Analyse Factorielle Neuronale pour Documents Textuels

Mathieu Delichère (1) & Daniel Memmi (2)

(1) Amoweba
1 ave Berthollet, 74000 Annecy (France)
mathieu@amoweba.com
(2) Leibniz-Imag
46 ave Félix Viallet, 38000 Grenoble (France)
memmi@imag.fr

Résumé - Abstract

En recherche documentaire, on représente souvent les documents textuels par des vecteurs lexicaux de grande dimension qui sont redondants et coûteux. Il est utile de réduire la dimension des ces représentations pour des raisons à la fois techniques et sémantiques. Cependant les techniques classiques d'analyse factorielle comme l'ACP ne permettent pas de traiter des vecteurs de très grande dimension. Nous avons alors utilisé une méthode adaptative neuronale (GHA) qui s'est révélée efficace pour calculer un nombre réduit de nouvelles dimensions représentatives des données. L'approche nous a permis de classer un corpus réel de pages Web avec de bons résultats.

For document retrieval purposes, documents are often represented by high-dimensional lexical vectors, which are costly and redundant. Reducing vector dimensionality is then useful for both technical and semantic reasons. Classical data analysis methods such as PCA cannot unfortunately process vectors of very high dimension. We have used instead an adaptive neural network technique, the Generalized Hebbian Algorithm (GHA), which makes it possible to reduce high-dimension spaces. This approach allowed us to cluster areal end-user corpus of Web pages with very significant results.

Mots-clés - Keywords

Recherche documentaire, modèle vectoriel, réduction de dimension, analyse factorielle, ACP, GHA, réseaux de neurones.
Information retrieval, vector-space model, dimensionality reduction, data analysis, PCA, GHA, neural networks.

1 Introduction

La croissance accélérée d'Internet et du Web donne une importance grandissante au traitement numérique de documents textuels. Pour effectuer les diverses tâches de classification, recherche et filtrage de documents, il faut d'abord représenter les textes de manière à la fois économique et significative. On sait que le modèle vectoriel est probablement l'approche la plus courante : on représente un texte par un vecteur numérique obtenu en comptant les éléments lexicaux les plus pertinents (Salton & McGill 1983)(Manning & Schütze 1999).
Ces vecteurs sont fournis par des prétraitements simples. On commence généralement par éliminer les mots grammaticaux (articles, prépositions, etc.) et par réduire les variantes morphologiques à une forme commune (souvent appelée *terme*). Puis on compte les occurrences des termes les plus importants de manière à représenter chaque document par un vecteur dans l'espace des termes. Un corpus de documents donne donc lieu à une matrice document-terme (Fig. 1). Cette représentation assez simpliste permet ensuite d'appliquer les opérations vectorielles usuelles avec des résultats sémantiquement pertinents dans l'ensemble.

	terme *1*	terme *2*	terme *3*	...	terme *n*
document *1*	x_{11}	x_{12}	x_{13}	...	x_{1n}
document *2*	x_{21}	x_{22}	x_{23}	...	x_{2n}
...	...	...	...	...	...
document *m*	x_{m1}	x_{m2}	x_{m3}	...	x_{mn}

Fig. 1 : Matrice document-terme (x_{ij} = fréquence).

Cette approche produit malheureusement des vecteurs lexicaux de très grande dimension (à grand nombre de traits). Il est fréquent d'aboutir à un lexique de trois à cinq mille termes, et donc à des vecteurs de même dimension. De tels vecteurs sont coûteux à stocker et à traiter. Ils sont pourtant très creux, car contenant généralement plus de 90% de valeurs nulles. Comme les termes sont aussi fortement corrélés entre eux dans un corpus donné, on a affaire à une représentation tout à fait redondante.
De plus ces vecteurs redondants sont trop gros, inutilement détaillés et donc peu lisibles pour un utilisateur humain qui voudrait s'en servir pour évaluer rapidement le contenu d'un document et chercher à voir les relations entre divers documents. En particulier les thèmes caractéristiques d'un corpus ne ressortent pas de manière évidente dans une telle représentation brute.
Enfin les corrélations entre traits lexicaux révèlent des synonymies entre termes, qui ont des conséquences négatives pour l'indexation et la recherche. On sait en effet que des documents voisins sémantiquement peuvent très bien ne pas contenir les mêmes termes. Détecter les relations entre termes permettra d'améliorer la recherche de documents.
Pour des raisons d'efficacité comme de lisibilité de la représentation, il serait donc utile de trouver une représentation plus compacte des documents. L'idée fondamentale est de trouver la dimension intrinsèque du domaine, c'est-à-dire la dimension minimale permettant de représenter les données sans perte d'information.
Par ailleurs nos intuitions spatiales habituelles se révèlent fausses et trompeuses en grande dimension. En particulier on se heurte au phénomène de "l'espace vide" : le volume de l'espace total croît très vite avec la dimension, ce qui fait que les données risquent de se

