الجمهورية الجزائرية الديمقراطية الشعبية
وزارة التعليم العالي و البحث العلمي
جامعة وهران للعلوم و التكنولوجيا محمد بوضياف

THÈSE

En vue de l'obtention du

Diplôme de Doctorat en Sciences

Présenté par : BENDAHMANE Abderrahmane

Intitulé

Apprentissage génératif connexionniste pour la classification des signaux complexes sous contraintes

Faculté ***: Mathématiques et informatique***

Département ***: Informatique***

Spécialité ***: Informatique***

Option ***: Intelligence artificielle***

Devant le Jury Composé de :

Membres de Jury	*Grade*	*Qualité*	*Domiciliation*
BENYETTOU Mohamed	*Professeur*	*Président*	*USTO-MB*
BENYETTOU Abdelkader	*Professeur*	*Encadrant*	*USTO-MB*
FIZAZI Hadria	*Professeur*		*USTO-MB*
BENYAMINA Abou El Hassan	*Professeur*	*Examinateurs*	*Univ. Oran 1*
TEMMAR Abdelkader	*Professeur*		*INTTIC*
HAFFAF Hafid	*Professeur*		*Univ. Oran 1*

Année Universitaire : 2017/2018

Liste des travaux scientifiques

Publications

Bendahmane, A., & Benyettou, A. (2016). Learning to Generate Optimized Term Weighting for Web Documents Classification - A Parallel Mimetic Approach Based on Support Vector Machines. *International Review on Computers and Software*, 8 (6), pp. 1374-1381. *DOI: 10.15866/irecos.v11i12.10964*

Silarbi, S., Bendahmane, A., & Benyettou, A. (2014). Adaptive Network Based Fuzzy Inference System for Speech Recognition Through Subtractive Clustering. *International Journal of Artificial Intelligence & Applications*, 5(6), 43. *DOI: 10.5121/ijaia.2014.5604*

Communications

Bendahmane, A., Benyettou, A., & Benabdesslem, K. (2015). Une approche évolutionnaire multi-contraintes pour l'optimisation de la génération automatique d'emplois du temps universitaires. *12ième édition du Colloque sur l'Optimisation et les Systèmes d'Information*, 1 au 3 juin 2015 – Oran, Algérie.

Benyettou, A., Bendahmane, A., & Benabdesslem, K. (2015). Sélection de variables pour l'amélioration de l'accès à l'information sur le Web. *12ième édition du Colloque sur l'Optimisation et les Systèmes d'Information*, 1 au 3 juin 2015 – Oran, Algérie.

Farhi, N., Mediouni, Z., & Bendahmane, A. (2013). Compression vidéo par réseaux de neurones. *Séminaire National sur les Technologies de l'information et de la communication*, USTO-MB Oran 2013.

Mokrane, I., Ouyahia, W., & Bendahmane, A. (2013). La réalité augmentée dans l'e-commerce. *Séminaire National sur les Technologies de l'information et de la communication*, USTO-MB Oran 2013.

Sellam, M., Sari, O., & Bendahmane, A. (2013). Suivi des objets en mouvement à partir d'une caméra mobile. *Séminaire National sur les Technologies de l'information et de la communication*, USTO-MB Oran 2013.

Bendahmane, A., Bendahmane, A., & Benyettou, A. (2013). Une approche évolutionnaire pour l'exploration collaborative optimisée d'un environnement Inconnu. *10ième édition du Colloque sur l'Optimisation et les Systèmes d'Information*, 09 au 11 juin 2013 – Alger, Algérie.

Houalef, S., Bendahmane, A., & Benyettou, A. (2012). Système de reconnaissance biométrique multimodale basé sur la fusion : empreinte digitale, visage, géométrie de la main. *11ème Colloque Africain sur la Recherche en Informatique en Mathématiques Appliquées*, du 13-16 Octobre 2012, Alger - Algérie.

Silarbi, S., Bendahmane, A., & Benyettou, A. (2012). Reconnaissance des individus basée sur la géométrie de la main. *International Conference on Indistrual Engineering & Manufacturing*, May 06-07 2012 à l'Université Hadj Lakhdar – Batna, Algérie.

Remerciements

Il est des remerciements qu'on prend plaisir à formuler, ceux que j'ai à adresser pour cette thèse font partie de cette catégorie. Une thèse ne peut se faire sans le soutien de nombreuses personnes envers qui je suis très reconnaissant et qui avaient ou ont pris une place importante pour moi. Je crains d'oublier certains ou certaines, mais qu'elles soient assurées que même si leurs noms ne figurent pas sur le papier, ils resteront inscrits dans ma mémoire.

Merci tout particulièrement :

– à mon défunt père et à ma mère, qui m'ont élevé dans un milieu favorable à la science, pour leur patiente, pour leur rigueur, pour leur gentillesse et leur générosité, à mon frère Lotfi, mes sœurs Chahinez et Amina, ma femme Kawter Nour El Houda et ma famille pour leur soutien et leurs encouragements, sans qui cette thèse n'aurait pas aboutie.

– à Mr Benyettou Abdelkader pour m'avoir proposé d'être mon directeur de thèse pour finalement devenir plus qu'un directeur de thèse.

– aux examinateurs d'avoir pris sur leur temps pour lire ce mémoire, et aux membres du jury pour l'intérêt dont ils font preuve à mon égard en assistant à ma soutenance.

– aux membres du laboratoire SIMPA et collègues à USTO-MB pour leur accueil chaleureux et leur soutien, pour ces années de cohabitation parsemées de fous rires et de ces discussions sur l'IA et les nouvelles technologies et techniques émergentes.

Table des matières

Liste des abréviations

AllNet	*Concaténation GoogleNet+Caffe+AlexNet+ResNet*
BIN	*Binary*
CD_{max}	*Maximum Class Density*
CGEDA	*Continuous Gaussian EDA*
CHI	*Chi-square* χ^2
CNN	*Convolutional Neural Network*
EA	*Evolutionary Algorithm*
EDA	*Estimation Distribution Algorithm*
HSL	*Hue Saturation Lightness*
ICF	*Inverse Class Frequency*
IDF	*Inverse Document Frequency*
IG	*Information Gain*
ITF	*Inverse Term Frequency*
K-NN	*k Nearest Neighbors*
LBP	*Local Binary Pattern*
logTF	*log(TF+1)*
$logTF^{IDF}$	*logTF×IDF*
$MacroF_1$	*Macro-average* F_1
MI	*Mutual Information*
MLP	*Multi-Layer Perceptron*
MSVM	*Multi-class SVM*
N-fold	*N-fold cross validation*
R8	*Reuters-21578 réduite à 8 categories*
ReLU	*Rectified Linear Units*
RF	*Relevance Frequency*
RFE	*Recurrent Feature Elimination*
SVM	*Support Vector Machines*
TF	*Term Frequency*
TF^{IDF}	*TF×IDF*
VSM	*Vector Space Model*
YCbCr	*L'espace des couleurs luminance chrominance*
WLLR	*Weighted Log Likelihood Ratio*

Liste des figures

Liste des tableaux

Introduction générale

Le succès des algorithmes d'apprentissage dépend en général de la représentation des données, ceci est principalement dû au fait que certaines représentations peuvent mieux expliquer la structure cachée des données, la découverte de cette structure est difficile, surtout si les données sont de nature très complexe. L'utilisation de connaissances ou d'hypothèses sur le domaine peut aider à élaborer des représentations très pertinentes, cependant elle nécessite un effort considérable d'experts humains. La réutilisabilité de ces représentations d'un domaine à un autre est souvent inadéquate et peut faire chuter considérablement les performances. Actuellement, la majorité de l'effort fourni par les chercheurs en apprentissage machine concerne le prétraitement et la transformation des données, cet effort peut s'avérer très couteux en temps et en argent à long terme et souligne la faiblesse des algorithmes actuels à établir de façon autonome des représentations discriminantes à partir des données sans une intervention humaine.

Dans cette thèse on part du principe qu'un système réellement intelligent devrait être capable d'apprendre et de générer par lui-même la représentation la plus adéquate pour le traitement qu'il veut faire sur les données. Afin de simplifier et vulgariser l'utilisation des techniques de l'apprentissage automatique, il serait préférable de réduire leur dépendance à l'expert humain, afin de permettre la construction rapide d'applications intelligentes en un moindre coût. Un système intelligent devrait être fondamentalement capable de comprendre le monde qu'il l'entour, ceci peut être atteint si le système est capable d'identifier à l'intérieur des données brutes issues de ses capteurs de bas niveau, des facteurs cachés capables d'expliquer la constitution des objets du monde observé. Actuellement, les recherches dans le domaine de l'apprentissage profond (*Deep Learning*) sont de plus en plus sollicitées via des techniques et des architectures connexionnistes de plus en plus sophistiquées, permettant de projeter les données dans plusieurs niveaux d'abstraction. Ces architectures, principalement neuronales ont prouvées leur efficacité dans de nombreux domaines en reconnaissance des formes, en extraction des connaissances et en compression des données.

Dans ce travail, on s'intéresse aux données complexes, plus particulièrement à la classification de documents issus du web et rédigés en langage naturel ainsi qu'à la classification des images aériennes. Le traitement des données issues du langage naturel représente un parfait défi à relever vu sa complexité et sa nature très ambiguë. Vu l'énorme quantité de documents disponibles sur le web il est crucial d'élaborer des outils de recherche et de classification automatisés afin d'organiser et faciliter l'accès à ces documents, il est aussi nécessaire d'élaborer des techniques permettant l'apprentissage et la génération automatique de représentations optimisées pour ces derniers. Bien qu'un document web peut contenir plusieurs types de données (textes, images, vidéos …etc.), dans cette thèse on s'intéresse principalement aux données textuelles. D'un autre côté, la classification des images aériennes représente un second défi qu'on a souhaité relever dans cette thèse, d'une nature complexe, les images aériennes sont composées de plusieurs types de textures disposées spatialement de façon anarchique avec une grande variabilité au sein d'une même catégorie : transformations

géométriques, luminosité, bruit etc. Cette variabilité, déroute les meilleurs classifieurs en apprentissage machine, n'arrivant pas à découvrir une représentation pertinente et discriminante des images, ces classifieurs ont tendance à trop se focaliser sur les données d'apprentissage et perdent leur faculté de généralisation.

Dans le premier chapitre, on aborde de façon générale la représentation des données, avant de nous focaliser sur la représentation des données textuelles et les techniques qui permettent d'optimiser cette représentation via la sélection et la pondération des attributs avec un état de l'art récent dans le domaine. On aborde ensuite, la représentation des images aérienne, en mettant l'accent sur les techniques classiques de représentation des textures, suivies d'un état de l'art récent dans ce domaine.

Le deuxième chapitre, introduit le domaine des techniques connexionnistes de façon générale, avant de se focaliser par la suite sur le *Deep Learning*, et plus précisément sur les réseaux de neurones à convolution et les nombreux avantages qu'offre cette architecture. On pose par la suite l'une des principales problématiques liée aux approches connexionnistes qui est le surapprentissage. L'apprentissage sous contraintes d'optimalités via les *SVM* est une solution très prisée pour réduire les risques dus à cette problématique. Dans ce chapitre, on aborde de façon générale les fondements des *SVM* avant de mettre en évidence le lien qui les relie du point de vue de leur architecture aux modèles connexionnistes.

Trouver une représentation optimisée qui maximise les performances d'un classifieur peut être vu comme un problème d'optimisation. Dans le domaine de l'optimisation, les approches évolutionnaires sont des techniques universelles permettant de trouver de meilleurs minimas que les algorithmes de recherche locale et ceci sans connaissances préalables sur le domaine. Cependant, ces approches sont très couteuses en temps d'exécution, plusieurs solutions sont envisagées et discutées en chapitre 3 via l'hybridation avec des approches locales (le paradigme mimétique) et la parallélisation du processus. La suite du chapitre explique l'approche proposée qui est basée sur une technique à estimation de distribution *CGEDA*, avec des détails sur la formulation évolutionnaire du problème, la fonction de fitness utilisée ainsi que l'hybridation avec la sélection d'attributs par *MSVM-RFE*. La parallélisation du processus sur un modèle d'îlots est décrite en fin de chapitre, où on propose une architecture composée de trois types d'îlots différents afin de booster la vitesse de l'approche proposée au maximum sans perte de performances.

En chapitre 4, on montre l'efficacité de l'approche proposée via plusieurs expériences effectuées sur plusieurs benchmarks en classification de documents web et en classification des images aériennes. Les expériences ont pour but de : déterminer la meilleure représentation initiales parmi les représentations classiques, de montrer l'efficacité de la sélection par *MSVM-RFE* en comparaison aux approches classiques, de montrer l'efficacité de l'hybridation proposée en terme de vitesse de convergence et de taux de réduction, de montrer le gain en temps d'exécution du modèle parallèle proposé sur un matériel dédié et l'impact des différentes politiques de migration sur les performances.

CHAPITRE 1

Représentation des données complexes

I.1 Introduction

Les performances de l'apprentissage machine dépendent largement du choix de la représentation des données. Actuellement, la plupart des algorithmes d'apprentissage nécessitent des phases de prétraitement et de transformation afin d'être applicable et efficace sur les données. Cette transformation nécessite un travail fastidieux d'experts humains et beaucoup de connaissances dans le domaine traité, cela dénote aussi la faiblesse de ces algorithmes à extraire automatiquement des représentations discriminantes à partir des données sans une intervention humaine. Un système artificiel réellement intelligent devrait être plus autonome et moins dépendant des interventions humaines, et cela ne peut être atteint que si le système est capable de découvrir la structure cachée des données brutes et l'expliquer.

Dans ce chapitre, on évoque la possibilité d'optimiser la représentation des données complexes pour la classification via la sélection et la pondération des attributs, avec comme exemples, les données textuelles et les images aériennes. La représentation classique des données textuelles consiste en des matrices creuses d'une très grande dimensionnalité comportant des centaines de milliers d'attributs. Transformer cette représentation complexe en une représentation plus compacte et plus discriminante est l'un des challenges visés par cette thèse. D'un autre côté, la classification des images aériennes est une tâche très complexe pour un classifieur automatique, vu l'immense variabilité de ces images au sein d'une même catégorie et une disposition spatiale anarchique des éléments qui les composent. Sans une représentation adéquate, cette variabilité a tendance à réduire les meilleurs classifieurs en surapprentissage.

I.2 La qualité d'une représentation

En reconnaissance des formes, chaque forme peut être décrite avec plusieurs types de caractéristiques, ces caractéristiques ne sont pas toutes informative sur la nature de la forme étudiée, et certaines caractéristiques sont plus expressives que d'autres. La représentation d'une forme comporte l'ensemble des caractéristiques avec lesquelles on décrit cette dernière, ces caractéristiques peuvent être des données brutes issues des capteurs (caméra, microphone…), ou des données prétraitées et transformées de plus haut niveau.

Certaines représentations sont meilleure que d'autres, en classification, la mesure de la qualité d'une représentation ce fait selon plusieurs critères :

1. La pertinence :
    - Est-ce que les caractéristiques sont informatives sur la nature de la forme ?
    - Est-ce que toutes les caractéristiques sont importantes ?
    - Est-ce qu'il y a de la redondance dans les caractéristiques ?

    Exemple : dire qu'on a un smartphone avec un écran tactile est une information non utile vu que tous les smartphones sont dotés d'un écran tactile.

2. La discrimination :
    - Est-ce que les caractéristiques sont en nombre suffisant pour pouvoir différencier entre les catégories des formes ?
    - Est-ce qu'il n'y a pas d'ambiguïté dans la description des formes ?

   Exemple : dire qu'on a un fruit de forme ovale et de couleur jaune n'est pas suffisant pour deviner sa nature, ça peut être un citron, un pamplemousse, un melon …

3. La robustesse :
    - Est-ce que les caractéristiques sont résistantes au bruit qui risque d'entacher la forme ?

   Exemple : Est-ce qu'on peut toujours reconnaitre une personne filmée par une caméra de surveillance de très basse résolution, ou comprendre une conversation Skype qui passe via une connexion internet dégradée ou de très bas débit.

On note que même si la représentation des données est de bonne qualité, il est souvent difficile d'extraire des caractéristiques correctes à partir des données brutes comme une image ou de la parole (voir Figure 1).

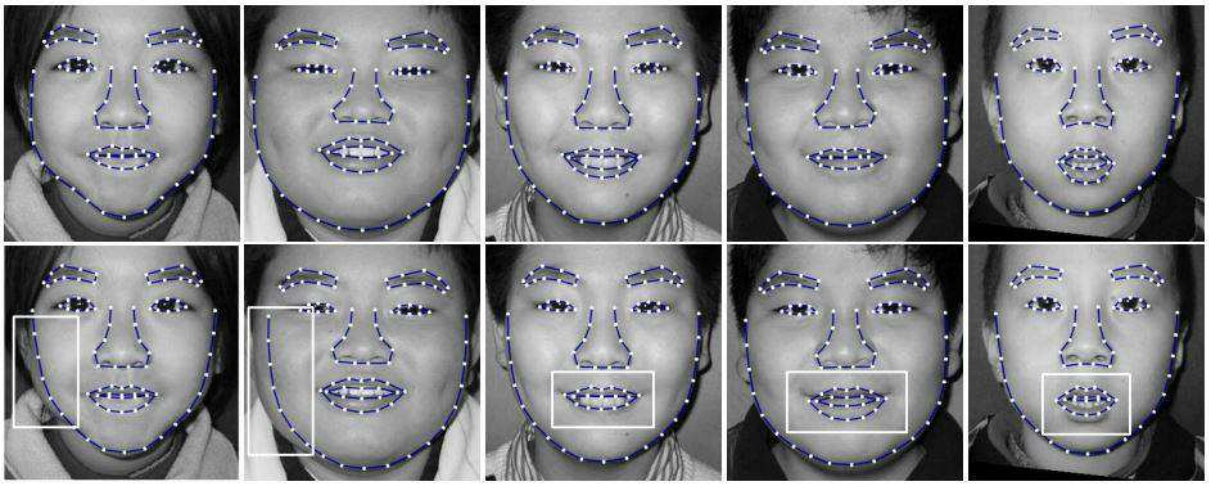


Figure 1 Exemple montrant la difficulté d'extraire des caractéristiques correctes à partir d'une image [1]

I.3 Les types de représentations des données

En reconnaissance des formes, il est d'usage de représenter une forme par un vecteur numérique multidimensionnel de caractéristiques ou d'attributs. Cette représentation simplifie la réalisation d'algorithmes grâce à deux caractéristiques :

[1] http://mmlab.ie.cuhk.edu.hk/project_face_alignment.html

1. Les caractéristiques sont quantifiées par de simples valeurs numériques : ce qui facile l'élaboration de modèles mathématiques et statistiques puissants afin de traiter les formes à analyser.
2. La dimension du vecteur de représentation est fixe : ce qui simplifie encore plus l'élaboration de modèles statiques sans mémoire et où la structure des données importe peu ou pas du tout.

