l'algorithme CEM est meilleure que celle obtenue par le clustering effectué sur les paramètres estimés du modèle LDA (cf FIG 3.1).Ce résultat semble logique dans le sens où l'on classait les documents en fonction d'un unique topic (le topic majoritaire). Ici, on classe les documents sur l'ensemble des topics trouvés par l'algorithme.

On voit que certaines classes expertes du corpus partagent la même sémantique. Par exemple, les classes 2 à 5 traitent des sujets en rapport avec l'informatique, les classes 8 et 9 sont liées par le domaine des véhicules motorisés, et les classes 10 et 11 partagent le thème du sport (cf annexe : tableau des classes expertes). La différence entre les résultats obtenus dans l'espace des topics et ceux obtenus dans l'espace des mots est la suivante : seul l'espace des topics représente au moins une classe de chaque "famille de classes" avec une F-mesure convenable. Autrement dit, dans l'espace des topics, tous les grands thèmes ont été captés par l'algorithme. Ce qui nous amène à nous demander si le clustering expert n'a pas un degré de précision trop élevé. Ainsi, en regroupant certains cluster expert partageant la même sémantique, nous pourrions peut être améliorer le clustering géneré par l'algorithme sans perdre trop d'informations.

3.2.4 Réduction du nombre de classes expertes

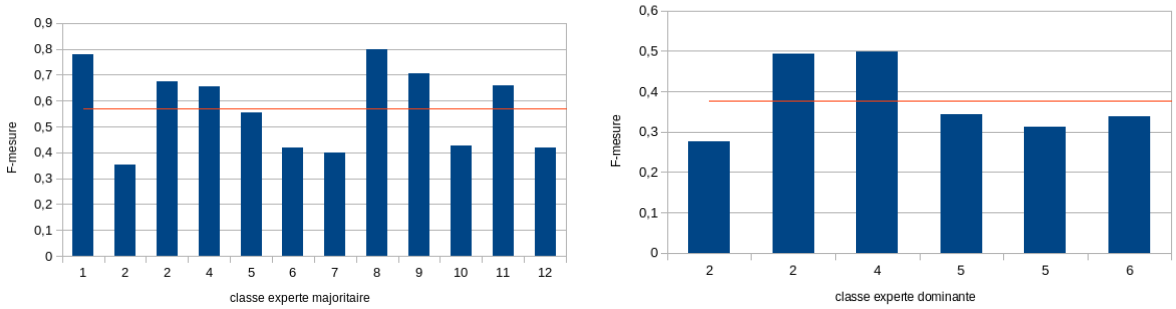


FIG. 3.4: Evolution de la F-mesure en fonction du nombre de classes expertes

Sur le diagramme de gauche de la Figure 3.4, certains thèmes proches ont été regroupés dans une même catégorie, réduisant le nombre de classes expertes à 12. Exemple : cinq catégories liées à l'Informatique ont été fusionnées, les Sports ont été rassemblés (sports automobiles compris), les divers sous-thèmes liés à la religion également, ... Seules les Sciences (Médecine, Spatial, ...) ont été laissées telles quelles car nous avons estimé que leur vocabulaire respectif était trop propre à la discipline concernée pour pouvoir être rassemblée en une même classe. L'idée de ce regroupement est de proposer un clustering expert plus pertinent que celui à 20 classes.

Et les résultats s'avèrent bien meilleurs pour ce nouveau clustering. On a une macro-F égale à 0,57056, soit un gain de 0.14 par rapport à l'ancien (0.43). De plus, aucune des 12 F-mesures n'est inférieure à 0,35, et la moitié d'entre elles sont supérieures à 0,60. La F-mesure étant relative à un clustering expert, l'issue observée ici peut être jugée plus pertinente, même si la classification est

a priori moins précise.

Il peut être alors intéressant de voir si une nouvelle réduction du nombre de classes expertes améliore encore les F-mesures pour chacune des classes. **Sur le diagramme de droite de la Figure 3.4**, on a regroupé en plus les Sciences sous une même classe contrairement au regroupement précédent. Et on s'aperçoit que cela entraîne une baisse significative des F-mesures par rapport au clustering à 12 topics : la macro-F tombe à 0,377179. Cette baisse de performance est due à un mauvaise réorganisation des clusters experts, comme par exemple les différentes Sciences évoquées plus haut.

Conclusion

Le *Latent Dirichlet Allocation* est un modèle efficace en terme de classification de documents, notamment grâce au nombre réduit de paramètres à estimer. Les résultats observés à partir des deux algorithmes étudiés ont révélé que ce-dernier capte les grands thèmes d'un corpus de texte de manière non supervisée pourvu que ceux-ci ne soient ni trop généralistes, ni trop spécialisés.Au cours de nos tests, nous avons pu observer les performances de l'EM variationnel et du Gibbs Sampling. Il s'est avéré qu'à degré informatif égal, l'échantilloneur de Gibbs est bien plus rapide. Par ailleurs, nous n'avons pas itéré l'EM suffisamment de fois pour voir s'il est plus précis. Au bout d'environ 25 itérations, l'amélioration ne paraissait plus significative. On pourrait donc se demander à juste titre si un très grand nombre d'itérations a véritablement un impact sur le résultat recherché en sortie d'algorithme.

Afin de mesurer la qualité du clustering renvoyé par l'algorithme, il faut le comparer à un clustering expert judicieusement choisi. Comme nous l'avons vu, il n'y a a priori pas de solution absolue pour ce problème ; un compromis doit être réalisé selon les objectifs de classification. Ayant mis en évidence un lien entre la sémantique des "top words" et la F-mesure des clusters, on peut, dans le cas ou l'on ne dispose pas de clustering expert, choisir le nombre de classes en analysant manuellement les "termes-clés".

Une piste de recherche ultérieure pourrait être d'étudier l'influence de la dimension de l'espace des topics généré avec le LDA.

Références bibliographiques

[1] David M. Blei, Andrew Y. Ng, and Michael I. Jordan, *Latent Dirichlet Allocation*, Journal of Machine Learning Research (2003), no. 3, 993-1022

[2] Matthew D. Hoffmann, David M. Blei, and Francis Bach, *Online Learning for Latent Dirichlet Allocation*, 2010

[3] William M. Darling, *A Theoretical and Practical Implementation Tutorial on Topic Modeling and Gibbs Sampling*, School of Computer Science, University of Guelph, 2011

[4] http ://fr.wikipedia.org/wiki/Loi_de_Dirichlet

[5] Sean Borman, *The Expectation Maximisation Algorithm, A Short Tutorial*, 2004

Annexe

Indice	Intitulé	Indice	Intitulé
1	« ALT »	11	« RecHockey »
2	« CompG »	12	« SciCrypt »
3	« CompOS »	13	« SciElectronics »
4	« CompIBM »	14	« SciMed »
5	« CompMAC »	15	« SciSpace »
6	« CompWindows »	16	« SocReligion »
7	« MISC »	17	« TalkGuns »
8	« RecAutos »	18	« TalkMideast »
9	« RecMotos »	19	« TalkMisc »
10	« RecBaseball »	20	« TalkReligion »

FIG. 3.5: Clustering expert initial du corpus 20NewsGroups