retrouver isolées dans cet espace. Il faudrait donc beaucoup plus de données pour bien représenter un domaine.
Pour toutes ces raisons, il est important d'essayer de réduire la dimension des données avant tout traitement ultérieur. La présence de corrélations multiples incite à chercher de nouvelles dimensions plutôt qu'une sélection parmi les traits originels. Il existe pour cela toute une gamme de méthodes permettant de calculer un nombre réduit de dimensions pour un ensemble de données. On les regroupe sous le nom général d'analyse factorielle, mais la méthode la plus connue est l'Analyse en Composantes Principales (ACP).
Malheureusement les réalisations courantes de l'ACP ne permettent pas en pratique de traiter des vecteurs de très grande dimension, ce qui est justement le cas dans le domaine des documents. Nous avons alors choisi d'utiliser une version neuronale de l'ACP, appelée Algorithme Hebbien Généralisé, permettant de traiter de tels vecteurs avec de bons résultats.

2 L'Analyse en Composantes Principales

C'est la méthode d'analyse factorielle la plus utilisée (Bouroche & Saporta 1980)(Jolliffe 1986)(Lebart et al. 2000). L'ACP consiste à calculer un nombre réduit de nouvelles dimensions, qui sont des combinaisons linéaires des dimensions originelles des données (c'est-à-dire des traits descriptifs). Ces nouvelles dimensions sont non corrélées et expriment le maximum de variance des données (en partant de données centrées sur la moyenne). Les nouveaux axes sont les vecteurs propres, ordonnés par valeurs propres décroissantes, de la matrice de covariance des données.
Autrement dit ce sont les principaux axes de dispersion du nuage de données, en ordre d'importance décroissante ; les valeurs propres correspondantes indiquent la part de variance exprimée par chaque axe. Les premiers axes rendent donc généralement compte de la plus grande partie de la variance. Les composantes principales sont les nouvelles valeurs des données sur chaque axe ainsi obtenu.
Cette méthode peut jouer un double rôle de compression des données et d'outil d'exploration dans des domaines fortement multidimensionnels. En effet les axes principaux ainsi calculés permettent à la fois une réduction des données et une interprétation plus facile du domaine traité, car les nouvelles dimensions sont souvent très significatives. Cela peut notamment se révéler intéressant pour l'analyse de données en langage naturel (Lebart & Salem 1994).
Le problème est que l'ACP demande le calcul préalable de la matrice carrée de covariance des données, qui est de taille n^2 pour des vecteurs de dimension n. Cette matrice est déjà coûteuse à calculer, et sa taille et son traitement deviennent prohibitifs en grande dimension. Ainsi des données de dimension 1000 donneraient lieu à une matrice de un million d'éléments !
L'ACP est donc difficile ou impossible à utiliser sur des vecteurs de documents textuels, dont on a vu qu'ils pouvaient comporter des milliers de traits. Diverses approches ont alors été proposées pour réduire la dimension des représentations vectorielles de textes.
On peut citer notamment la méthode appelée Latent Semantic Indexing (LSI) ou Latent Semantic Analysis (LSA), qui n'utilise pas la matrice de covariance, mais extrait de nouveaux axes directement de la matrice document-terme (Deerwester et al. 1990). L'approche consiste à effectuer une décomposition en valeurs singulières de la matrice des données pour trouver les nouveaux axes (voir Lebart et al. 2000). Cette technique peut être considérée comme une généralisation de l'extraction des vecteurs propres caractéristiques de l'ACP, généralisation permettant de traiter des matrices rectangulaires et des données non centrées.
Comme le nombre de documents est souvent plus petit que le nombre de termes, cette matrice rectangulaire est moins encombrante et moins coûteuse que la matrice de covariance. La méthode LSI semble donner de bons résultats en pratique pour l'indexation et la recherche,

mais contrairement à l'ACP, la signification théorique des nouveaux axes est loin d'être claire et ceux-ci sont peu intuitifs. D'autre part les coûts de calcul demeurent importants, car on manipule encore des matrices de grande taille.
Nous avons donc décidé de rester dans le cadre de l'ACP et d'utiliser plutôt une méthode neuronale itérative permettant de ne considérer qu'un vecteur de données à la fois sans jamais calculer la matrice de covariance.