Dans ce type de représentation, la fonction de décision est une simple fonction mathématique multi-variables qui calcule la sortie du modèle en fonction du vecteur d'entrée, et où les variables sont supposées être indépendantes les unes des autres (équation 1).

$$reponse = f(x_1, x_2, ..., x_n) \qquad 1$$

Cependant, il subsiste plusieurs domaines où cette représentation est insuffisante pour capturer la structure ou la dynamique des formes. Par exemple, la composante spatiale des éléments d'une image, la composante temporelle d'un signal de parole, les nœuds d'un fichier XML ou bien l'ordonnancement des mots dans un paragraphe etc. Dans ce sens, des structures de représentations plus complexes peuvent être envisagées telles que les séquences, les arbres ou les graphes.

Certains types de données s'organisent naturellement sous forme de séquences de vecteurs caractéristiques, surtout celles qui dépendent d'un facteur temporel ou bien celles dont l'ordre de ses éléments est porteur d'informations pertinentes. Ignorer cette composante temporelle peut dégrader fortement les performances d'un classifieur. On peut citer plusieurs exemples qui rentrent dans ce domaine appelé communément « apprentissage des séquences (Sun 2001)» :

- La reconnaissance de la parole où un mot est composé de plusieurs phonèmes/bi-phones/tri-phones dont l'ordonnancement et la durée importent beaucoup pour la reconnaissance.
- En robotique où les données sont reçues à partir de capteurs à chaque intervalle de temps sous forme de séquences.
- En prévision météo, où les prévisions d'une période de la journée dépendent fortement des périodes précédentes.
- L'ordonnancement des mots dans une phrase ou des idées dans un paragraphe.
- La prédiction des séries temporelles.
- Le séquencement ADN etc.

Il existe plusieurs outils en apprentissage machine qui permettent de traiter directement des données séquentielles, sans prétraitement préalable tel que la distance *DTW* (*Dynamic Time*

Warping) ou les *HMM* (*Hidden Markov Model*) qui ont largement fait leurs preuves en reconnaissance de la parole surtout avec l'arrivé des *HMM* hiérarchiques (Fine et al. 1998) qui sont implémentés dans des plateformes très connues dans le domaine tel que *HTK* et *SPHINX*. D'autres outils puissants en apprentissage machine, tel que les Séparateurs à Vaste Marge (*SVM*) ou les réseaux de neurones artificiels à l'origine conçus pour traiter de simples données vectorielles, ont été adaptés pour traiter des données séquentielles via l'ajout de noyaux temporelles ou de mécanismes de mémorisation à travers des filtres ou de connexions récurrentes (Anderson et al. 1999), (Zhang et al. 2016).

```
<ROOT>
  <TITLE>T1, T2 </TITLE>
  <H1>T3, T2 </H1>
  <P>
    <I>T4</I>
    <I>
      <B>T2</B>
       T5
      <B>T3, T4</B>
    </I>
  </P>
</ROOT>
```

Figure 2 Représentation brute d'un fichier XML

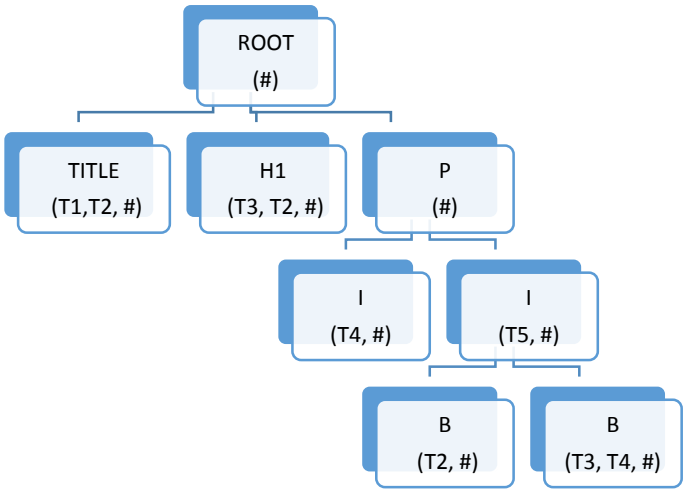


Figure 3 Représentation structurée d'un fichier XML

D'autres types de données contiennent des structures beaucoup plus complexes et peuvent être représentés par des arbres ou des graphes (Hagenbuchner et al. 2003). Une représentation

arborescente est souvent utilisée afin de représenter une relation hiérarchique entre les nœuds de l'arbre, l'un des exemples des plus répondus est le format *XML* qui est très utilisé pour le stockage et le transfert des données et en création d'interfaces graphiques (voir Figure 2 et Figure 3). Les graphes permettent quant à eux de représenter des relations d'adjacence ou de voisinage entre les nœuds, on peut citer comme exemple une image qui peut être décomposée en plusieurs régions ou composantes interconnectées (Figure 4). Par exemple, un visage est constitué des yeux, d'un nez et d'une bouche disposés spatialement avec une relation de voisinage bien spécifique.

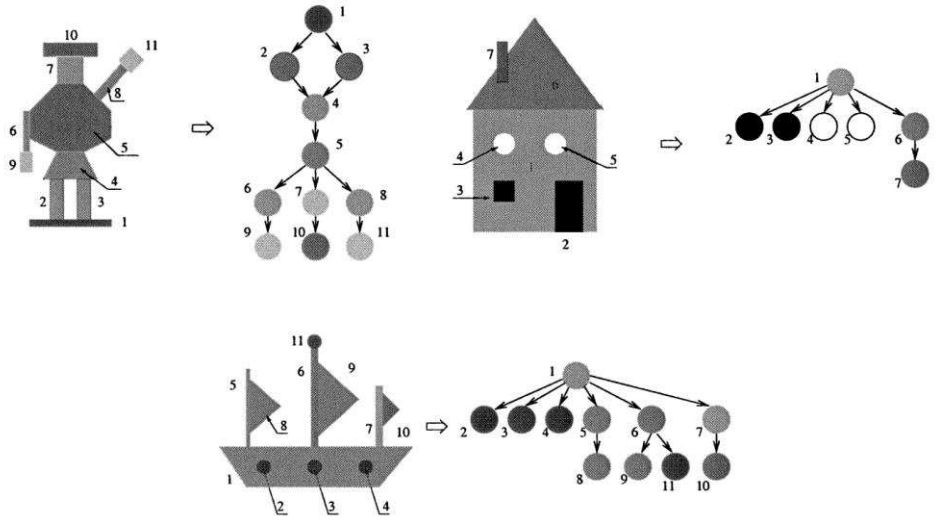


Figure 4 Exemple d'une représentation structurée pour des images (Hagenbuchner et al. 2003)

Dans le domaine de l'apprentissage machine, certaine représentations permettent une meilleure extraction d'informations en révélant des structures cachées (ou latentes) à l'intérieur des données. Cette caractéristique très prisée est présente dans les *HMM* qui permettent de modéliser des transitions cachées dans une séquence, en apprentissage profond (Deep Learning) qui projette les donner dans un espace hiérarchique de variables latentes, permettant du coup une meilleure explication de la structuration des données. Le fait d'identifier ces structures cachées, relève du domaine de l'apprentissage des représentations (Bengio et al. 2013), un axe de recherche qui vise à optimiser la représentation des données afin de simplifier l'extraction des caractéristiques de ces dernières. Un article par exemple est composé de mots, chaque groupe de mots peut constituer une idée secondaire, un groupe d'idées secondaires peut constituer une idée principale dont le groupement permet d'expliquer la thématique de l'article. La Figure 5 illustre un exemple, où il est plus facile d'expliquer la thématique générale de la phrase « J'aime le football, la natation, la guitare et les échecs » qui est « mes loisirs » en projetant des groupes de mots dans un espace de variables intermédiaires (dans cet exemple : sport, musique et jeux) que d'essayer de la déterminer directement à partir des mots. Bien sûr il est possible de faire des projections à plusieurs niveaux d'abstractions afin d'obtenir des représentations plus pertinentes.

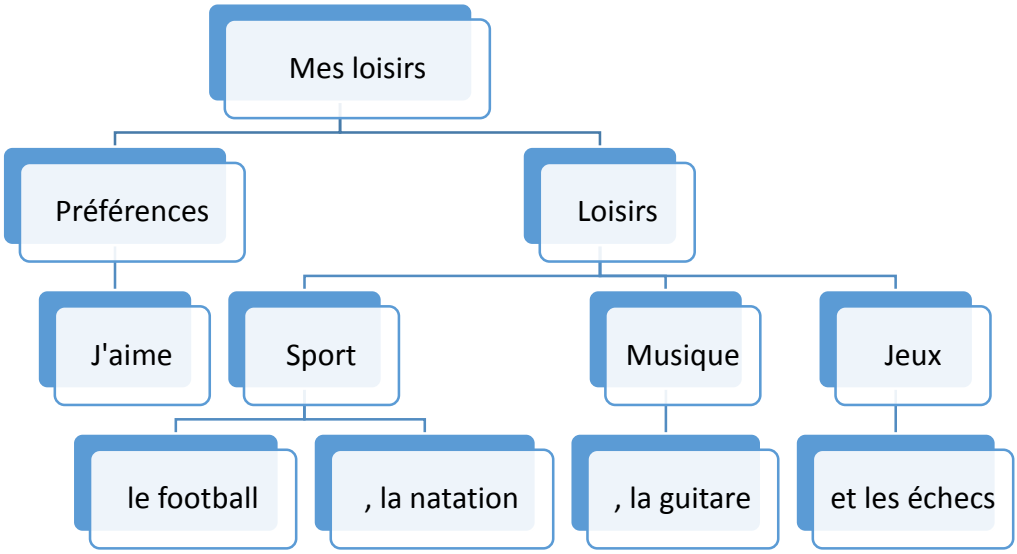


Figure 5 Exemple d'une représentation arborescente d'une phrase

I.4 Représentation vectorielle des données textuelles

Le traitement automatique du langage naturel est une composante principale de tout système intelligent qui vise à simplifier la communication entre l'homme et la machine en permettant à la machine de comprendre le langage des hommes. La compréhension d'un message écrit parait rudimentaire à toute personne, cependant, essayer de reproduire et d'automatiser ce traitement sur une machine est une tâche très difficile spécialement dû aux ambiguïtés qu'on rencontre à tous les niveaux : lexical, syntaxique, sémantique …etc.

Une des représentations des plus utilisées en traitement des données textuelles est le modèle vectoriel (*VSM* : *Vector Space Model*) où chaque énoncé, article ou message sont représentés en sac de mots (*Bag of words*). Ce modèle vise à transformer les données textuelles en vecteurs numériques afin d'utiliser le techniques habituelles en apprentissage machine pour la classification et la recherche.

A la base, cette représentation suppose que le mot est l'unité d'information principale, et ne prend pas en considération la notion d'ordre des mots qui peut parfois être très importante en recherche d'informations ou en classification. Plusieurs études ont montré qu'utiliser le mot (ou terme) comme unité de base assure de bons résultats en classification. Il subsiste plusieurs variantes pour représenter un terme dans un modèle vectoriel (fréquence/occurrence, avec ou sans pondération), la matrice qui en résulte est appelée matrice termes fréquences.

La pondération des termes peut être vue comme une généralisation de la sélection d'attributs, elle peut être réalisée en mode supervisé ou non supervisé. En traitement du langage naturel, les termes n'ont pas la même expressivité, il est souvent plus efficace de donner un poids plus fort aux termes importants. Cependant, le problème se pose dans la définition de critères afin de mesurer l'expressivité de chaque terme. Vu son extrême importance dans le domaine du traitement du langage naturel, la pondération des termes est devenue un axe de recherche très actif et plus particulièrement en recherche et extraction d'informations.

La pondération non supervisée n'utilise pas les labels des documents pour la mesure de l'importance des termes, plusieurs approches ont été définies dans la littérature, les plus communes sont reportées en Tableau 1.

Tableau 1 Techniques communes de pondération non supervisée

Méthode	*Formule de pondération*	*Informations*
TF	$TF(t_i, d)$	D: ensemble des documents t_i : terme i d: un document $TF(t_i, d)$: Nombre d'occurrences du terme t_i dans le document d
Binaire	$\begin{cases} 1 & si\ t_i \in d \\ 0 & sinon \end{cases}$	
IDF	$IDF(t_i, D) = \frac{\lvert D \rvert}{\lvert \{d \lvert t_i \in d, d \in D\} \rvert}$	
TF-IDF	$TF(t_i, d)\ IDF(t_i, D)$	
Log de TF-IDF	$log(TF(t_i, d) + 1)\ IDF(t_i, D)$	
ITF	$1 - \frac{1}{TF(\mathrm{t_i}, \mathrm{d}) + 1}$	

La pondération non supervisée est basée sur certaines idées de base (Debole & Sebastiani 2003), (Salton & Buckley 1988):

- Un terme apparaissant dans peu de documents d'un corpus peut être spécifique à un document ou un groupe de documents : principe de la pondération *IDF* (*Inverse Document Frequency*).
- Les termes fréquents dans un document peuvent être caractéristiques de ce document : *TF* (*Term Frequency*), *ITF* (*Inverse Term Frequency*).
- Seule l'existence du terme dans le document est importante : pondération binaire.

Des tests statistiques proposés par Song et ses collègues (Song et al. 2005) sur deux benchmarks *Reuters-21578* et *20Newsgroups* montrent que l'utilisation du logarithme des vecteurs de fréquence normalisés et pondérés par l'approche *IDF* permet d'améliorer les performances de la classification. La normalisation des vecteurs de fréquence (*TF*) supprime l'information sur la longueur du document qui n'est pas nécessairement significative en classification des documents.

La pondération *ITF* essaie de réduire l'influence des termes trop fréquents, l'idée est qu'à partir d'une certaine fréquence, les termes ont approximativement les mêmes poids. La pondération *ITF* a montrée des résultats prometteurs qui sont légèrement meilleurs que la

pondération *TF-IDF* dans les travaux de Hassan et ses collègues (Hassan et al. 2007). Dans d'autres travaux, une pondération binaire semble suffisante pour atteindre de bonnes performances (Forman 2004).

I.4.1 Problèmes liés à la représentation vectorielle

Le principal inconvénient avec la représentation vectorielle est la dimensionnalité de l'espace des attributs qui augmente considérablement avec le nombre de documents disponibles, et qui atteint facilement les centaines de milliers de termes. Ces termes ne sont pas forcément tous pertinents pour la classification.

Un deuxième problème lié à cette représentation est la nature creuse des matrices de fréquences. Chaque document ne contient qu'une centaine de termes parmi les centaines de milliers de termes recensés sur l'ensemble des documents, plus de 99% de la matrice termes/fréquences est nulle, cette caractéristique peut causer des dégradations de performances sur certains classifieurs.

Un troisième problème lié à cette représentation est l'absence de modélisation de liens entre les termes, bien que des termes différents peuvent être très proche sémantiquement (la notion de synonymes), ou bien, un même terme peut avoir des significations différentes (la notion d'homonymes).

Certaines étapes en prétraitement permettent de réduire l'espace des attributs telles que la racinisation (*stemming*) et le filtrage des mots vides (*stop words*). D'après les expériences de Song et ses collègues (Song et al. 2005), le filtrage des mots vides et la racinisation peuvent réduire le nombre de termes sans affecter les performances de la classification.

Les techniques sémantiques de réduction de dimensionnalité s'affranchissent des trois problématiques en même temps en projetant les données dans un espace de variables latentes et/ou probabilistes et seront détaillées dans le paragraphe suivant. Ces techniques cherchent un espace de projection qui représente au mieux le contenu des documents, espace qui n'est pas forcément discriminant entre les catégories des documents, c'est pour cela qu'elles sont plus utilisées en recherche d'informations qu'en classification.

Dans cette thèse, on a choisi d'optimiser le caractère discriminant des données via une pondération supervisée des termes couplée à un mécanisme d'élimination via des techniques de sélection d'attributs (I.5) sur une représentation vectorielle des données.

I.4.2 La réduction de la dimensionnalité : les modèles à concepts latents

Le plus souvent, un document abordant une thématique principale peut être vu comme une composition de plusieurs sous thématiques (ou idées secondaires/concepts). Par exemple : un article peut aborder à 70% la finance, à 20% la politique et à 10% la guerre. Plusieurs idées secondaires permettent de construire et de générer un document complet avec un sujet principal, cependant, ce mécanisme de génération est une part cachée du résultat final, bien que déductible. L'ensemble des idées secondaires représente un résumé (ou une vue réduite) du

document qu'on peut utiliser par la suite soit pour déduire plus facilement l'idée générale (ou le sujet principale) du document ou bien régénérer carrément une version approximative de ce dernier, de la même façon qu'un étudiant rédige un cours complet à partir de fragments de notes. Les modèles de concepts latents sont des techniques mathématiques ou probabilistes qui déduisent ces composantes cachées à partir d'un corpus permettant ainsi de représenter un document par un ensemble de concepts latents au lieu d'un ensemble de mots. Cette démarche permet à la fois de réduire l'espace de représentation, ainsi que de pouvoir comparer des documents au niveau sémantique et non au niveau des mots. Par exemple, deux documents qui parlent de politique sont proches même s'ils ne contiennent pas les mêmes mots.

L'analyse sémantique latente (*LSA : Latent Semantic Analysis*) est une technique de réduction de dimensionnalité très répondue dans le domaine de la recherche d'information, elle repose sur une décomposition en valeurs singulières (*SVD*) de la matrice termes/fréquences. On suppose qu'on a un ensemble de documents représenté par la matrice X de taille $D \times n$ où D est la taille du vocabulaire (le nombre de termes) et n est le nombre de documents. Après une décomposition en *SVD* de la matrice X on obtient :

$$X_{D\times n} = U_{D\times m}\, S_{m\times m}\, V_{n\times m}^{T} \qquad 2$$

Après la décomposition *SVD*, on garde seulement les $k << m = min(D, n)$ valeurs singulières les plus grandes, on obtient : $\hat{U}_{D\times k}$ les k premières colonnes de U qui représentent les relations entre les documents et les concepts, $\hat{S}_{k\times k}$ la première sous matrice $k\times k$ de S et $\hat{V}_{n\times k}$ les k premières colonnes de V qui représentent les relations entre les concepts et les termes. Mathématiquement $\hat{U}_{D\times k}\, \hat{S}_{k\times k}\, \hat{V}_{n\times k}^{T}$ représente la meilleure approximation au sens des moindres carrés de la matrice X avec k concepts (ou k valeurs singulières). Les n colonnes de $\hat{S}_{k\times k}\, \hat{V}_{n\times k}^{T}$ représentent les nouvelles coordonnées de chaque document après la réduction de dimensionnalité. Pour un nouveau document x^* qui ne se trouve pas dans le corpus de départ, la réduction se fait par une simple multiplication matricielle : $\hat{U}^{T} x^*$.

L'analyse sémantique latente probabiliste ou (*PLSA : Probabilistic Latent Semantic Analysis*) proposée par (Hofmann 1999) est une extension de la *LSA* dans le domaine probabiliste. Le modèle *PLSA* repose sur l'hypothèse d'une indépendance conditionnelle entre les documents d_i et les mots w_i sachant le concept caché z_i (Figure 6).

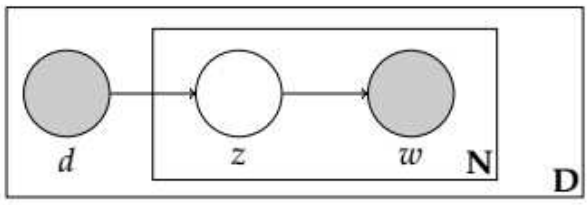


Figure 6 Représentation graphique du modèle PLSA

Le modèle PLSA permet d'associer un concept cachée $z \in Z = \{z_1, z_2, \dots z_k\}$ à un vecteur termes/fréquences représentant la distribution des termes $w \in W = \{w_1, w_2, \dots w_N\}$ dans un document $d \in D = \{d_1, d_2, \dots d_M\}$. Du point de vu génératif, on peut le voir dans le sens suivant :

- La sélection de *d* (un document) avec la probabilité *P(d)*.
- La sélection d'un concept *z* avec la probabilité *P(z|d)*.
- La génération de *w* (un mot) ayant la probabilité *P(w|z)*.

Le modèle génératif obtenu est le suivant :

$$P(d,w) = P(d)\sum_{z\in Z} P(w \mid z)P(z \mid d) \qquad 3$$

La différence entre ce modèle et les modèles classiques (supervisés ou non), est la distribution des mots *w* dans un document *d*, *P(w|d)* obtenue à partir d'une combinaison convexe de probabilités *P(w|z)*. Cette représentation des documents (distribution de concepts) est plus malléable pour déterminer la catégorie d'un document ou bien calculer la similarité entre des documents. L'estimation des paramètres du modèle se fait avec la technique du maximum de vraisemblance (*EM : Expectation Maximization*).

L'allocation latente de Dirichlet (*LDA : Latent Dirichlet Allocation*) représente un modèle probabiliste génératif utilisant la distribution de Dirichlet afin de modéliser la relation entre les mots et les concepts cachés qui expliquent l'apparition de ces mots dans un corpus de documents (Blei et al. 2003). Cette distribution est la généralisation de la distribution *beta* en mode multi-variée, sa densité de probabilité est représentée comme suit :

$$P(x \mid \alpha_1,\dots,\alpha_k) = \frac{\Gamma(\sum_{i=1}^{K}\alpha_i)}{\prod_{i=1}^{K}\Gamma(\alpha_i)}\prod_{i=1}^{K} x_i^{\alpha_i - 1} \qquad 4$$

Où α est un vecteur avec k éléments positif, c'est un paramètre qui fait varier la distribution et Γ est une généralisation de la fonction Gamma où :

$$\Gamma(z) = \int_0^{\infty} t^{z-1}\, e^{-t}\, dt \qquad 5$$

Représenté graphiquement, le modèle génératif de la LDA est schématisé dans la Figure 7. Le procédé de génération est comme suite :

1. Choisir N le nombre de mot du vocabulaire V suivant une loi de Poisson.
2. Tirer une distribution ϕ_k suivant une loi de Dirichelet à paramètre β : $Dir(\beta)$ pour chaque concept k
3. Pour chaque document d :
    - Faire un tirage de la distribution des concepts dans ce document θ_d suivant une loi de Dirichelet à paramètre α, $Dir(\alpha)$.
    - Pour chaque $i \leq N$:
        - Tirer un concept $z_{d,i}$ suivant une loi multinominal(θ_d) parmi les K concepts.
        - Tirer un mot $w_{d,i}$ suivant une loi multinominal($\phi\ z_{d,i}$) parmi les V mots.

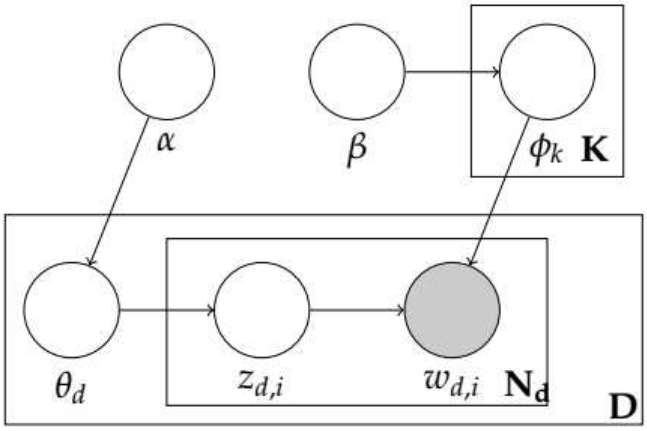


Figure 7 Représentation graphique du modèle LDA

I.5 La sélection d'attributs

Le modèle vectoriel en sac de mots induit des matrices creuses d'une très grande dimension où chaque attribut n'est pas nécessairement pertinent pour la classification. Contrairement aux techniques de réduction de dimensionnalité qui essaient de trouver une représentation compacte des données originales, la sélection d'attributs vise à éliminer les attributs non pertinents. Les techniques de sélection d'attributs se divisent en techniques supervisées et non supervisées où les dernières sont parfois confondues avec les techniques de pondération. Les approches de sélection d'attributs sont divisées de façon générale en trois familles :

1. Les approches filtres : l'évaluation de chaque attribut ce fait de façon indépendante des autres attributs, les attributs avec les meilleurs scores sont gardés. En traitement automatique du langage naturel, il est commun d'utiliser des techniques telles que l'information mutuelle (*MI*), le gain d'information (*IG*) ou le *CHI* statistique (Yang & Pedersen 1997). Ces approches ont l'avantage d'être simples et rapides, cependant, l'interdépendance entre les attributs et la redondance sont ignorées du fait que les attributs sont évalués individuellement ce qui peut dégrader les performances en

classification. Certaines approches filtres telles que *Correlation Feature Selection* (*CFS*) (Senliol et al. 2008) ou *Minimum Redundancy Maximum Relevance* (*MRMR*) (Akadi et al. 2009) tentent d'estimer l'interdépendance entre les attributs et la redondance afin de rectifier l'évaluation, cependant, elles sont beaucoup plus couteuses en temps de calcul que les approches filtres standards.

2. Les approches intégrées : l'évaluation se fait via un modèle d'apprentissage de deux façons différentes.
    a. La sélection arrière : consiste à commencer l'apprentissage du modèle sur l'ensemble complet des attributs, évaluer chaque attribut, puis éliminer récursivement un sous ensemble d'attributs dont l'évaluation est la plus faible.
    b. La sélection avant : contrairement à la sélection arrière, on commence l'apprentissage sur un petit ensemble d'attributs et on ajoute des attributs au fur et à mesure.

    Dans cette famille d'approches on trouve : *Forward Selection with Least Square* (*FSLS*) (Stoppiglia et al. 2003), *Graphting* (Perkins et al. 2003), les arbres de décision (Grabczewski & Jankowski 2005) , l'orthogonalization de Gram-Schmidt (Chen et al. 1989) et l'élimination récurrente par *SVM* (*SVM-RFE* : *Support Vectors Machines Recursive Feature Elimination*) (Guyon et al. 2002). L'avantage de ces approches est la prise en considération de l'interdépendance entre les attributs dans le processus de sélection, cependant, plusieurs apprentissages sont nécessaires, dont le premier sur l'ensemble complet des attributs, ce qui peut s'avérer couteux en temps de calcul selon le modèle utilisé. On note aussi que le sous ensemble d'attributs gardés peut être optimal pour le modèle utilisé en sélection mais peut ne pas l'être sur d'autres modèles.

3. Les approches enveloppées : une technique d'optimisation est utilisée pour choisir un sous ensemble d'attributs qui maximise ou minimise un critère, généralement un modèle prédictive est utilisé afin d'évaluer les sous-ensembles. Les approches enveloppées sont plus gourmandes en temps d'exécution que les approches filtres et les approches intégrées mais trouve généralement de meilleurs résultats. Quelques techniques d'optimisation des plus utilisées en sélection d'attributs sont : les algorithmes génétiques (Ghareb et al. 2016), les approches à estimation de distribution (Larrañaga & Lozano 2002), les essaims particulaires (Aghdam & Heidari 2015) et les colonies d'abeilles artificielles (Karaboga & Basturk 2007).

I.5.1 *CHI statistique*: χ^2

χ^2 mesure la dépendance entre l'attribut *t* et la catégorie *c* en utilisant une table de contingence attribut/catégorie. Dans cette thèse, on a utilisé l'équation 6 proposée dans (Yang & Pedersen 1997). *N* représente le nombre de documents, *A* le nombre de documents dans la catégorie *c* contenant le terme *t*, *D* le nombre de documents qui ne sont pas dans la catégorie *c*

et ne contiennent pas le terme *t*, C le nombre de documents dans la catégorie *c* ne contenant pas le terme *t*, et *B* le nombre de documents qui ne sont pas dans la catégorie *c* et contienne *t*. La mesure χ^2 prend une valeur de zéro en cas où *t* et *c* sont complètement indépendants.

$$\chi 2(t,c) = \frac{N \times (AD - CB)^2}{(A+C) \times (B+D) \times (A+B) \times (C+D)} \quad 6$$

L'équation 6 mesure la dépendance entre les termes et une catégorie, afin de généraliser cette mesure au cas multi-classes on utilise une fonction d'aggrégation, dans nos expériences on a opté pour la fonction moyenne (équation 7) . Cependant, la mesure χ^2 est connue pour être peu significative pour les termes rares.

$$\chi 2(t) = \frac{1}{m} \sum_{i=1}^{m} \chi 2(t, c_i) \quad 7$$

I.5.2 *MI* : Information Mutuelle

L'information mutuelle (*MI : Mutual Information*) est une mesure de la dépendance stochastique entre deux variables aléatoires (Costantea et al. 2008). *MI* est très largement utilisée en modélisation du langage, elle est formulée en équation 8 où *t* est un terme et *c* une catégorie. Cette mesure prend une valeur de zéros si le terme est complètement indépendant de la catégorie, ce qui veut dire : $P(t,c)=P(t)\times P(c)$.

$$I(t,c) = log \frac{P(t,c)}{P(t) \times P(c)} \quad 8$$

Afin de généraliser cette équation au cas multi-classes, on utilise une fonction d'aggrégation de type moyenne (équation 9). L'une des limites de cette mesure sont les termes rares qui peuvent obtenir de très grands scores et être favorisé par rapport aux termes communs.

$$I(t) = \frac{1}{m} \sum_{i=1}^{m} I(t, c_i) \quad 9$$

I.5.3 *GI* : Le gain d'information

En traitement automatique du langage naturel, la *MI* mesure la quantité d'information obtenue pour la prédiction d'une catégorie en sachant la présence d'un terme. Cependant, dans certains cas l'absence d'un terme peut être aussi pertinente. Dans ce contexte, le gain d'information

(*IG : Information Gain*) est une métrique pour la sélection d'attributs qui mesure la quantité d'information obtenue en sachant la présence ou l'absence d'un terme pour chaque catégorie. Dans cette thèse on a utilisé l'équation 10, ou *Rank(t)* mesure la pertinence du terme *t* pour la prédiction des catégories c_i avec $i=1..m$.

$$Rank(t) = -\sum_{i=1}^{m} P(c_i) log(P(c_i)) + P(t) \sum_{i=1}^{m} P(c_i \mid t) \\ log(P(c_i \mid t)) + P(\neg t) \sum_{i=1}^{m} P(c_i \mid \neg t) log(P(c_i \mid \neg t)) \quad 10$$

I.5.4 La sélection d'attributs par l'approche *MSVM-RFE*

La fonction de décision d'un *SVM* linéaire binaire pour la $i^{ème}$ catégorie est formulée dans l'équation 11, où ω_i est le vecteur des poids du *SVM*, *x* est le vecteur en entrée, *b* un scalaire et *nf* le nombre d'attributs.

$$f_i(x) = \sum_{t=1}^{nf} \omega_i(t) \times x(t) + b \quad 11$$

La sélection d'attributs par *SVM-RFE* (*Support Vector Machines Recurrent Feature Elimination*) est une approche de sélection embarquée où l'évaluation de la pertinence des attributs est faite via l'apprentissage d'un *SVM* linéaire sur les données. Proposée par (Guyon et al. 2002) afin d'améliorer la classification des cas de cancer via la sélection des gènes, la mesure de la pertinence de l'attribut *t* est calculée sur la base de l'amplitude de son poids $\omega_i(t)^2$. Intuitivement, les attributs avec une grande amplitude sont ceux qui contribuent le plus dans la fonction de décision du *SVM*, et donc ce sont les plus informatifs. L'élimination des attributs consiste à répéter les étapes suivantes :

1. Faire l'apprentissage d'un *SVM* linéaire sur la base d'apprentissage.
2. Evaluer la pertinence de chaque attribut en utilisant les poids du *SVM*.
3. Trier les attributs par pertinence.
4. Eliminer *N* attributs dont la pertinence est la plus faible de la base.
5. Répéter ce processus sur la nouvelle base restreinte.

MSVM-RFE (Multi-class *SVM-RFE*) est l'extension de cette approche au cas multi-classes (Zhang & Huang 2015). Une façon intuitive pour faire cette extension est d'utiliser une fonction d'agrégation sur les mesures de la pertinence évaluées à partir de plusieurs *SVM* binaires. Dans ce travail on a utilisé la fonction moyenne (voir équation 12), où *Rank(t)* représente l'évaluation de la pertinence moyenne de l'attribut *t* sur toutes les catégories dont le nombre est *nc*.

$$Rank(t) = \frac{1}{nc}\sum_{i=1}^{nc}\omega_i(t)^2 \qquad 12$$

I.6 Travaux récents en sélection et pondération de termes

Dans ce qui suit, les mots : termes, attributs et caractéristiques sont utilisés en tant que synonymes afin de faire référence aux colonnes de la matrice documents/termes/fréquences.

En traitement automatique du langage naturel, il est souvent plus efficace de donner un poids plus fort aux termes expressifs. Cependant, le problème réside dans la définition de critères de mesure de l'expressivité des termes. En recherche d'informations, les données ne sont pas étiquetées et donc une pondération non supervisée est utilisée. Cependant en classification des données textuelles, la pondération supervisée est privilégiée : *RF*, *WLLR*, *CDmax* et *ICF* sont des approches qui utilisent les fréquences de cooccurrence entre les termes et les catégories afin de mesurer la pertinence des termes (Domeniconi et al. 2016).

La pondération des attributs peut être vue comme une forme générale de la sélection d'attributs où les attributs de faible pertinence reçoivent un poids nul au lieu d'être rejetés. En même temps, la plupart des algorithmes de sélection d'attributs trient les caractéristiques par pertinence avant la phase d'élimination, ce classement peut être utilisé pour pondérer les caractéristiques. C'est pourquoi, la plupart des mesures sont utilisés à la fois pour la pondération et la sélection des attributs. Les mesures de sélection d'attributs communes en catégorisation des données textuelles sont *MI*, *IG* et *CHI* (Yang & Pedersen 1997).

Mary Amala Bai et Manimegalai (Mary Amala Bai & Manimegalai 2013) ont proposé un nouveau schéma de pondération supervisé des termes nommé « Termes dans un Document » qui utilise la probabilité d'un terme dans tous les documents d'une catégorie. Les évaluations sur les benchmarks *Reuters-21578* et *20Newsgroups* en utilisant un classifieur basé sur des centroïdes et les *SVM* ont montré une amélioration des performances par rapport aux pondérations *TF*, *IDF* et *RF*.

Wang et ses collègues (Wang et al. 2015) ont proposé deux schémas de pondération supervisée basée sur l'entropie, « *DC: Distributional Concentration* » et «*BDC : Balanced Distributional Concentration* » avec l'hypothèse que les termes ayant une entropie faible dans les catégories du corpus sont les plus discriminants. Contrairement à la plupart des approches de l'état de l'art, *DC* et *BDC* s'appliquent directement au cas multi-classes au lieu de le convertir en plusieurs cas binaires. Les résultats sur deux benchmarks (*Reuters-21578* réduit à 8 catégories et *Snippets*) en utilisant un *SVM* linéaire et les *K-NN* montrent que les schémas de pondération proposés surpassent les schémas de l'état de l'art : *IDF, CHI, IG, RF et ICF*. *BDC* améliore les performances du schéma *DC* sur des catégories avec très peu d'instances.

Abdel Fattah (Abdel Fattah 2015) a proposé de nouveaux schémas de pondération utilisant la densité de l'espace des classes: « *CDall_c : Class Density relative to all class documents*», « *CD_c ; Class Density relative to all documents in the same class* » et «*log_CD : logarithmic ratio between Class Densities*» . Les expériences menées sur des ensembles de données en

analyse des sentiments (commentaires sur des films, livres, DVD, produits électroniques et cuisine) en utilisant les *SVM*, les Réseaux de Neurones Probabilistes (*PNN*) et les Modèles à Mélange de Gaussiennes (*GMM*) montrent que les schémas de pondération proposés (en particulier *CDall_c*) surpassent les schémas classiques : *IDF, ICF, WLLR, MI et CHI*. Une combinaison entre *SVM, PNN* et *GMM* à travers une stratégie de vote a amélioré d'avantage les performances.

Lamirel et ses collègues (Lamirel et al. 2015) montrent qu'une adaptation de la métrique de maximisation des termes permet d'améliorer la sélection d'attributs en classification supervisée. L'approche proposée : « *FMC : Feature Maximization and Contrast* » a été comparée aux approches *CHI*, *IG*, filtres de consistance, l'incertitude symétrique et l'analyse en composantes principales sur les benchmarks *Reuters-21578*, *Amazon*, *20Newsgroups* et *WebKB* en utilisant plusieurs classificateurs: *SVM,* arbres de décision, forêt d'arbres décisionnels, *K-NN*, classifieur bayésien naïve, réseau bayésien, et montre une amélioration significative des performances.

Domeniconi et ses collègues (Domeniconi et al. 2016) ont développé de nouvelles variantes supervisées de la pondération *IDF*. La première variante, « *IDFEC : Inverse Document Frequency Excluding Category* », prend en compte le fait que des termes apparaissent fréquemment dans une même catégorie, afin que ces termes ne soient pas pénalisés par la pondération *IDF*. La deuxième variante, « *IDFEC-b : Inverse Document Frequency Excluding Category-Based*», combine *IDFEC* avec une pondération *RF* afin de booster ses performances. Les expériences avec un *SVM* linéaire sur les benchmarks *Reuters-21578*, *20Newsgroups*, *Movie Review Data* et *Multi-Domain Sentiment Datasets*, montrent via des tests statistiques que *IDFEC-b* surpasse les pondérations *IDFEC*, *IDF*, *RF*, *IG*, *WLLR* et *CDmax*.