3 L'Algorithme Hebbien Généralisé

Rappelons d'abord qu'un réseau neuronal (ou connexionniste) se compose de neurones formels connectés entre eux, par analogie avec la neurobiologie (Hertz et al. 1991)(Hérault & Jutten 1994). Un tel système consiste donc en un réseau de petits automates simples interconnectés, et c'est le fonctionnement de l'ensemble qui permet d'accomplir les tâches désirées (souvent classification ou diagnostic). Chaque neurone formel calcule la somme pondérée de ses entrées et transmet son état interne aux neurones auxquels il est connecté. Certains neurones serviront d'entrée, et d'autres de sortie, mais le traitement est distribué sur l'ensemble du réseau.
De plus chaque connexion entre neurones est affectée d'un poids modulant la transmission de l'activité. Ces poids sont ajustés graduellement par des procédures d'apprentissage à partir d'une présentation itérative des données. Cela permet d'adapter le système en fonction des entrées de manière à résoudre le problème posé. Il existe diverses méthodes, mais l'apprentissage non supervisé de type *hebbien* que nous allons utiliser ne nécessite que les données d'entrée, sans autre information.
L'avantage des réseaux connexionnistes est de répartir un traitement complexe sur un ensemble de petits automates et d'ajuster automatiquement les paramètres (les poids) du système de manière itérative. Dans le cas qui nous intéresse, on pourra donc ne considérer qu'un vecteur d'entrée à la fois, mais au prix d'un temps d'apprentissage plus ou moins long.
On cherche ici non pas à effectuer une classification, mais à détecter les corrélations entre les données. Le système va apprendre indirectement les corrélations entre traits d'un même vecteur d'entrée et d'un vecteur à l'autre en cherchant à maximiser la sortie du réseau. Autrement dit, il va extraire les composantes principales.
L'Algorithme Hebbien Généralisé est ainsi une variante neuronale de l'ACP élaborée par Sanger sous le nom de Generalized Hebbian Algorithm (GHA). Mais c'est Oja qui proposa le premier une règle d'apprentissage (Oja 1982) permettant à un simple neurone linéaire d'extraire la première composante principale de données centrées :

$$\mathbf{w}(t+1) = \mathbf{w}(t) + \eta(t)\,[y(t)\mathbf{x}(t) - y^2(t)\mathbf{w}(t)]$$

En notant les vecteurs en gras selon la convention usuelle, $\mathbf{x}$ est le vecteur d'entrée, $\mathbf{w}$ le vecteur de poids, $y = \mathbf{w}.\mathbf{x}$ la sortie, et η le pas d'apprentissage (typiquement choisi entre 0,1 et 0,01).
Le terme $y(t)\mathbf{x}(t)$ est de type hebbien, maximisant la variance de sortie, et $y^2(t)\mathbf{w}(t)$ est un facteur de normalisation, conçu pour assurer $||\mathbf{w}|| = 1$.

Cette procédure d'apprentissage a pour effet d'adapter les poids aux corrélations entre entrées et sortie, et du même coup aux corrélations entre les données. L'apprentissage a donc pour effet d'amener itérativement le vecteur de poids à représenter le premier vecteur propre de la covariance des données. La sortie du neurone, projection du vecteur d'entrée sur le vecteur de

poids normalisé, donne la nouvelle valeur de l'entrée le long de cet axe, c'est-à-dire la première composante principale.

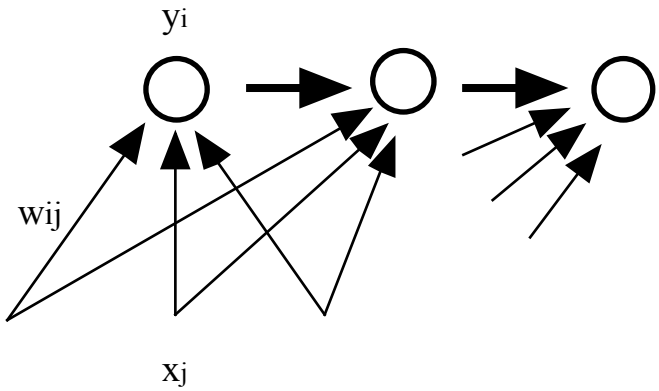

Fig. 2 : Schéma d'un réseau GHA.