Li (Li 2016) met en évidence deux problèmes dans les mesures de sélection de termes classiques qui sont basées sur la fréquence des documents : la distribution des termes sur les catégories et l'importance des termes pour chaque document sont ignorées. Il propose une mesure supervisée améliorée « *CBIWDF : Class Based and Importance Weighted Document Frequency* », les expériences avec un classifieur linéaire sur les benchmarks *20Newsgroups* et *Sector* montrent que *CBIWDF* est plus performant que *CHI* et *IG*.

Cependant, la mesure de pertinence individuelle des termes n'est généralement pas suffisante pour choisir un sous-ensemble pertinent de caractéristiques. Javed et ses collègues (Javed et al. 2015) considèrent que la redondance entre les caractéristiques peut réduire les performances du classifieur et suggère un algorithme de sélection d'attributs en deux étapes. Cet algorithme commence par un tri des termes à l'aide d'une méthode de sélection d'attributs standard (*BNS : séparation bi-normal, IG*), suivie par le filtre *Markov Blanket*, un algorithme de sélection de sous-ensemble d'attributs, qui permet d'éliminer les attributs redondants. Les expériences sur les benchmarks *Reuters-21578*, *TREC* et *OHSUMED* ont révélé l'efficacité de cette combinaison par rapport à l'approche *BNS*, *IG*, *DFS: Distinguishing Feature Selector* et une combinaison entre l'approche *IG* et l'analyse en composantes principales.

Plutôt que de mesurer la pertinence de chaque attribut individuellement, un modèle peut être utilisé pour estimer la pertinence d'un attribut au sein d'un ensemble d'attributs. Zhang et Huang

(Zhang & Huang 2015) ont étendu l'approche *SVM-RFE* (Guyon et al. 2002) à un cas multi-classes. *MSM-RFE* réalise l'apprentissage de plusieurs *SVM* linéaires binaires afin de mesurer la pertinence des attributs pour chaque catégorie et une fonction d'agrégation est utilisée pour combiner la mesure pour l'ensemble des catégories. Un sous ensemble d'attributs ayant la mesure la plus faible est éliminé, et ce processus est réitéré sur les attributs restants. Les expériences sur des données génétiques montrent l'efficacité de la sélection *MSVM-RFE* en termes de taux de réduction, d'amélioration des performances de la classification, ainsi qu'à traiter des bases d'apprentissage d'une très grande dimension tout en tenant compte des interdépendances entre les attributs.

Les approches proposées par Zhang (Zhang & Huang 2015) et Javed (Javed et al. 2015) ne permettent pas de considérer les interdépendances et la redondance entre les attributs simultanément. Trouver un sous-ensemble minimal de termes qui maximise les performances en classification et réduit la redondance est une tâche d'optimisation très difficile. Les métaheuristiques et surtout les Approches Evolutionnaires (AE) sont des outils puissants pour résoudre des problèmes d'optimisation difficiles.

Dans ce cadre, Aghdam et Heidari (Aghdam & Heidari 2015) décrivent une procédure dans laquelle l'optimisation par essaims de particules (*PSO*) est utilisée pour la sélection des caractéristiques dans la catégorisation des données textuelles. Le nombre de caractéristiques éliminées et la performance du classificateur (*K-NN*) sont maximisés dans la fonction de fitness. Une série de tests a été menée sur le benchmark *Reuters-21578* en utilisant l'optimisation *PSO* et comparée à *IG*, *CHI* et l'optimisation par les algorithmes génétiques afin de montrer l'efficacité des *PSO* en sélection d'attributs.

Dans le même contexte, Paul et Das (Paul & Das 2015) ont utilisé un algorithme évolutionnaire multi-objectifs basé sur la décomposition *MOEA/D* pour la pondération et la sélection simultanées des attributs. Deux mesures sont considérées : maximiser la distance inter-classes et minimiser la distance intra-classe d'un ensemble de données. Une pénalité est ajoutée à la fonction de fitness afin de réduire le nombre d'attributs. Les expériences utilisant un classifieur *K-NN* sur des benchmarks montrent l'efficacité de cette approche.

Pérez-Rodríguez et ses collègues (Pérez-Rodríguez et al. 2015) ont présenté un framework pour la sélection et la pondération simultanées des instances et des attributs en utilisant une approche évolutionnaire. La fonction de fitness vise à maximiser les performances de la classification avec la règle du plus proche voisin en utilisant un ensemble minimal d'attributs et d'instances. Les expériences sur des benchmarks avec différentes combinaisons de ces quatre étapes : sélection d'instances, sélection d'attributs, pondération d'instances et pondération d'attributs, montrent que la pondération d'attributs seule permet d'obtenir les meilleures performances en classification, cependant la combinaison sélection d'attributs et d'instances permet une meilleure réduction. Une amélioration marginale est obtenue en ajoutant une pondération des attributs à la dernière combinaison.

Escalante et ses collègues (Escalante et al. 2015) décrivent une approche pour l'apprentissage d'un algorithme de pondération de termes optimisé pour la classification des données textuelles en utilisant la programmation génétique. Le programme génétique apprend à combiner de façon

optimisée un ensemble de méthodes de pondération standards par le biais d'opérateurs mathématiques afin d'en améliorer les performances en classification avec un *SVM* linéaire. Des expériences sur les benchmarks : *Reuters-21578, 20Newsgroups, WebKB, CADE-12, TDT-2, Classic-4*, montrent que le programme génétique arrive à apprendre de nouveaux schémas de pondération qui surpasse les schémas standards : *BIN, TF, IDF, RF, CHI, IG*.

I.7 Représentation des images aériennes

La classification des images aériennes représente un chalenge très difficile en traitement d'images. La structure d'une image aérienne est très complexe, elle comporte plusieurs motifs et textures disposés d'une façon complètement anarchique. Plusieurs autres facteurs viennent se rajouter à cette difficulté : les transformations géométriques en rotation et en translation ou le changement d'échelle, les conditions de luminosité ou en général les conditions météorologiques en plus des altérations dues à l'homme ou aux phénomènes naturels. La classification des images aériennes dépend justement de la qualité des caractéristiques extraites à partir des images qui se doivent d'être robustes et invariantes vis-à-vis ces altérations. Parmi les techniques de l'état de l'art, l'apprentissage profond plus connu par le nom *« Deep Learning »* est une véritable révolution dans le domaine vu la pertinence et la robustesse des caractéristiques qui en dérivent et sera abordé dans le chapitre suivant. Dans ce qui suit on parlera brièvement de la représentation des textures dans une image avant de nous attarder sur un état de l'art récent des travaux menés sur la base *UCMerced LandUse* qui est un benchmark de référence dans le domaine des images aériennes.

La texture joue un rôle important dans plusieurs domaines en vision artificielle, la texture offre des informations très pertinentes pour l'analyse, la description et l'interprétation des images comme la granularité, la régularité, la périodicité, les structures géométriques etc. La texture représente une partie d'une image avec une répétition d'un même motif où l'intensité des pixels est structurée d'une façon très complexe. Par exemple la texture herbe, textile, briques d'un mur ou même les motifs dans une IRM d'un cerveau (voir Figure 8). Cette structure complexe de la texture rend sa classification très difficile, et donc, la nécessité de techniques de représentations pertinentes et robustes aux légères variances, au bruit et aux transformations géométriques.

Les techniques de représentation des textures peuvent être divisées en plusieurs catégories :

- Les techniques basées sur la variation de l'intensité des pixels : *GLCM* (*Gray-Level Co-occurrence Matrix*) ou matrice de cooccurrence, *LBP* (*Local Binary Patterns*) ou histogramme des motifs binaires locaux etc.
- Les transformées : transformée de Fourier 2d, Gabor, en ondelettes etc.
- Les techniques basées sur les points d'intérêts (ou caractéristiques locaux) : *SIFT* (*Scale-Invariant Feature Transform*), *SURF* (*Speeded Up Robust Features*) etc.
- Les techniques basées sur les régions : croissance des régions, clustering etc.

- Les techniques basées sur des modèles extracteurs de caractéristiques : chaines de Markov, réseaux de neurones et réseaux bayésiens etc.

Dans cette thèse on s'intéresse aux méthodes de la première et à la dernière catégorie, vu leur rapidité, leur robustesse au bruit et l'efficacité des techniques récentes qui en dérivent. Le chapitre suivant abordera quelques architectures neuronales qui peuvent être utilisées comme extracteurs de caractéristiques.

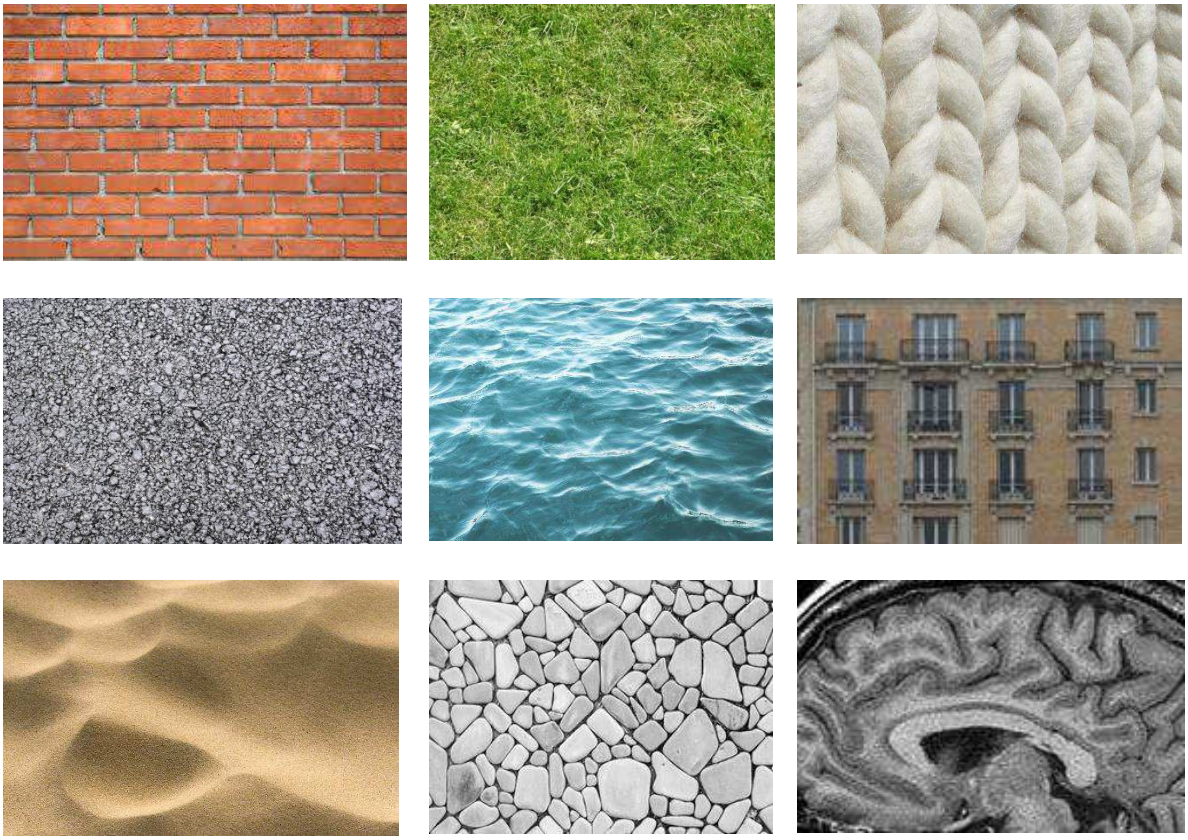


Figure 8 Exemples d'images avec des textures

I.7.1 L'espace des couleurs

Le choix de l'espace des couleurs adéquat est très important en traitement d'images. L'espace *RVB* (Rouge, Vert, Bleu) est l'espace de représentation par défaut de la plupart des formats d'images. Les autres espaces de couleurs peuvent être obtenus à partir de transformations linéaires ou non linéaires de l'espace *RVB*. Cette transformation d'espace permet d'extraire des caractéristiques plus pertinentes et plus robustes aux variations. Par exemple, il est commun d'utiliser la chrominance et d'ignorer l'intensité des pixels afin de comparer des images prises dans des conditions de luminosités différentes.

Plusieurs espaces de couleurs ont été proposés dans la littérature, l'espace *RVB* étant le plus utilisé pour la capture, le stockage et la représentation des images numériques. Certaines variantes ont été proposées en introduisant un mécanisme de normalisation (ex : R+V+B=1) afin d'ignorer la composante de luminosité. Certaines caractéristiques perceptuelles des pixels comme la teinte, la saturation et l'intensité ne peuvent pas être décrites directement à partir de l'espace RVB, plusieurs transformations non linéaires sont proposées afin de faire cette conversion.

Dans l'espace *TSL* ou *HSL* (*Hue, Saturation, Lightness*) chaque couleur est définie par les composantes suivantes :

- Teinte : décrit la couleur du pixel.
- Saturation : décrit la chromaticité (l'intensité de la coloration du pixel), ce qui peut indiquer une couleur vive ou fade.
- Luminosité : ou intensité décrit la variation de la luminosité de la couleur du pixel.

Dans le but de réduire la redondance qu'on peut trouver dans certains espaces de couleurs, l'espace *YCbCr* (Luminance chrominance) se caractérise par son orthogonalité. La luminance (Y) est décrite par une somme pondérée des composantes *RVB*, alors que la chrominance (Cb et Cr) est calculée par soustraction de la luminance des composantes B et R.

Dans l'espace *Lab*, une couleur est représentée par la Luminance (*L*) et deux composantes de chrominance (*a* et *b*) qui représentent de façon uniforme la différence perçu entre deux couleurs pour l'œil humaine. Cependant cette uniformité est obtenue par une transformation couteuse en temps de calcul. Cet espace permet de décrire les couleurs de surface, et donc, il est plus utilisé pour l'impression ou la fabrication d'objets colorés que pour l'affichage dans un écran. Néanmoins, il permet au mieux d'imiter la réponse de l'œil humaine face aux nuances des couleurs et est donc très répondu en traitement d'images.

I.7.2 L'histogramme des couleurs

L'histogramme permet d'estimer la distribution des nuances de couleurs dans une image, en divisant l'espace des valeurs possibles en plusieurs intervalles et en calculant le nombre d'occurrences des couleurs dans chaque intervalle. Le vecteur obtenu est souvent normalisé afin d'obtenir des probabilités et obtenir un vecteur caractéristique indépendant de la taille de l'image.

L'histogramme permet une très bonne description des composantes d'une texture dans une image, cependant il est plus commun de le calculer sur des images monochromes (ex : en niveau de gris). L'extension aux images couleurs se fait généralement soit par un calcul indépendant de l'histogramme sur chacune des dimensions puis une concaténation des vecteurs, soit par un clustering des couleurs puis une quantification. Pour des images comportant plusieurs types de textures, l'estimation des histogrammes est faite généralement sur des blocs de petite taille de l'image.

Afin d'obtenir des caractéristiques plus robustes, il est très commun d'estimer l'histogramme sur d'autres espaces de couleurs que le *RVB*, tel que le *HSL*, le *YCbCr* ou le *Lab*.

I.7.3 GLCM : Matrice de cooccurrence des niveaux de gris

GLCM : *Gray-Level Co-occurrence Matrix* en anglais, est une méthode de représentation statistique qui permet d'examiner une texture tout en considérant des relations spatiales entre les pixels, aussi connue sous l'appellation : matrice des dépendances spatiales. Dans cette représentation, une matrice carrée permet de caractériser une texture par le nombre d'occurrences où chaque paire de pixels avec des valeurs spécifiques sont dans une relation spatiale définie. Plus spécifiquement, la matrice *GLCM* est représentée en lignes et en colonnes par les différentes valeurs possibles des nuances des couleurs des pixels, la valeur de la case (i, j) représente le nombre de paires de pixels de l'image ($x1$, $y1$) et ($x2$, $y2$) où image($x1$, $y1$)=i et image($x2$, $y2$)=j avec la contrainte que les coordonnées ($x1$, $y1$) et ($x2$, $y2$) soient dans une relation (voisinage) prédéfinie $f(o,d)$, où o représente une orientation et d une distance. Dans le cas de f ($90°$, d) on comptabilise toutes les paires de pixels ($x1$, $y1$) et ($x2$, $y2$) où ($x2$, $y2$) est adjacent à ($x1$, $y1$) par une distance de d pixels dans une direction de 90° (voir équation 13, où # est le symbole de la cardinalité).

$$GLCM_{f(90°,d)}(i,j)=\#\{(x_1,y_1),(x_2,y_2)|\ x_1=x_2,\ |y_2-y_1|=d,\ image(x_1,y_1)=i\ \&\ image(x_2,y_2)=j\} \qquad 13$$

Après une estimation de la matrice *GLCM* d'une texture, il est possible d'ajouter au vecteur de représentation obtenu certaines informations statistiques telles que le contraste, la corrélation, l'énergie et l'homogénéité. Il est possible d'étendre cette représentation aux images couleurs par concaténation des caractéristiques des différentes dimensions des couleurs.

I.7.4 *LBP* : Motif binaire local

Cette méthode (Ojala et al. 2002) de représentation des textures monochromes est à la fois invariante aux rotations, aux translations, aux variations de la luminosité ainsi qu'au faible bruit. Cette méthode a l'avantage de ne comporter que des calculs simples ainsi que très peu de paramètres, il s'agit de définir le rayon R d'une fonction de voisinage circulaire et symétrique autour du pixel à coder ainsi que P le nombre de pixels voisins (voir Figure 9).

Le codage d'un pixel de l'image par l'approche LBP passe par la soustraction de l'intensité du pixel avec chacun de ses voisins, puis par la binarisation du résultat où les valeurs négatives sont converties en 1, sinon en 0. La chaine binaire obtenue est convertie au décimal par la suite en multipliant la valeur binaire de chaque pixel voisin par 2^n où n est l'ordre du pixel voisin (voir Figure 10). Le vecteur caractéristique de l'image avec la représentation LBP est obtenu en estimant un histogramme sur les valeurs codées.

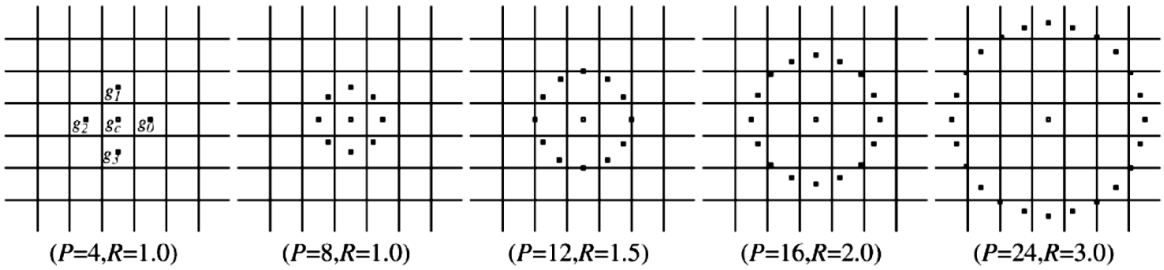


Figure 9 Voisinages circulaires en fonction de P et R (Ojala et al. 2002)

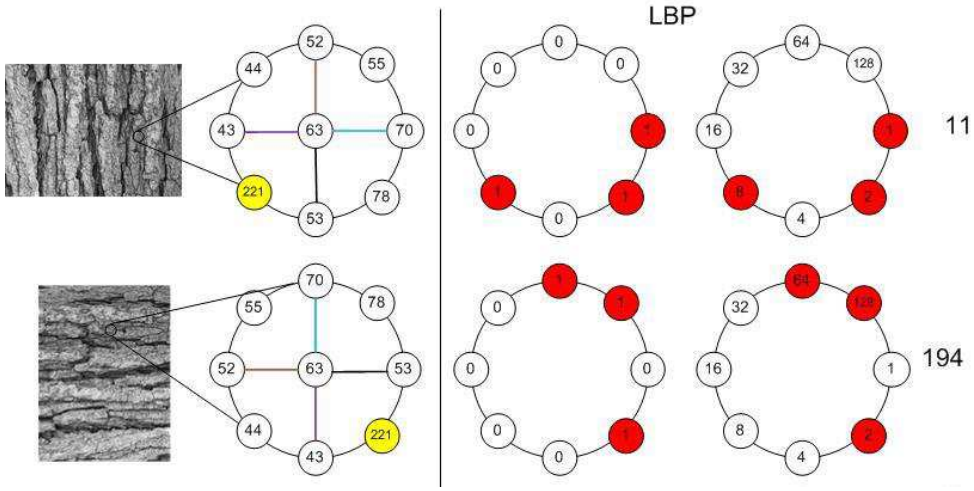


Figure 10 Exemple du codage LBP d'un pixel dans une image

I.8 Travaux récents sur la classification d'images aériennes

La classification des images aériennes représente une des tâches des plus complexes en vision artificielle. Selon les catégories d'images prises en considération, elles comportent plusieurs motifs et textures différents dont la structuration spatiale est déterminante pour la classification. L'état de l'art qui suit démontre qu'une bonne représentation des images est cruciale sur les performances de la classification.

Yang et Newsam (Yang & Newsam 2010) ont présenté une méthode de représentation d'images aériennes haute résolution basée sur un sac de mots visuels. Cette représentation s'intéresse à la fréquence des caractéristiques et non à leur localisation comme trait discriminent entre les classes par analogie à l'approche sac de mots en classification des documents textuels qui ignore l'ordre des mots. Les auteurs considèrent aussi deux extensions spatiales : en premier, un noyau qui traite l'ordre spatial absolu des caractéristiques de l'image, en deuxième, le noyau de cooccurrence spatial qui traite l'ordre relatif des caractéristiques. Ces approches ont été évaluées et comparées avec des approches standards sur la base d'apprentissage *UCMerced LandUse* qui a été réalisée par les auteurs du papier. Les résultats montrent que l'intégration des noyaux spatiaux dans la représentation en sac de mots permet d'améliorer les

performances et représente une alternative robuste qui concurrence de prêt les meilleures approches standards.

Negrel et ses collègues (Negrel et al. 2014) ont expérimenté l'utilisation des statistiques du second ordre afin d'améliorer la qualité des attributs extraits à partir des images : les vecteurs de Fisher ainsi que les vecteurs des descripteurs et de tenseurs agrégés localement. Ces vecteurs sont combinés avec des techniques de normalisation et d'orthogonalisation et appliqués sur la base *UCMerced LandUse*, les résultats montrent la pertinence de ses attributs et une amélioration par rapport à l'approche en sac de mots visuels.

La modélisation d'une image décomposée en parties plus petites, est une technique qui a prouvé son efficacité dans la littérature. Les *Sparselets* sont des modèles partagés universels préétablis à partir d'une grande collection de modèles de parties d'images et permettent du coup d'optimiser la vitesse de la modélisation. Cheng et ses collègues (Cheng et al. 2015) proposent une nouvelle approche pour l'apprentissage de *Sparselets* beaucoup plus performantes. L'apprentissage est divisé en deux parties : la première réalise un apprentissage non supervisé des *Sparselets* via un auto-encodeur à une seule couche cachée, puis un apprentissage supervisé simultané des *Sparselets* et d'un réseau de neurone à une seule couche cachée pour faire la classification. Afin d'obtenir des vecteurs creux (avec des zéros), les auteurs ont imposé les contraintes de sparsité de la norme *L0* sur les vecteurs d'activation du réseau de neurone. Les tests sur les bases *PASCAL VOC 2007*, *MIT Scene-67* et *UCMerced LandUse* montrent des résultats prometteurs en classification comparés avec des *Sparselets* classiques.

Penatti et ses collègues (Penatti et al. 2015) ont évalué les caractéristiques extraites à partir de réseaux *CNN* pré-entrainés sur la base *ImageNet*. Cette base est un ensemble d'images d'objets du quotidiens contenant 1000 catégories. L'étude montre que les caractéristiques extraites à partir d'un *CNN* pré-entrainé sur une grande base d'images (qui n'a rien avoir avec des images aériennes) sont très pertinentes et peuvent être réutilisées sur d'autres bases d'images. La combinaison entre les caractéristiques des *CNN* pré-entrainés (*OverFeat* et *Caffe*) a permis d'améliorer les résultats de la classification des images aériennes de la base *UCMerced LandUse* par rapport aux descripteurs classiques. Les tests des descripteurs des *CNN* pré-entrainés sur la classification d'images satellitaires (*Brazilian Coffee Scenes dataset*) ont donné d'excellents résultats bien qu'ils n'étaient pas les meilleurs. Une expérience similaire a été faite par Marmanis et ses collègues (Marmanis et al. 2016) qui ont mené une étude sur le *CNN* pré-entrainé *OverFeat*, les tests sur la base *UCMerced LandUse* ont montré un gain important en classification par rapport aux approches de l'état de l'art : sac de mots visuels, codage creux, MinTree+KD-Tree.

Luus et ses collègues (Luus et al. 2015) montrent que présenter la même image en multiples échelles à un même *CNN* profond permet d'améliorer les performances en classification. Les tests sur la base *UCMerced LandUse* montrent un gain de plus de 5% par rapport à un apprentissage sur une seule échelle.

Wu et ses collègues (Wu et al. 2016) proposent une architecture hybride, le banc de filtres profonds qui combine un auto-encodeur multi-colonnes à valeurs creuses et les vecteurs de Fisher afin d'extraire des vecteurs de caractéristiques hiérarchiques à partir de l'image à la fois

représentatifs, discriminants et robustes. L'architecture est optimisée par un apprentissage afin d'extraire les caractéristiques les plus adaptées à la classification des images aériennes et satellitaires. Les tests sur les bases *UCMerced LandeUse* et *RSSCN7 data* ont montré un gain en performances par rapport à plusieurs approches classiques tel que le codage *LBP*, les sacs de mots visuels, la transformée de Gabor en couleur, la transformée en Ondelette et les histogrammes de couleur.

Zou et ses collègues (Zou et al. 2016) proposent une nouvelle approche pour la représentation des images basée sur la fusion de caractéristiques spatiales locales et globales. L'image est divisée en plusieurs régions denses, et un codebook visuel en est construit en premier lieu suivit d'un clustering par *k-means*. Un codage linéaire est appliqué sur les régions denses via le codebook construit, et une stratégie de matching pyramidale est utilisée afin de combiner les caractéristiques locales de l'image. Cette approche est appliquée à la fois à l'image en nuances de gris et à sa transformée de Gabor. L'extraction des caractéristiques locales et globales est faite avec des noyaux de classification collaboratifs basés représentation, l'image est classifiée dans la catégorie qui minimise l'approximation résiduelle après la fusion des caractéristiques. L'approche est testée sur plusieurs bases d'images : *UCMerced LandeUse*, *19-class satellite scene*, *15-scene categories*, *sports event categories*. Les résultats montrent un gain en performances comparés avec des techniques classiques comme les sacs de mots visuels, la transformée de Gabor, d'ondelettes et les histogrammes en couleur et des variantes du codage *LBP*.

Avramović et Risojević (Avramović & Risojević 2016) ont publié une étude comparative sur différentes approches de représentations des images aériennes et proposent une réduction de la dimensionnalité de ces approches basée sur une analyse en composantes principales. Les tests sur les bases *UCMerced LandeUse* et *In-house* montrent un gain en performances sur les approches de représentation basées sur les points *SIFT*.

Cheng et ses collègues (Cheng et al. 2016) se sont intéressés aux *CNNs : AlexNet*, *VGGNet* et *GoogleNet,* pré-entrainés sur la grande base *ImageNet*. Les tests portés sur l'utilisation en premier lieu des caractéristiques extraites (la sortie de la couche avant dernière) de ces réseaux avec un *SVM* linéaire sur la base *UCMerced LandeUse* ont montré des taux de reconnaissance très élevés. En deuxième lieu, les auteurs proposent de remplacer la dernière couche de ces *CNNs* (la couche discriminative des 1000 classes d'*ImageNet*) par une nouvelle couche initialisée par des poids aléatoires à *C* classes, où *C* représente le nombre de classes de la base d'image traitée. L'apprentissage est repris avec un petit pas, les caractéristiques obtenues après la phase de réapprentissage à partir des images sont classifiées par un *SVM* linéaire. Une fonction de décision concurrentielle a été ajoutée afin de choisir la plus forte réponse parmi le premier *SVM* et le deuxième, les tests sur la base *UCMerced LandeUse* ont montré un gain de 2 à 5% sur les résultats du premier test. Une approche similaire est proposée par Scott et ses collègues (Scott et al. 2017) en dressant le problème de la taille des bases d'apprentissage qui est trop petite. Les auteurs proposent non seulement d'utiliser des *CNNs* pré-entrainés sur la base *ImageNet* (*CaffeNet, GoogLeNet, ResNet*) comme réseaux initiaux pour l'apprentissage ainsi qu'une augmentation de la base des exemples par des transformations géométriques : translations et rotations. Cette proposition a permis d'obtenir un réel booste des performances

sur les bases *UCMerced LandeUse* et *RSD*. Des expériences similaires menées par Weng et ses collègues (Weng et al. 2017) sur le réseau *AlexNet* montrent une amélioration des performances sur la base *UCMerced LandeUse* en remplaçant la dernière couche du *CNN* par un classifieur *ELM* (*Extreme Learning Machine*).

I.9 Conclusion

Ce chapitre passe en revue, de façon générale, des schémas pour la représentation de différents types de données et souligne la possibilité d'automatisé le processus de génération, d'apprentissage et de recherche de représentations optimisées pour la classification. Ce chapitre ce focalise par la suite sur les différentes représentations utilisées en traitement automatique du langage naturel et en classification des textures.

Ce chapitre met l'accent sur les limites du modèle vectoriel, communément utilisé dans le domaine de la classification des données textuelles, avant de présenter des alternatives via les techniques de réduction de la dimensionnalité à base de concepts latents ainsi que la réduction de l'espace de représentation via la sélection d'attributs. Un état de l'art des travaux récents est présenté afin de montrer l'orientation actuelle des techniques de représentation dans le domaine.

La deuxième partie du chapitre aborde les difficultés issues du domaine de la classification des images aériennes et le met en relation avec le domaine de la classification des textures. Plusieurs techniques de représentation des textures sont abordées, et un état de l'art riche et récent est présenté afin de mettre en évidence les méthodes de représentation actuelles en classification d'images aériennes.

CHAPITRE 2

Apprentissage supervisé via les modèles connexionnistes

II.1 Introduction

Le connexionnisme est un paradigme des sciences cognitives qui vise à expliquer les capacités intellectuelles avec un réseau complexe d'unités de calcul simples analogues des neurones biologiques. Les réseaux de neurones artificiels essaient de modéliser les facultés du cerveau humain pour des tâches complexes comme la reconnaissance des formes ou la modélisation du langage en calculant des poids de connexion entre ces unités de calcul (Garson 2016). Par analogie à la biologie, on parle d'apprentissage et de poids synaptiques entre neurones artificiels. L'application des réseaux de neurones en intelligence artificielle a connu une monté fulgurante surtout après l'implémentation proposée par (Rumelhart et al. 1986) et les travaux de (Hornik 1991) qui ont prouvé que ces modèles étaient capables d'apprendre n'importe quelle fonction à condition d'avoir une relation déterministe entre les entrées et les sorties et qu'un nombre suffisant de neurones et de connexions ait été spécifié.

Le connexionnisme est en constante concurrence avec la modélisation symbolique du cerveau qui applique des règles déterministes manipulant des symboles qui représentent des concepts complexes. Le principal avantage des modèles connexionnistes est leur robustesse face aux entrées bruitées, les exceptions et même la perte de quelques neurones ce qui les rend très pratiques pour des implémentations réels (Kaumanns 2016). Les principales limites des modèles actuels viennent du fait d'ignorer certaines propriétés du cerveau telles que la variété des types de neurones et les effets des neurotransmetteurs et les hormones. En plus, plusieurs règles d'apprentissage telles que la rétropropagation du gradient ne trouvent pas d'analogue dans le domaine biologique car trop lentes et l'absence de connexions inverses nécessaires dans ces règles les rend biologiquement improbables. Pour finir la plupart des modèles connexionnistes sont peu performants pour le traitement à base de règles, le résonnement et d'autres formes d'intelligence de plus haut niveau (Garson 2016).

Dans ce chapitre, on aborde brièvement quelques architectures neuronales à multiples couches parmi les plus connues en apprentissage supervisé, celles à propagation strictement en avant et celles avec des connexions récurrentes. On présente par la suite le principe des modèles connexionnistes en *Deep Learning* avant de nous attarder avec plus de détails sur les réseaux de neurones à convolution et les modèles pré-entrainés. La fin du chapitre met en avant certaines limites des réseaux de neurones liées au surapprentissage, et propose des solutions via les modèles *SVM* qui imposent des contraintes sur les séparateurs entre les catégories, et limite du coup le risque du surapprentissage.

II.2 Les modèles connexionnistes à multiples couches à propagation avant

La révélation des limites du perceptron linéaire par (Minsky & Papert 1972) dans le traitement des problèmes non linéairement séparables, a encouragé les recherches à s'orienter vers des modèles beaucoup plus complexes, notamment les architectures à multiples couches. Ces modèles sont capables de projeter des données complexes vers un nouvel espace de représentation, plus communément appelé espace des concepts afin de simplifier la représentation de ces données et faciliter leur interprétation.

Le perceptron multicouche (*MLP* : *Multi-Layer Perceptron*) est un modèle de type *feedforward*, ce qui veut dire que les données circulent dans une seule direction de la couche d'entrée vers la couche de sortie. Ce modèle contient trois types de couches, une couche d'entrée qui assure le passage des données dans le réseau et ne contient pas de neurones, une à plusieurs couches cachées qui permettent de transformer la représentation des données et une couche de sortie qui représente la réponse du modèle. Dans ce modèle chaque couche est complètement interconnectée avec les neurones de la couche suivante par des connexions pondérées (poids synaptiques) simulant des connexions synaptiques d'un neurone biologique (voir Figure 11).

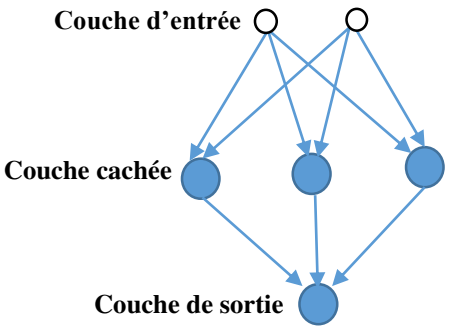


Figure 11 Exemple d'un perceptron multicouche

Chaque neurone calcule une sommation pondérée des entrées qui passe par une fonction d'activation non linéaire (ex : une sigmoïde ou une tangente hyperbolique) donnant la réponse du neurone (voir Figure 12). L'apprentissage du modèle se fait généralement par la règle du rétropropagation du gradient de l'erreur qui consiste à trouver les poids synaptiques qui font correspondre au mieux sa réponse avec la sortie désirée et ce fait en deux phases :

a. Une passe avant qui permet de calculer la réponse du modèle et l'erreur par rapport à la sortie désirée.
b. Une passe arrière qui permet de propager l'erreur calculée vers les couches inférieures et de corriger les poids synaptiques.

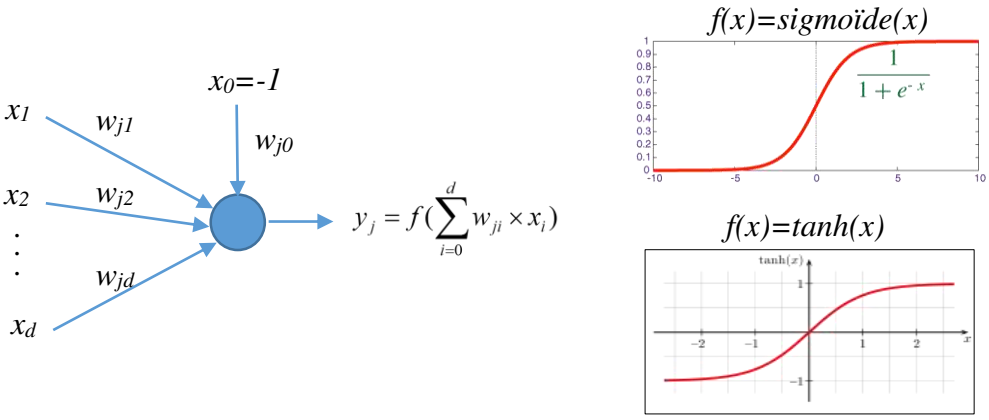
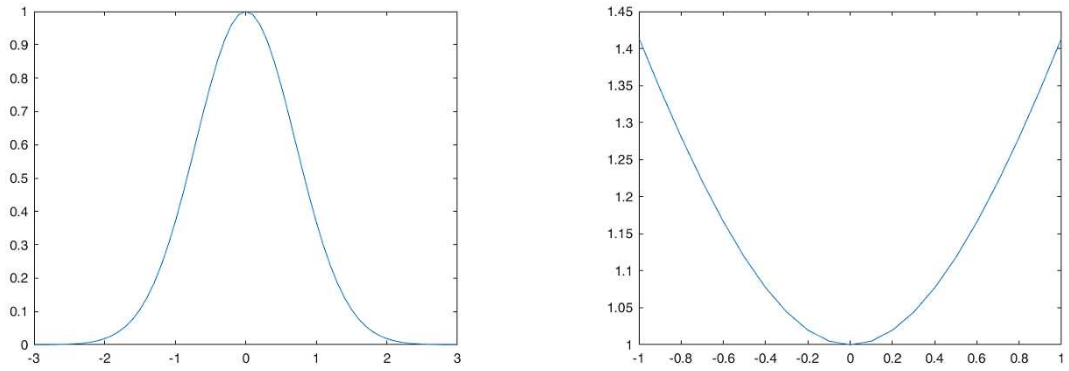


Figure 12 Calcul de la réponse d'un neurone artificiel

Dans le cadre d'un problème de classification de données non linéairement séparables, la couche cachée permet de faire une transformation en projetant les données dans un nouvel espace de représentation où ces données deviennent linéairement séparables. En théorie une seule couche cachée est suffisante pour faire du *MLP* un approximateur universel à condition d'y mettre un nombre suffisant de neurones. Cependant, on verra par la suite que l'utilisation de plusieurs couches cachées peut être très bénéfique (l'apprentissage profond, section II.4). Hormis sa puissance, ce modèle est souvent sujet aux optimums locaux, au surapprentissage et à la lenteur de la convergence pour de grandes bases d'apprentissage, un réglage empirique des paramètres du modèle est souvent nécessaire afin d'obtenir de bonnes performances.