Sanger a ensuite généralisé cette règle d'apprentissage à un réseau à une couche de neurones linéaires, de manière à pouvoir calculer les composantes principales successives (Sanger 1989). Chaque neurone reçoit le même vecteur d'entrée et chacun extrait une composante principale en ordre décroissant de variance (Fig. 2). La règle d'apprentissage est maintenant la suivante, en notation indicée :

$$w_{ij}(t+1) = w_{ij}(t) + \eta(t) \, [y_i(t)x'_j(t) - y_i^2(t)w_{ij}(t)]$$

chaque neurone successif utilisant une entrée modifiée $x'_j(t)$ calculée en soustrayant de l'entrée réelle l'effet des neurones précédents :

$$x'_j(t) = x_j(t) - \sum_{k<i} w_{kj}(t)y_k(t)$$

L'apprentissage pour GHA peut aussi s'exprimer de manière plus concise :

$$w_{ij}(t+1) = w_{ij}(t) + \eta(t) \, y_i(t) \, [x_j(t) - \sum_{k\leq i} w_{kj}(t)y_k(t)]$$

On démontre que cette règle généralisée fait converger les poids vers les vecteurs propres de la covariance, par ordre décroissant des valeurs propres. L'algorithme ne demande qu'un vecteur de donnée à la fois et évite donc le calcul de la matrice de covariance. On peut si on le souhaite le réaliser par des opérations purement locales en utilisant des connexions entre les neurones de sortie.
Le réseau peut fonctionner en parallèle (on ajuste alors tous les poids en même temps) mais il est plus simple et plus efficace de faire l'apprentissage de chaque neurone successivement. C'est la méthode que nous avons retenue.
Il existe aussi des variantes de cet algorithme, notamment le système APEX utilisant des connexions latérales pour exprimer l'influence d'un neurone sur les suivants (Diamantaras & Kung 1996)(Fiori 2000). Mais ces variantes sont plus complexes et améliorent peu les performances de l'algorithme de base, que nous avons retenu ici.

4 Traitement des documents

Nous avons utilisé un corpus de 90 pages Web environ, recueillies en vrac à partir des signets favoris (*bookmarks*) d'un utilisateur réel. Chaque page Web contenait une page ou deux de texte ordinaire en français, et l'ensemble touchait à une dizaine de thèmes différents (les centres d'intérêt de l'utilisateur). Notre but était d'effectuer une catégorisation non supervisée (*clustering*) de ces pages pour établir un profil de l'utilisateur dans le contexte du développement d'un moteur de recherche collaboratif, le système Human Links.
Nos choix ont été faits dans le cadre de ce système, et pour plus de détails sur la réalisation du travail, voir (Delichère 2000). Pour plus d'information sur le système Human Links développé par la start-up Amoweba pour accéder à une expertise distribuée sur Internet, on peut consulter le site Web :

http://www.amoweba.com

4.1 Prétraitements

La vectorisation de chaque page Web a été faite selon une procédure classique de manière à ne retenir que les mots ou termes les plus significatifs (Salton & McGill 1983). On a d'abord enlevé les mots grammaticaux (par simple consultation d'une liste), puis tronqué les mots lexicaux à leur 5 premières lettres pour réduire les variantes morphologique à une forme unique (technique simpliste mais en général efficace). On a compté ensuite les occurrences dans chaque document des termes ainsi retenus, et attribué à chaque terme une valeur pondérée selon une variante simplifiée de la mesure TFIDF pour favoriser les termes les plus discriminants entre documents :

fréquence dans le document / fréquence totale dans le corpus

On a gardé les termes dépassant un seuil minimal de fréquence et seulement s'ils figuraient dans au moins dans 10 à 20% des documents (pour assurer un niveau minimal de pertinence dans le corpus). Mais nous n'avons pas utilisé de seuil maximal de fréquence, qui ne semblait pas utile pour un petit corpus assez hétérogène. On a ainsi obtenu un vecteur numérique par document.
Ces prétraitements simples et rapides, ne faisant appel qu'à très peu de connaissances linguistiques, nous ont donné un espace de représentation de 600 traits lexicaux environ. C'est en fait assez bas pour ce domaine (où on atteint aisément des dimensions de 3000 à 5000 pour des corpus plus gros) mais cela reste une dimension élevée pour des traitements ultérieurs ou une analyse factorielle.