Un réseau à fonctions de base radiales (*RBF* : Radial Basis Function) est un autre modèle de type *feedforward*, son architecture ressemble à celle du *MLP*, cependant il utilise des fonctions d'activation de type radial. C'est une classe de fonctions dont la réponse croit ou décroit de façon monotone suivant la distance par rapport à un point central et un écart type (Figure 13).



(a) *(b)*

Figure 13 Exemples de fonctions radiales

(a) : $f_{\mu,\sigma}(x) = \exp(-\frac{(x-\mu)^2}{\sigma^2})$ *avec μ=0 et σ=1* *(b) :* $f_{\mu,\sigma}(x) = \frac{\sqrt{\sigma^2 + (x-\mu)^2}}{\sigma^2}$ *avec μ=0 et σ=1*

Introduit par Hardy (Hardy 1971) et développé par Powell (Powell 1985) ce réseau était utilisé initialement en interpolation, puis généralisé comme un modèle connexionniste par la suite. Comme pour le *MLP* ce réseau comprend une couche d'entrée, une ou plusieurs couches cachées et une couche de sortie (Figure 14). Cependant, les poids synaptiques ne sont présents qu'au niveau de la couche de sortie. Initialement, ce modèle comprenait une seule couche cachée où le nombre de neurones était égale au nombre d'exemples. Cette solution n'est pas envisageable pour de grandes bases d'apprentissage avec beaucoup d'instances, de ce fait, le modèle est spécifié avec un nombre fixé empiriquement de neurones en couche cachée, l'apprentissage consiste à trouver les points centraux et les écarts types des fonctions radiales

ainsi que les poids synaptiques de la couche de sortie. Ce modèle est aussi sujet au surapprentissage et aux optimums locaux.

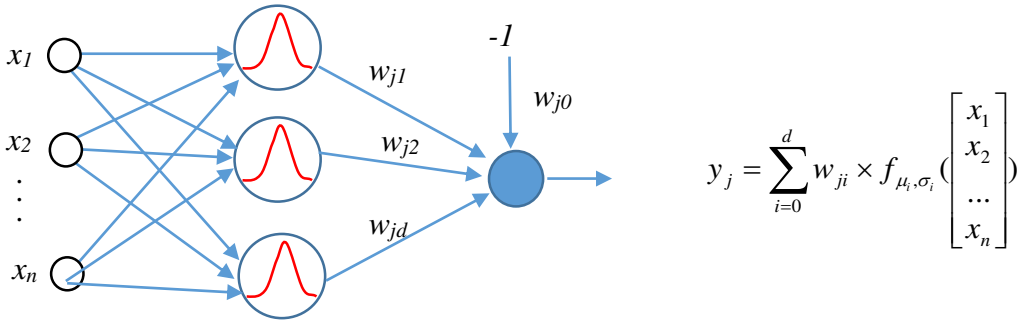


Figure 14 Calcul de la réponse d'un réseau RBF

Un réseau de neurones probabiliste (*PNN* : *Probabilistic Neural Network*) est un autre modèle de type *feedforward* très similaire au *RBF* (Specht 1990), cependant, certaines restrictions sont maintenues. Une gaussienne est utilisée comme fonction radiale (ou fonction noyau), le nombre de neurones dans la couche cachée est égale au nombre d'exemples d'apprentissage où chaque gaussienne est centrée sur un exemple et est représentatif de sa classe. Chaque neurone de la couche de sortie est représentatif d'une classe spécifique, la réponse du neurone est une probabilité d'appartenance de l'élément en entrée à cette classe.

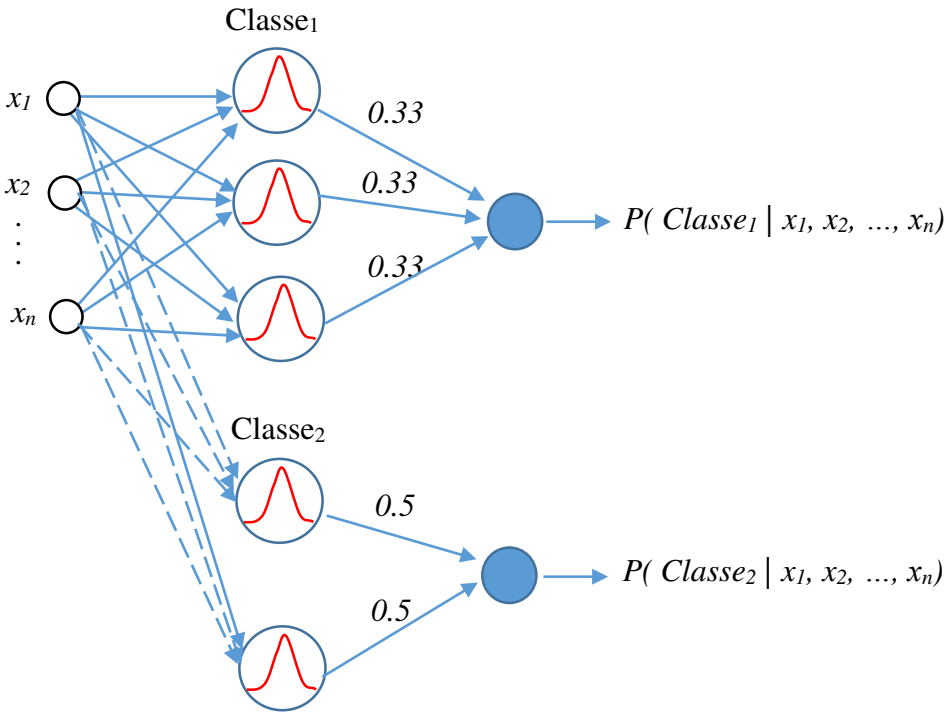


Figure 15 Exemple d'un réseau de neurone probabiliste

Avec 3 exemples d'apprentissage dans la classe1 et 2 en classe2

Chaque neurone de la couche de sortie n'est connecté qu'avec les fonctions radiales représentatives de classe concernée. Les poids synaptiques de la couche de sortie ne sont utilisés que pour normaliser la somme des fonctions radiales à un maximum de 1 afin d'obtenir une probabilité (Figure 15). Ce réseau ne représente pas d'inconvénients liés aux optimums locaux vu qu'il n'y a pas de paramètres à apprendre, cependant, pour de grandes bases d'apprentissage avec beaucoup d'instances le calcul de la réponse du réseau peut s'avérer très couteux en temps d'exécution.

II.3 Les modèles connexionnistes dynamiques à multiples couches

Ce qui caractérise les modèles connexionnistes dynamiques est la mise en place de connexions récurrentes et de mécanismes de mémorisation au niveau des couches et au niveau des neurones. Là où les modèles statiques fournissent toujours la même réponse pour le même signal en entrée, les modèles dynamiques fournissent une réponse qui dépend de l'entrée actuelle ainsi que les signaux présentés précédemment ce qui les rend adéquat pour le traitement des données séquentielles. Cette capacité de mémorisation est représentée soit de façon explicite via l'implémentation de mécanismes de mémorisation telle que les *TDNN* (Time Delay Neural network) ou les réseaux *GAMMA*, ou de façon implicite via des connexions récurrentes tel que les réseaux de *Elman*, de *Jordan* et les *LSTM* (Long Short Term Memory).

II.3.1 Les réseaux de Jordan et de Elman

L'un des premiers modèles de réseaux récurrents est aussi le plus simple est le réseau de Jordan (Jordan 1997). Ce réseau ressemble à un perceptron multicouches avec une vue sur le contexte antérieur immédiat, ce contexte correspond aux décisions antérieures du réseau (sorties). L'idée est de simplement recopier la couche de sortie du réseau dans une partie de la couche d'entrée (Figure 16).

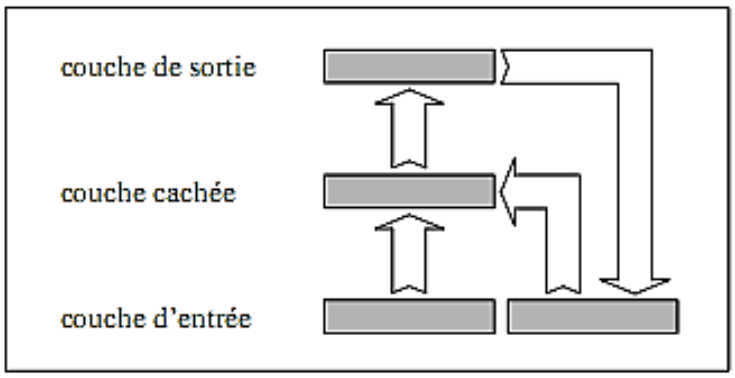


Figure 16 Réseau de Jordan

Le réseau de Elman (Elman 1990) recopie un vecteur de valeurs d'une des couches de niveau supérieur vers la couche d'entrée. Mais à la différence du modèle de Jordan, c'est la couche cachée qui est recopiée en entrée du réseau (Figure 17).

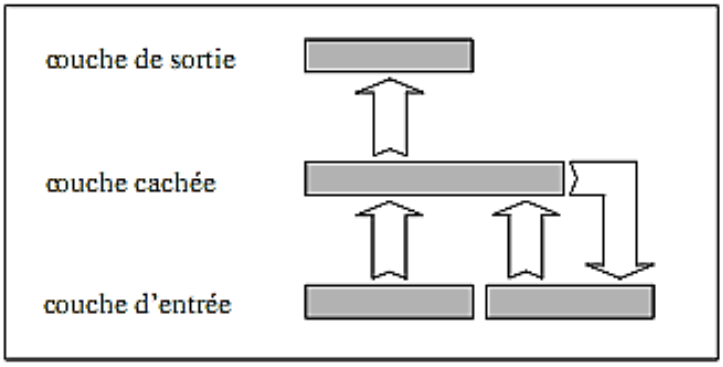


Figure 17 Réseau de Elman

Le réseau de Elman a été appliqué avec succès en reconnaissance de la parole et s'est montré bien plus performant que le perceptron multicouches pour des phonèmes dont la durée variait fortement.

II.3.2 Le Time Delay Neural Network

Ce réseau décrit dans (Waibel et al. 1989) fourni une représentation spatiale à la séquence traitée, l'entrée est composée de délais de retard intégrés. Cette couche d'entrée est responsable de la conversion spatio-temporelle du signal en entrée. Ce modèle dépend d'une taille de fenêtre qui détermine sa capacité de mémorisation, plus elle est grande, plus le réseau est capable de mémoriser des évènements dans le temps, mais le nombre de paramètres (poids synaptiques) augmente aussi avec les risques de la lenteur de l'apprentissage et le surapprentissage (Figure 18).

Chaque neurone de la couche cachée du *TDNN* est connecté à l'ensemble des lignes de la couche précédente, cependant il ne prend pas en considération tous les délais de chaque ligne. Donc, les neurones de la couche cachée font un masquage sur les neurones de la couche précédente afin de n'avoir une vue que sur une partie de l'information. Cela permet grâce à la notion de poids synaptiques partagés de spécialiser les neurones sur un seul type de traitement.

Le *TDNN* dispose d'une mémoire limitée dont la taille peut affecter les performances, cette taille est souvent fixée de façon empirique et handicap le réseau dans le traitement multi-échelles où une mémoire de taille dynamique est plus adéquate.

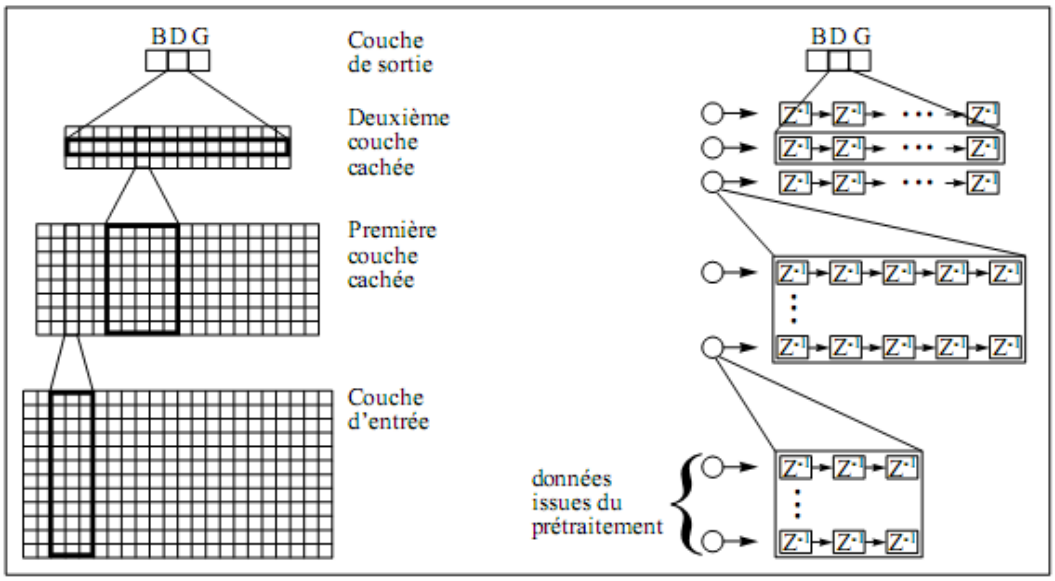


Figure 18 Architecture et schéma calculatoire d'un TDNN

II.3.3 Les réseaux *GAMMA*

La mémoire *GAMMA* est un concept introduit par Bert De Vries et Jose C.Principe en 1992 (de Vries & Principe 1992). L'idée est d'extraire d'un signal de taille N, une quantité d'informations pertinentes en compressant le signal dans une fenêtre de taille ($K \prec N$), par utilisation d'une succession de k filtres de type gamma. La quantité d'informations gardées en mémoire est contrôlée par un paramètre du filtre noté « u ». La stabilité du filtre a été démontrée dans l'intervalle]0, 2[(voir équation 14).

$$y_{i,t} = u\ y_{i-1,t} + (1-u)\ y_{i,t-1} \qquad 14$$

Pour $u=1$ le filtre gamma se comporte comme un délai simple et seul les k dernières valeurs du signal sont stockées en mémoire. Faire tendre u vers 0 a pour effet d'agrandir la résolution temporelle de la fenêtre de mémorisation, et donc davantage d'informations seront comprimées en mémoire. Les filtres gamma possèdent d'autres qualités, qui viennent du fait que plus le paramètre u tend vers 0 plus le filtre se comporte comme un filtre passe bas et n'admet donc pas de changements brusques des données. Cela lui donne la faculté de lisser les données, et donc, de filtrer un éventuel bruit. Ces caractéristiques ont encouragé l'introduction des filtres gamma à l'intérieur des neurones, et donc, de définir le neurone *GAMMA*. On peut considérer qu'il existe deux types de mémoires dans un neurone *GAMMA*. La première est statique et située au niveau des connexions entre les neurones et qui permet de conserver les connaissances définies lors de la phase d'apprentissage ne variant pas au cours de l'utilisation. Une deuxième mémoire, dynamique, est implantée dans le réseau au niveau des récurrences locales des neurones *GAMMA*. Ces récurrences vont permettre de conserver de l'information sur les

variations temporelles de la forme à analyser et va donc varier au cours de l'utilisation du réseau (voir Figure 19).

Pour le réseau *GAMMA*, les mémoires gamma présentent dans la première couche permettent de coder par adaptation du paramètre « *u* » lors de l'apprentissage des résolutions temporelles communes à toutes les catégories, quant aux neurones de la couche de sortie permettent de coder des résolutions spécifiques à une catégorie précise.

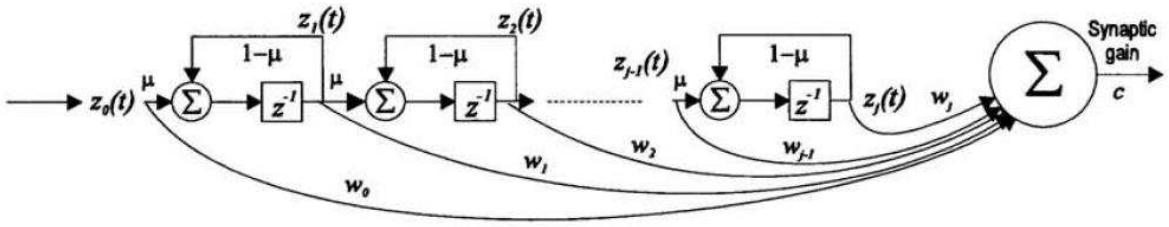


Figure 19 Calcul de la réponse d'un neurone avec une mémoire gamma (Lawrence et al. 1996)

La règle d'apprentissage utilisée pour l'adaptation des paramètres du réseau gamma, est la rétro propagation du gradient à travers le temps (*BPTT*) et l'apprentissage récurrent en temps réel (*RTRL*).

II.3.4 Le réseau *LSTM*

L'un des principaux problèmes dans les réseaux de neurones récurrents est la décadence exponentielle du gradient de l'erreur due à l'utilisation du rétropropagation du gradient de l'erreur dans le temps. Cela engendre une incapacité à apprendre correctement la corrélation qui existe entre les entrés et les erreurs commises pendant de longs retards temporels. La résolution temporelle se voit du coup limitée, ce qui engendre une défaillance sérieuse à apprendre des dépendances temporelles à très long terme naturellement existantes dans certains types de séquences telles que la parole.

Les réseaux *LSTM* (*Long Short Term Memory*) ont été proposés (Hochreiter & Schmidhuber 1997) afin de pallier à ce problème, dans ce réseau, on force un flot d'erreur constant le long d'une séquence ce qui permet au réseau de découvrir des dépendances temporelles à très long terme entre les entrés et l'erreur. Les réseaux *LSTM* parviennent facilement à reconnaître des grammaires à contexte lié tel que $a^n b^n c^n$ pour un très grand n, jusqu'à $n=2000$ à partir d'un ensemble d'apprentissage qui comprend des séquences de $1 < n < 30$.