4.2 Réduction de dimension

Les vecteurs représentant les documents ont alors été donnés à un réseau GHA de 600 entrées et 20 neurones de sortie. Le nombre de sortie a été choisi empiriquement, en remarquant que des neurones supplémentaires ne donnaient plus que des sorties très faibles. Pour accélérer l'apprentissage, nous avons utilisé une version séquentielle de GHA, en adaptant les poids de chaque neurone successivement et non en parallèle.
Nous avons expérimenté avec différentes valeurs du pas d'apprentissage (de 0,1 à 0,01) mais en le gardant fixe durant un apprentissage. Nous avons systématiquement arrêté

l'apprentissage après 3 mn de calcul (sur une station de travail) car l'algorithme devait servir en ligne dans un système de recherche interactif.
Le réseau GHA a donc réduit à 20 les 600 dimensions d'origine, projetant les documents dans un espace beaucoup plus compact. Ces dimensions se sont révélées très significatives, ce qui nous a amenés à les qualifier de *concepts*. En effet ces nouvelles dimensions représentent essentiellement les corrélations entre termes dans un corpus donné, révélant de la sorte les thèmes principaux du corpus presque aussi bien qu'une classification des documents.
Voici les axes fournis par les neurones du réseau, correspondant donc aux composantes principales de plus forte variance. En représentant chaque axe par les termes contribuant le plus à la nouvelle dimension, on reconnaît sans mal des événements récents et les centres d'intérêt divers de l'utilisateur des signets de pages :

Axe 1 : israéliens palestiniens barak sharon ehud
Axe 2 : neurone couche entrée poids matrice
Axe 3 : kasskooye manager newsletter incubation business
Axe 4 : bové josé alegre brésil porto
Axe 5 : kasskooye manager incubation business strategy
Axe 6 : hoax virus hoaxbuster hybris pétition
Axe 7 : hockey tennis sélectionner football
Axe 8 : kasskooye bové newsletter manager josé
Axe 9 : zdnet napster peer fichiers freenet
Axe 10 : algorithme individus génétique génome mutation
Axe 11 : hoax pétition hoaxbuster poulet algorithme
Axe 12 : hoax pétition hoaxbuster rallye hockey
Axe 13 : hybris virus infection code matrice
Axe 14 : artificielle robotique individus infection algorithme
Axe 15 : peer serveur joueurs client club
Axe 16 : lire mondialisation alegre porto janvier
Axe 17 : joueurs artificielle club robotique capitaine
Axe 18 : championnat zdnet mercato club président
Axe 19 : rallye volley tennis sélectionner génome
Axe 20 : zdnet capitaine poulet lire neurone

On voit qu'un terme peut très bien contribuer à des axes différents, représentant les divers faisceaux de corrélation auxquels ce mot appartient dans le corpus. On remarque aussi que les premiers axes sont thématiquement très cohérents, alors que les suivants deviennent plus difficiles à interpréter (axes 12, 19 ou 20 par exemple). Il vaudrait peut-être mieux ne pas retenir les derniers axes.

4.3 Classification

Nous avons ensuite utilisé les vecteurs dans le nouvel espace (donnés par les valeurs de sortie du réseau) pour effectuer une classification non supervisée des documents (Bouroche & Saporta 1980)(Anderberg 1973). Nous avons essayé plusieurs variantes de l'algorithme bien connu des centres mobiles (*k-means*) en employant une distance euclidienne (pour ne pas avoir à normaliser les vecteurs). Mais nous aurions pu aussi bien utiliser des réseaux neuronaux compétitifs pour rester dans un cadre connexionniste (Memmi & Meunier 2000).
Les classes ainsi obtenues étaient tout à fait significatives, montrant clairement les thèmes principaux du corpus de pages. Les classes correspondaient bien à la dizaine de centres