Les réseaux *LSTM* sont des réseaux de neurones récurrents utilisant des cellules avec des connections internes qui représentent l'état de la mémoire du réseau. Ces cellules sont protégées par des portes multiplicatives non linéaires. La rétropropagation de l'erreur est effectuée de telle façon à éviter une décadence exponentielle de cette erreur. Un état non borné permet au réseau d'enregistrer des informations pendant de longues périodes de temps. Des portes sont utilisées afin d'aider le réseau à contrôler le flot d'informations qui traverse ses états internes. Les états

sont organisés en blocks, chacun ayant une porte d'entrée (*input gate*) qui permet au block d'ignorer provisoirement et sélectivement certaines activations, une porte de sortie (*output gate*) qui permet au block de se mettre sélectivement hors service (arrêter d'emmètre), afin de le protéger du retour (rétropropagation) de l'erreur, et une porte de réinitialisation (*forget gate*) qui permet au block d'effacer sa mémoire de manière sélective. Chaque porte possède sa propre activation dans l'intervalle [0,1]. En utilisant la décente du gradient pour optimiser les poids des connections à l'intérieur des blocks, un réseau *LSTM* va apprendre à contrôler un flux d'informations.

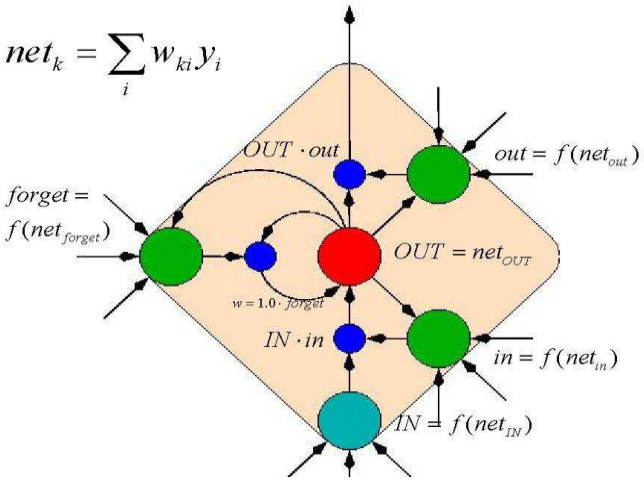


Figure 20 Calcule de la sortie d'un neurone LSTM (Hochreiter & Schmidhuber 1997)

L'apprentissage dans le réseau *LSTM* est un mélange entre la rétropropagation (pour les neurones de la couche de sortie), la rétropropagation dans le temps (*BPTT* : pour les portes de sorties), apprentissage récurrent en temps réel (*RTRL* : pour les cellules, les portes d'entrés et de réinitialisation).

II.4 L'apprentissage profond : *Deep Learning*

L'apprentissage profond est une nouvelle branche de recherche en apprentissage machine, qui a été introduite dans le but de rendre l'apprentissage automatique plus intelligent. Il s'agit d'apprendre plusieurs niveaux de représentation et d'abstraction ce qui permet une meilleure compréhension de la structure cachée des données telles que les images, la parole et le texte. Dans ce qui suit, on présente deux des architectures des plus citées dans la littérature.

II.4.1 *DBN* : Deep Belief Network

Un *DBN* est un réseau de neurones artificiel, profond, qui peut aussi se comporter comme un modèle génératif (Hinton et al. 2006). Dans ce réseau, l'apprentissage des couches de neurones

est fait de façon successif, chacune essayant de prédire au mieux la sortie de la couche qui la précède. Durant l'apprentissage, on suppose que chaque couche forme avec la couche précédente une machine de Boltzmann restreinte, les poids sont appris via un apprentissage non supervisé (Figure 21).

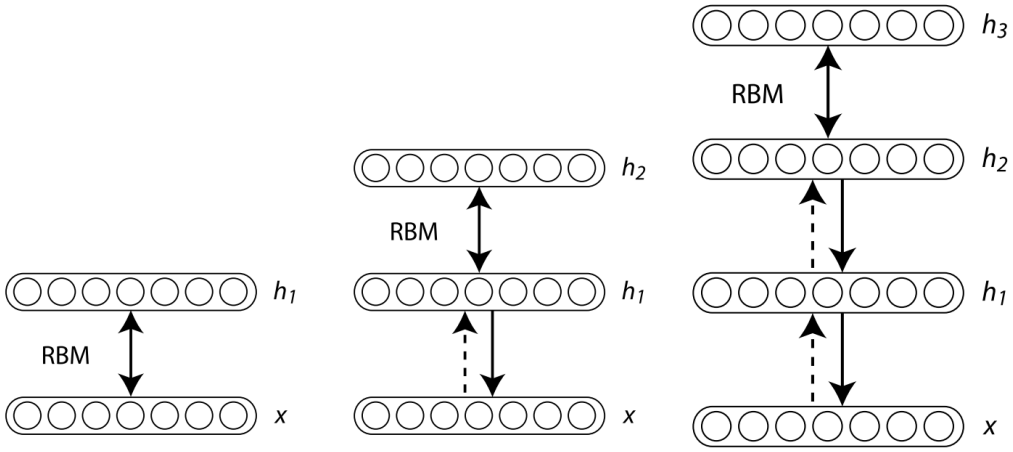


Figure 21 Apprentissage du Deep Belief Nework

II.4.2 Les réseaux de neurones à convolution

Le cœur des réseaux de neurones artificiels est une transformation affine : un vecteur en entrée qui est multiplié par une matrice produisant la réponse du réseau qui passe ensuite par une fonction d'activation non linéaire. Cependant, ce genre de calcule oblige l'aplatissement des représentations sous forme d'un vecteur, que ce soit un ensemble de caractéristiques, une image, du son ou une vidéo. Cet aplatissement détruit la structure intrinsèque de ses données et donc induit une perte d'informations qui peuvent être pertinentes.

Une image, un son ou une vidéo sont stockés sous forme de matrices multidimensionnelles avec une certaine relation d'ordre et où chaque axe offre une vue différente des données (ex : l'axe du temps dans le son ou les canaux gauche et droite du stéréo, les axes de largeur et de hauteur dans une image ainsi que les canaux des couleurs). La transformation affine réalisée par les réseaux de neurones artificiels classiques n'exploite pas la relation structurelle et topologique qui existe entre les différents axes. Cette structuration des données peut se révéler très pertinente, et d'où l'utilisation de transformations non destructrices comme la convolution multidimensionnelle est nécessaire.

Les réseaux de neurones convolutionnels (*CNN* : *Convolutional Neural Network*) sont des réseaux de neurones multicouches utilisés dans des domaines variés en reconnaissance des formes (Lecun et al. 1998), très réputés pour le peu de prétraitement nécessaire à leur fonctionnement, leur robustesse au bruit et leur excellentes capacités de généralisation. La première partie de l'architecture alterne entre couches de convolution (masques de convolution 2d) et couches d'agrégation (sous échantillonnage, *MaxPooling* …), et joue le rôle d'extracteur automatique de caractéristiques, tandis que la seconde partie représente un perceptron multicouche et joue le rôle d'un séparateur (Figure 22).

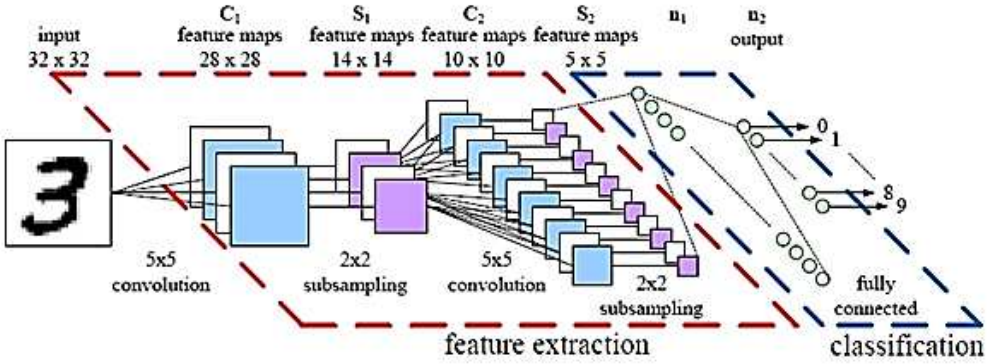


Figure 22 Schématisation du fonctionnement d'un CNN

Actuellement les *CNNs* comprennent beaucoup de paramètres, cela rend leur utilisation délicate sans une réelle compréhension de ces derniers. La sortie du *CNN* est affectée à la fois par l'entrée, par la succession et les types des couches du réseau, la taille des filtres (ou noyaux), le « zéro padding », les « strides » etc. Dans ce qui suit, on va définir les types de couches et les paramètres les plus communs d'un *CNN* utilisé dans une tâche de classification.

II.4.2.1 La couche de convolution

Une convolution discrète est une transformation linéaire qui préserve la notion d'ordre des données. Les paramètres des noyaux de convolution sont réutilisés à chaque fois sur une petite partie des données à plusieurs positions (voir Figure 23).

Image (en bleu) et résultat de la convolution (vert) **Filtre de convolution utilisé (noyau)**

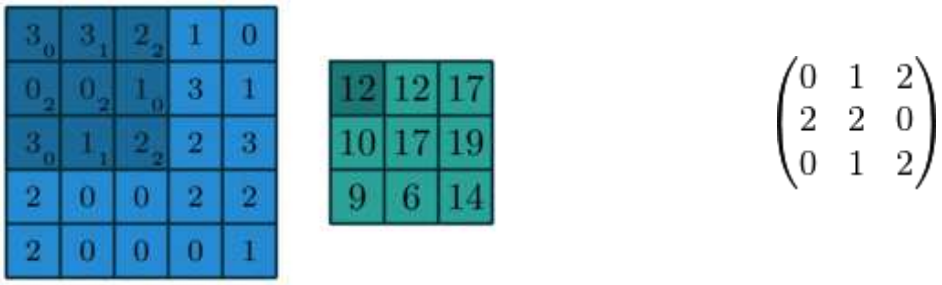


Figure 23 Exemple d'un filtre de convolution 2D

La couche de convolution du *CNN* peut comporter plusieurs filtres de convolution à partir desquels on obtient différentes vues des données en entrée (ou matrices de caractéristiques). Dans le cas où on a plusieurs matrices en entrée (ex : les canaux *RVB* d'une image), des filtres

3D sont utilisés dont le résultat équivaut l'application puis la sommation du résultat de plusieurs filtres 2D.

La couche de convolution du *CNN* comporte plusieurs paramètres dont :

- Le nombre de canaux en entrée n.
- Le nombre de filtres ou de noyaux m.
- La dimension des filtres $[k \times l]$.
- Le padding (nombre de zéros ajoutés dans les bords, voir Figure 24).
- Le stride (le pas de déplacement des filtres).
- La dilatation (pour un traitement multi-résolutions sur les données).

Le padding (par défaut : 0) est une technique utilisée en filtrage afin de traiter les bords des données, ceci permet aussi de contrôler la dimension des données en sortie du filtre.

Figure 24 Exemple d'un filtrage à convolution 2D avec un Padding=1

Par contre le stride (par défaut : 1) permet de régler le pas de glisse du filtre sur les données, en général la glisse se fait avec un pas de 1, il permet aussi de contrôler la dimension des données en sortie du filtre.

Introduit par Yu et Koltun (Yu & Koltun 2015), la dilatation (par défaut :1) permet de faire un filtrage multi-résolutions sur les données en ajoutant un paramètre de distance entre les éléments d'un filtre de convolution (voir Figure 25), ce paramètre permet aussi de contrôler la dimension des données en sortie du filtre (voir Figure 26 pour un récapitulatif des différents paramètres de la couche de convolution).

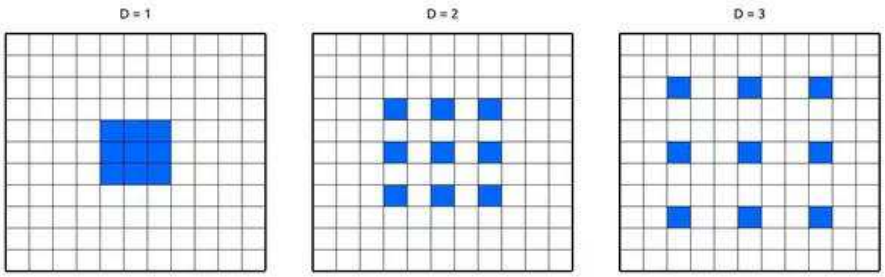


Figure 25 Exemple d'application d'un filtre à convolution 2D de taille [3×3] en variant la dilatation D

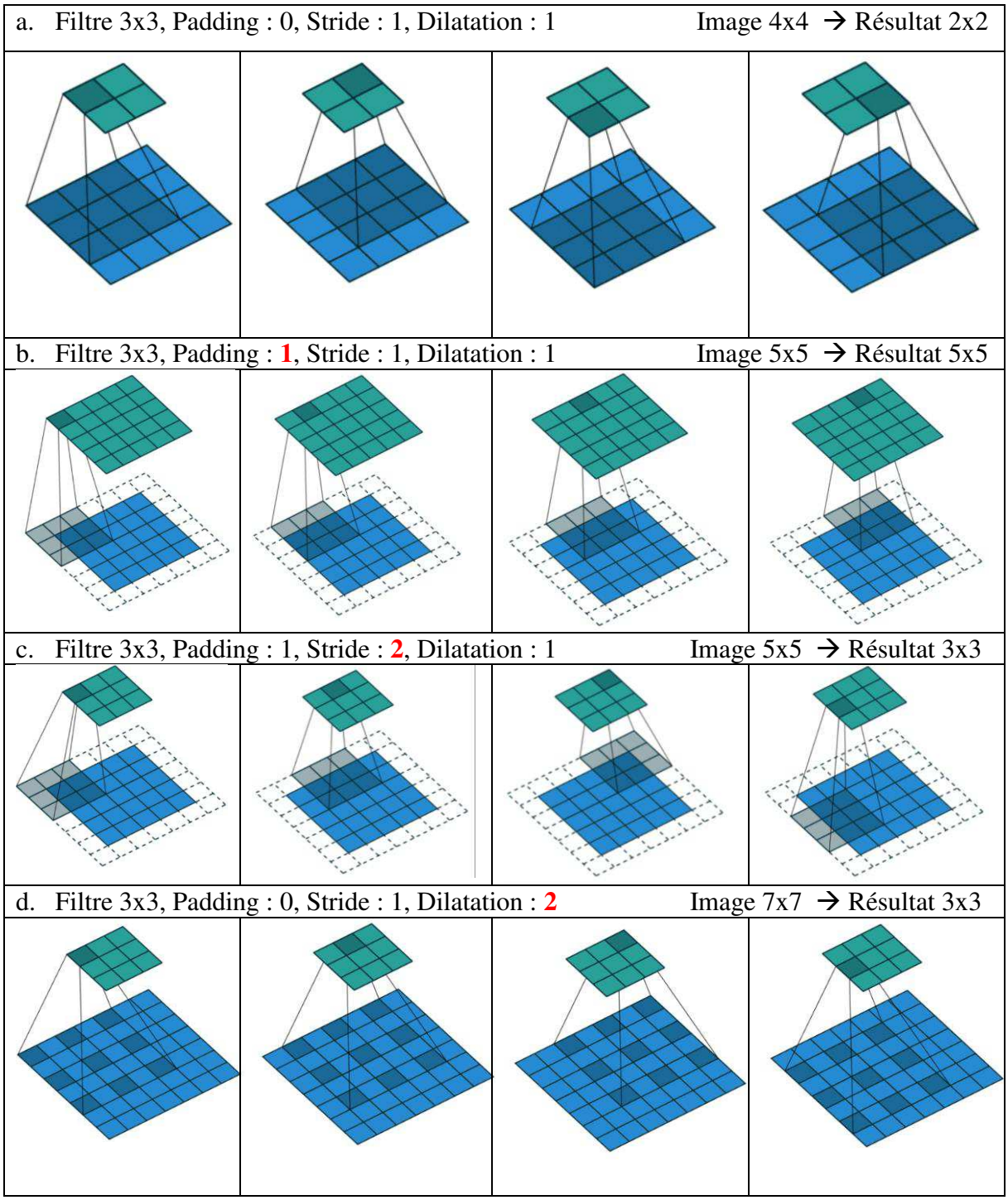


Figure 26 Récapitulatif du résultat du filtrage par convolution avec des paramètres variés.

II.4.2.2 Couche de sous échantillonnage

Il est commun d'insérer une couche de sous échantillonnage (ou d'agrégation) entre deux couches de convolution successives dans un *CNN*, sa fonction est de réduire progressivement la dimension spatiale des caractéristiques à travers les couches afin de réduire le nombre de

paramètres et les calculs du réseau, et donc de réduire par la même occasion le risque de surapprentissage. Cette couche traite chaque canal des données en entrée séparément pour produire une matrice réduite selon la taille du filtre du sous échantillonnage.

Le filtre le plus utilisé actuellement est le filtre *Max Pooling* qui est un filtre glissant 2D de taille *[k×l]* avec un paramètre stride supérieur à 1 et qui réalise une simple opération du calcul du maximum des données couvertes par celui-ci. Par exemple, un filtre *Max Pooling* de dimension de *[2×2]* avec un stride de *2* permet de réduire les données de 75% (voir Figure 27).

Dans l'état de l'art on peut trouver aussi le filtre *Average Pooling* qui calcul la moyenne des éléments au lieu du maximum, cependant, il a été prouvé qu'en pratique le filtre *Max Pooling* donnait de meilleurs résultats.

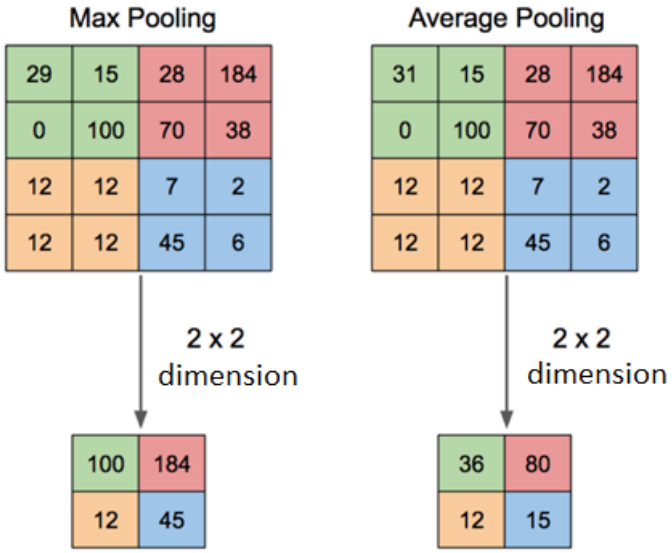
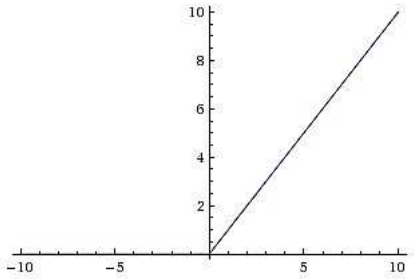


Figure 27 Exemple de sous échantillonnages par des filtres Max et Average Pooling de dimension de [2×2].