d'intérêt qui avaient motivé la collection des signets de l'utilisateur humain : des articles d'acualité et de sport, la nouvelle économie, les réseaux de neurones, les rumeurs et virus informatiques, le *peer to peer*, les algorithmes génétiques...
Plus précisément, en demandant à l'algorithme des centres mobiles de retrouver 10 classes, c'est-à-dire le nombre de centres d'intérêt de l'utilisateur, l'algorithme retrouve bien en général (en faisant des tirages répétés avec des initialisations aléatoires variées) la classification d'origine des documents par l'utilisateur. Il faut avouer cependant que le problème de classement était assez facile car cet ensemble de documents est particulièrement bien structuré d'un point de vue thématique, ce qui voit déjà dans les résultats de l'analyse factorielle.
Surtout cette classification après réduction s'est révélée plus rapide et plus pertinente que dans l'espace d'origine. De fait, il aurait été impossible de classer correctement les vecteurs d'origine dans le temps limité imposé par la tâche interactive, puisque la classification devait se faire à la demande et en ligne.
On peut cependant noter ici le problème classique d'évaluation de la classification non supervisée. Contrairement à la classification supervisée, il n'y malheureusement pas de critères généraux objectifs de qualité des classes obtenues, et il faut en fin de compte faire appel au jugement humain dans le contexte d'une tâche particulière. Mais le corpus choisi se prêtait bien à une telle évaluation sémantique, puisque les centres d'intérêts de l'utilisateur étaient cohérents et assez bien équilibrés.
En bref, la réduction de dimension permet non seulement une amélioration de l'efficacité informatique, mais elle donne aussi des résultats intéressants d'un point de vue sémantique. Cela se comprend aisément si on observe que les nouvelles dimensions correspondent aux corrélations les plus importantes dans les données, et donc à des traits significatifs du corpus traité. L'ACP est ainsi un remarquable outil d'analyse thématique d'un corpus de documents.

5 Discussion technique

L'algorithme GHA a été généralement appliqué au traitement d'images, notamment pour une compression des images avec perte minimale d'information (codage optimal). A notre connaissance, c'est la première fois qu'on l'emploie pour le traitement de documents textuels. Nous avons pu vérifier que cette technique permet bien de traiter les vecteurs de grande dimension typiques des vecteurs de documents, alors que le traitement d'image se fait souvent avec des vecteurs beaucoup plus petits (de 64 valeurs par exemple en découpant l'image en blocs analysés l'un après l'autre).
Certes, la dimension des vecteurs que nous avons utilisés est encore relativement faible (600 environ) comparée aux vecteurs de plusieurs milliers d'éléments souvent employés en représentation de documents. Mais GHA a aussi été employé en traitement d'images pour des vecteurs de grande dimension (plusieurs milliers de traits), et nous sommes convaincus que de telles dimensions ne posent pas de problèmes majeurs. En effet le coût d'apprentissage et de fonctionnement est en première analyse simplement proportionnel à la dimension du vecteur d'entrée.
D'autre part, pour plus de clarté théorique, nous avons fait une description de l'ACP standard portant sur des données centrées (de moyenne nulle). Mais diverses variantes sont possibles, et nous avons utilisé en fait dans ce travail des données non centrées afin d'accélérer les calculs. Dans ce cas, on ne peut plus parler de variance en toute rigueur, et les nouveaux axes passent par l'origine et non par le centre de gravité des données. Mais l'algorithme neuronal effectue toujours une réduction de dimension, et on a vu que les nouvelles dimensions obtenues étaient tout à fait significatives.

L'Algorithme Hebbien Généralisé permet donc bien d'extraire les composantes principales des vecteurs de très grande dimension caractéristiques des documents textuels, de manière à réduire la dimension de la représentation.
De plus GHA ne calcule que les premières composantes principales (les plus importantes), ce qui peut représenter une économie de calcul importante. La nature adaptative de l'algorithme permet aussi de se contenter d'une estimation approchée des résultats (que l'on pourra affiner plus tard au besoin), alors que les logiciels standard d'ACP calculent tous les vecteurs propres avec la précision maximale.
Cependant GHA présente aussi certains inconvénients. Comme souvent avec les techniques neuronales, le pas d'apprentissage doit être estimé empiriquement, et les temps d'apprentissage peuvent être longs si on cherche une bonne précision, surtout pour les composantes suivantes de moindre variance. Et cet algorithme fait que les erreurs de calcul s'accumulent d'un neurone au suivant, ce qui diminue inévitablement la précision des composantes successives. Il est donc important en pratique de se limiter aux premières composantes principales.
Il faut rappeler que l'algorithme ne donne pas directement la part de variance correspondant à chaque composante principale. On peut alors calculer facilement la variance de sortie de chaque neurone sur un échantillon du corpus de manière à trouver le nombre de composantes réellement utiles (on arrêtera l'apprentissage de nouveaux neurones lorsque leur variance devient insuffisante).
En somme, GHA ne se justifie que lorsqu'on peut se contenter des premières composantes principales. Mais c'est souvent le cas, puisqu'elles représentent la plus grosse part de l'information contenue dans les données.
Enfin, tout comme l'ACP, GHA est une méthode purement linéaire qui ne peut capturer que des corrélations linéaires entre les données (cela revient en fait à n'utiliser que la covariance). C'est probablement suffisant pour traiter des documents, mais si on veut dépasser cette limitation, il faudrait envisager d'autres méthodes comme l'Analyse en Composantes Indépendantes (voir Hérault & Jutten 1994) ou les Cartes de Kohonen (Ritter & Kohonen 1989)(Kohonen 1998).

6 Conclusion

Les vecteurs de grande dimension caractéristiques de la représentation des documents textuels sont très redondants et inutilement coûteux. Pour représenter l'information pertinente de manière plus compacte, nous avons utilisé une technique neuronale, l'Algorithme Hebbien Généralisé (GHA), qui permet d'extraire automatiquement les premières composantes principales correspondant à la majeure partie de l'information contenue dans les données. Cela permet une réduction de dimension à la fois techniquement économique et significative pour un utilisateur humain.
L'algorithme calcule de nouvelles dimensions sans employer la matrice de covariance, ce qui permet d'envisager des vecteurs de très grande dimension qu'une ACP classique ne pourrait pas traiter. Nous avons trouvé que cette technique connexionniste donnait de bons résultats pour une tâche de classification dans le cadre d'un système interactif réel. L'approche mérite donc d'être considérée pour réduire la dimension des données textuelles avant les traitements ultérieurs.
Cette réalisation neuronale de l'ACP présente aussi l'intérêt de faire émerger automatiquement les thèmes principaux d'un corpus de documents. C'est là un outil intéressant, efficace et relativement facile à mettre en oeuvre pour l'analyse thématique d'un corpus de textes. Nous pensons qu'il y là matière à des travaux ultérieurs.

Références

Anderberg M.R. (1973) *Cluster Analysis for Applications*, Academic Press.

Bouroche J.M. & Saporta G. (1980) *L'Analyse des Données*, Que sais-je, PUF.

Deerwester S., Dumais S.T., Furnas G.W., Landauer T.K. & Hashman R. (1990) Indexing by latent semantic analysis, *Journal of the American Society for Information Science* 41 (6), p. 391-407.

Delichère M. (2001) Etat de l'art et implémentation d'algorithmes de recherche et de classification automatique de documents sur Internet - Intégration à l'outil Human Links, rapport de stage de fin d'étude, EPITA, Paris.

Diamantaras K.I. & Kung S.Y. (1996) *Principal Component Neural Networks: Theory and Applications*, John Wiley & Sons.

Fiori S. (2000) An experimental comparison of three PCA neural networks, *Neural Processing Letters* 11 (3), p. 209-218.

Hertz J., Krogh A. & Palmer R.G. (1991) *Introduction to the Theory of Neural Computation*, Addison Wesley.

Hérault J. & Jutten C. (1994) *Réseaux Neuronaux et Traitement du Signal*, Hermès.

Jolliffe I.T. (1986) *Principal Component Analysis*, Springer Verlag.

Kohonen T. (1998) Self-organization of very large document collections: state of the art, *Proc. of ICANN'98*, London.

Lebart L., Morineau A. & Piron M. (2000) *Statistique Exploratoire Multidimensionnelle*, Dunod.

Lebart L. & Salem A. (1994) *Statistique Textuelle*, Dunod.

Manning C.D. & Schütze H. (1999) *Foundations of Statistical Natural Language Processing*, MIT Press.

Memmi D. & Meunier J.G. (2000) Using competitive networks for text mining, *Proc. of Neural Computation 2000*, Berlin.

Oja E. (1982) A simplified neuron model as a principal component analyzer, *Journal of Mathematics and Biology* 15, p. 267-273.

Ritter H. & Kohonen T. (1989) Self-organizing semantic maps, *Biological Cybernetics* 61 (4), p. 241-254.

Salton G. & McGill M. (1983) *Introduction to Modern Information Retrieval*, McGraw-Hill.

Sanger T.D. (1989) Optimal unsupervised learning in a single-layer linear feedforward neural network, *Neural Networks* 2 (6), p. 459-473.