II.4.2.3 Couche d'activation

Cette couche applique une fonction d'activation non linéaire sur l'entrée de type *ReLU* (Rectified Linear Units, voir Figure 28) ou rarement une sigmoïde.



$$f(x) = \max(x, 0)$$

Figure 28 Fonction d'activation ReLU

Actuellement la plupart des réseaux en *Deep Learning* comportent des couches d'activation non linaires *ReLU* au lieu d'une sigmoid/tangente hyperbolique, cette fonction a une sortie nulle pour une entrée négative, sinon elle fait resortir l'entrée telle quelle. L'utilisation de cette fonction d'activation comporte plusieurs avantages :

1. Le calcule de cette activation ainsi que sa dérivée qui équivaut 1 est très rapide ce qui réduit le temps d'apprentissage et de simulation du réseau.
2. La dérivée de la fonction sigmoïde vaut au maximum un *0.25*, propagée sur *n* couches induit un gradient (au mieu de 0.25^n) qui tend vers 0 pour un grand nombre de couches, ce problème est très connu en connexionnisme sous l'apellation de l'évanouissement du gradient de l'erreur qui implique un non apprentissage (ou un apprentissage très lent) des couches en profondeur. Ce problème ne se manifeste pas avec la fonction ReLu, vu que sa dérivée vaut 1.
3. Offrir une représentation creuse (*sparse*) des caractéristiques, en transformant les entrées négatives en 0, cela permet une extraction plus sélective des caractéristiques.

II.4.2.4 Couche Dropout

Le *Dropout* est une technique de régularisation dans les modèles de réseaux de neurones proposée par Srivastava et ses collègues (Srivastava et al. 2014) afin de prévenir le surapprentissage. La couche *Dropout* implémente cette technique avec comme paramètre le pourcentage du *Dropout*, cependant cette couche n'est utilisée que dans la phase d'apprentissage et est ignorée dans la phase de test du réseau.

Cette technique consiste à sélectionner aléatoirement à chaque itération un pourcentage (pourcentage *du Dropout*) de neurones à ignorer durant l'apprentissage, ce qui veut dire que leur contribution dans le calcul de la passe avant est temporairement supprimée, et aucune mise à jour des poids n'y est appliquée dans la passe arrière.

Durant la phase d'apprentissage, les poids du réseau s'adaptent afin de spécialiser chaque neurone dans l'extraction d'une caractéristique spécifique. Cette spécialisation résulte en une trop forte dépendance entre les neurones voisins afin de fournir une réponse correcte, cette dépendance induit un modèle fragile trop spécialisé dans les données apprises et lui fait perdre sa capacité à généraliser, surtout en *Deep Learning* où le nombre des paramètres du réseau est très grand. Le fait d'ignorer certains neurones du réseau durant la phase d'apprentissage implique d'ignorer leurs contributions dans la réponse du réseau, cela oblige les autres neurones à ne pas dépendre les uns des autres en compensant la perte de cette contribution par l'apprentissage de caractéristiques plus robustes. Il en résulte de multiples représentations internes et indépendantes apprises par les neurones. Vu qu'à chaque itération, le réseau génère aléatoirement une architecture différente, on obtient une réponse plus robuste obtenue à partir de l'agrégation de plusieurs classifieurs différents au sein du même réseau. Le réseau devient moins sensible à des poids spécifiques de neurones et est capable d'une meilleure généralisation réduisant le risque de surapprentissage.

II.4.2.5 Couche de décision

Connue aussi sous l'appellation couche *Softmax*, Cette couche normalise la sortie du réseau sous forme de probabilités d'appartenance de l'image en entrée à chacune des classes de sortie.

II.4.3 Le DAG-CNN

Dans un *CNN* classique les données circulent de couche en couche, et donc la réponse de chaque couche ne dépend que des données de la couche précédente. Proposé par Yang et Ramanan (Yang & Ramanan 2015) le *DAG-CNN* (*Directed Acyclic Graph CNN*) est réseau de neurones à convolution dont les connexions sont sous la forme d'un graphe acyclique orienté. Dans cette configuration, la couche de sortie a accès aux réponses de plusieurs couches précédentes, cela permet de raisonner en termes de caractéristiques de bas, moyen et de haut niveau durant la classification. Cette architecture à multiples niveaux, a accès à la fois à des caractéristiques de haut niveau qui permettent de différencier des images trop différentes (reconnaissance grossière), et a accès en même temps à des caractéristiques de bas niveau qui permettent de différencier des images trop ressemblantes (reconnaissance fine, voir Figure 29). En utilisant cette architecture, l'étude de Yang et Ramanan a montré une réduction importante de l'erreur de classification : de 23.9% et de 9.5% sur les benchmarks *MIT67* et *Scene15* par rapport aux meilleurs taux de l'état de l'art.

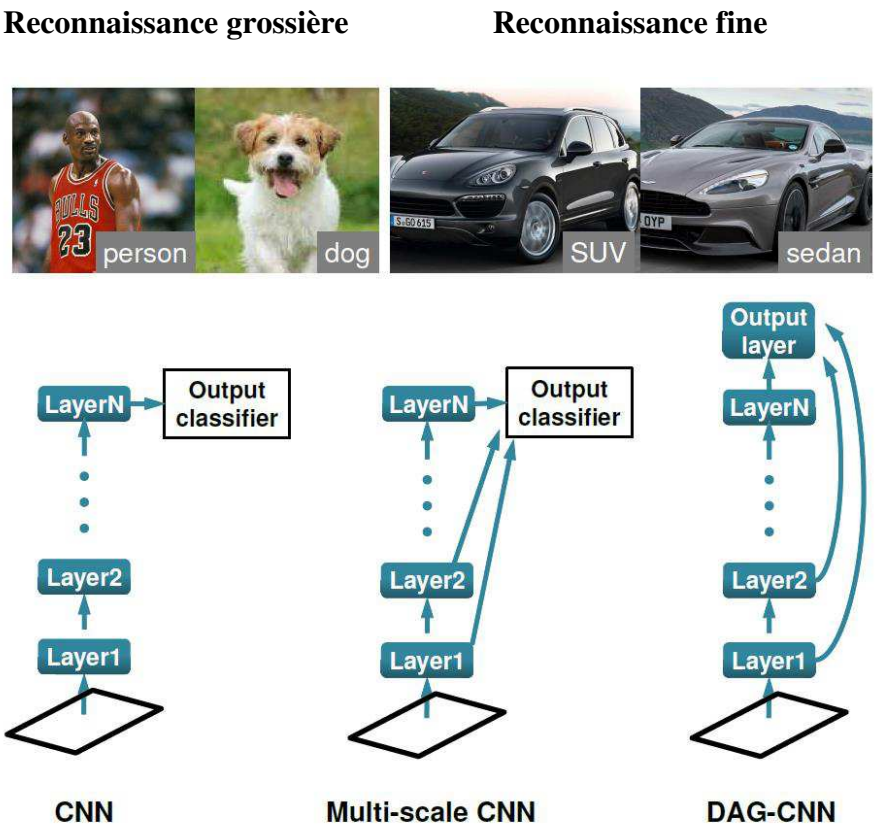


Figure 29 Illustration du principe des DAG-CNN (Yang & Ramanan 2015)

II.4.4 Les modèles pré-entrainés et le transfert de l'apprentissage

Bien que les performances du matériel informatique est en constante croissance, avoir du matériel performant est un investissement qui n'est pas à la portée de tout le monde. En *Deep Learning* l'ajout d'une couche supplémentaire résulte en plusieurs heures d'apprentissage additionnelles ou la nécessité de ressources matérielles supplémentaires. Le transfert de l'apprentissage est un concept qui permet de palier à ce problème en permettant de résoudre des problèmes complexes en *Deep Learning* en un temps insignifiant et sans investissement en matériel. De la même manière qu'un enseignant est capable de partager des années d'expériences et d'apprentissage avec ses étudiants, le transfert de l'apprentissage en *Deep Learning* permet de réutiliser avec de petites modifications des réseaux performants déjà entrainés sur des problèmes connexes au problème traité. Le transfert de l'apprentissage permet de gagner des heures d'apprentissage en extraction des caractéristiques (déjà acquise par les réseaux pré-entrainés) et de se concentrer sur la phase de la classification au lieu de refaire tout un apprentissage de zéro.

Un modèle pré-entrainé représente des heures d'apprentissage dans un matériel dédié sur d'énormes benchmarks, les premières couches du réseau s'occupent de l'extraction des caractéristiques des données tandis que les dernières couches s'occupent de la séparation des classes. La réutilisation de ce modèle dans un domaine connexe (qui ne comporte pas forcément les mêmes classes) est une tâche facile, vu que les premières couches du modèle ont déjà appris à extraire les caractéristiques de base, il suffit de ré-entrainer seulement les couches discriminantes (généralement la dernière) sur les nouvelles données. Il est même possible d'injecter les caractéristiques extraites à partir du modèle pré-entrainé dans un classifieur séparé comme un *SVM* ou un *K-NN*. Cela permet un gain considérable en temps et en performances.

Les réseaux pré-entrainés sur la fameuse base d'images *ImageNet*, comportant 1.2 millions d'images réparties sur 1000 catégories, ont montré de très grandes capacités à se généraliser à des images en dors de cette base, parmi les plus connus on trouve : *Caffe*, *AlexNet*, *GoogLeNet* et *ResNet*. Ces réseaux peuvent être utilisés comme extracteurs de caractéristiques pour des images, ou afin de faire une initialisation intelligente des poids d'un nouveau réseau, on peut éventuellement les remodeler pour convenir à un problème similaire.

- *Caffe* (Jia et al. 2014) et *AlexNet* (Krizhevsky et al. 2012) décrivent des *CNN* pré-entrainés sur la base *ImageNet* alternant 21 couches de type convolution, *ReLU* et *MaxPooling*, la dernière couche (la couche séparatrice) comporte 4096 caractéristiques en entrée et 1000 classes en sortie. Spécialement *AlexNet* comprend 60 millions de paramètres répartis sur 650.000 neurones, entrainé avec des couches *Dropout* afin de réduire le surapprentissage. Le réseau *AlexNet* a obtenu un taux d'erreur de 17% dans la compétition *ImageNet LSVRC-2010* sur le top-5 des réponses du réseau. Une variante améliorée c'est classée première dans la compétition *ILSVRC-2012* avec un taux d'erreur du top-5 des réponses du réseau de 15.3%.

- *GoogLeNet* (Szegedy et al. 2015) décrit un *DAG-CNN* qui alterne 152 couches de type convolution, *MaxPooling*, normalisation, concaténation et *ReLU* puis une couche sépartrice comprenant 1024 caractéristiques en entrée et 1000 classes en sortie. Ce réseau a réussi à remporter la première place dans la compétition *ILSVRC 2014* en innovant avec le développement d'un nouveau module nommé module d'inception. Ce module a permis de réduire considérablement le nombre de paramètres du réseau *CNN* avec 4 millions de paramètres seulement pour *GoogLeNet* comparés aux 60 millions de paramètres de *AlexNet*.

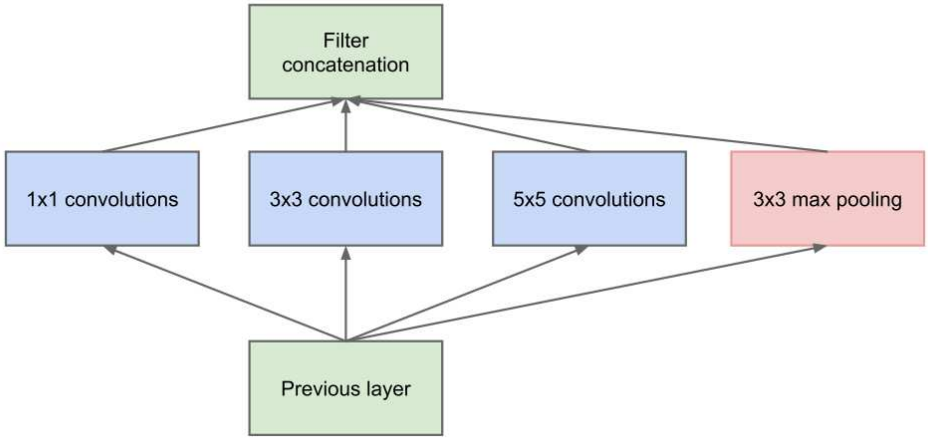


Figure 30 Illustration du principe du module d'inception (Szegedy et al. 2015)

Le module d'inception agit comme de multiples filtres de convolution et de sous échantillonnage appliqués sur les mêmes données en entrée, les résultats obtenus sont concaténés ce qui permet au réseau de faire une extraction des caractéristiques à de multiples niveaux (voir Figure 30).

- *ResNet* (He et al. 2016) décrit un *DAG-CNN* qui alterne 175 couches de type convolution, *MaxPooling*, normalisation et *ReLU* puis une couche séparatrice comprenant 2048 caractéristiques en entrée et 1000 classes en sortie. Ce réseau a remporté la première place dans les compétitions *ILSVRC & COCO 2015* avec un taux d'erreur de 3.57% sur la base *ImageNet*, et mettant en avant l'apprentissage résiduelle qui facilite l'apprentissage de réseaux de profondeurs plus élevés. Les couches du réseau sont reformulées sous la forme de fonctions résiduelles avec des références aux couches précédentes (voir Figure 31).

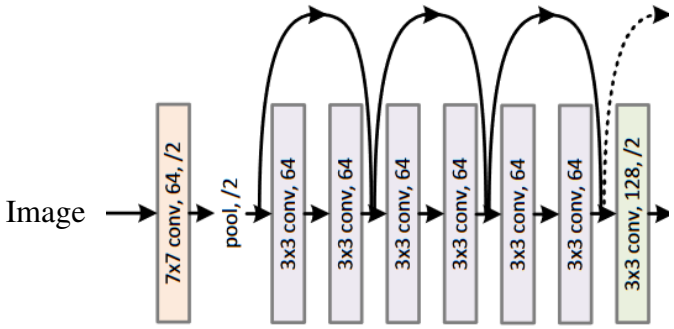


Figure 31 Illustration de l'architecture d'un réseau résiduel (He et al. 2016)

II.5 La classification sous contraintes d'optimalité : les Séparateurs à Vaste Marge

En classification, il existe plusieurs séparateurs qui permettent de séparer parfaitement entre deux groupes d'échantillons (Figure 32). Cependant, ces séparateurs ne sont pas tous adéquats et engendrent un risque de surapprentissage. Ajouter des contraintes sur le choix du séparateur, en choisissant le séparateur le plus éloigné par rapport aux échantillons (principe de la maximisation de la marge) permet de minimiser le risque de surapprentissage qui handicap les techniques connexionnistes classiques.

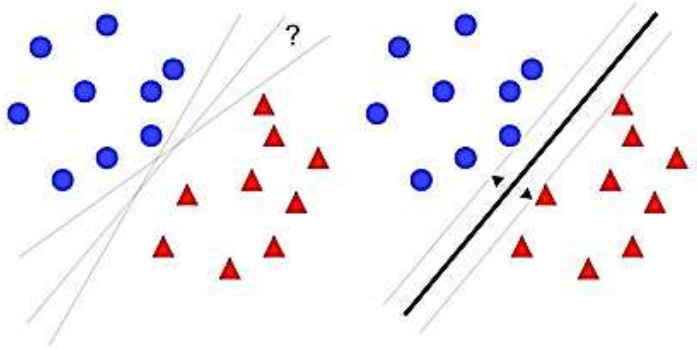


Figure 32 Illustration du principe de la maximisation de la marge

Les *SVMs* (Séparateurs à Vaste Marge) sont des techniques d'apprentissage machine qui s'inspirent des approches utilisées dans le domaine de l'apprentissage statistique, largement développés par Vapnik et ses collaborateurs. Les *SVMs* reprennent l'idée énoncée plus haut, en transformant le problème d'apprentissage en un problème d'optimisation quadratique sous contraintes. Le principe est donc de trouver un séparateur, ou une fonction de discrimination, dont la capacité de généralisation est la plus grande possible.

Plusieurs travaux ont montré l'efficacité des *SVMs* en classification (Amayri & Bouguila 2010), (Joachims 2002), (Shi et al. 2010), (Tan et al. 2014). Les *SVMs* sont très performants sur des données de grande dimensionnalité telles que les données textuelles ou génétiques (Brown

et al. 2000), (Furey et al. 2000). A la base, les *SVMs* sont utilisés en classification binaire, cependant, beaucoup de problèmes en classification sont multi-classes, une extension des *SVMs* au cas multi-classes est possible via l'utilisation de modes d'apprentissage tels que le « un contre tous » ou le « un contre un ». Les *SVMs* minimisent le risque de surapprentissage en maximisant la distance entre le séparateur linéaire et les vecteurs de support. La fonction de décision d'un *SVM* linéaire est formulée par l'équation 15, où w est le vecteur des poids, x est l'instance à classifier, b un scalaire et *sgn* la fonction signe.

$$f(x) = \mathrm{sgn}(w^T x + b) \tag{15}$$

Ayant un ensemble d'instances avec des labels $(x_i, y_i), i = 1, ..., n, x_i \in R^n, y_i \in \{-1, +1\}$, l'apprentissage des *SVM* est reformulé sous la forme d'un problème d'optimization quadratique à multiples variables (équation 16), ξ est une fonction de perte qui mesure la quantité des données males classées, les fonctions de pertes les plus utilisées sont les fonctions *L1-SVM* et *L2-SVM* (équation 17). $C > 0$ est un paramètre de pénalité qui fixe un compromi entre la largeur de la marge et les erreurs de classification. Il est possible d'ajouter un biais b afin d'améliorer la qualité de la classification en augmentant la dimension du vecteur w et les vecteurs x_i tel que : $w^T \leftarrow [w^T, b]$ et $x_i^T \leftarrow [x_i^T, 1]$.

$$\min_w \left\{ \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{n} \xi(w; x_i, y_i) \right\} \tag{16}$$

$$\xi(w; x_i, y_i) = \begin{cases} \max(1 - y_i w^T x_i, 0) & L1 - SVM \\ \max(1 - y_i w^T x_i, 0)^2 & L2 - SVM \end{cases} \tag{17}$$

Les *SVMs* peuvent être étendus au cas non linéaire en introduisant des fonctions noyaux. Parmi les noyaux les plus utilisés on trouve le noyau *RBF* (*Radial Basis Function*) et le noyau polynomial. Ces noyau permettent de projeter des données non linéairement séparables dans un espace différent où la séparation linéaire devient possible.

Pour un *SVM* linéaire l'optimisation des équations ci-dessus peut se faire selon différentes formulations, les formulations les plus communes sont la régulation *L1* et la régulation *L2*.

II.5.1 Régularisation *L2* avec fonction de coût *L1* et *L2* en classification

Dans une formulation régularisée *L2* avec une fonction de perte *L1* on résout le problème primal suivant :

$$\min_{w}\left\{\frac{1}{2}w^T w + C\sum_{i=1}^{n}(\max(1-y_i w^T x_i,0))\right\} \quad 18$$

Pour une fonction de perte *L2* on résout le problème primal suivant :

$$\min_{w}\left\{\frac{1}{2}w^T w + C\sum_{i=1}^{n}(\max(1-y_i w^T x_i,0))^2\right\} \quad 19$$

Leur forme duale est :

$$\min_{\alpha}\left\{\frac{1}{2}\alpha^T \bar{Q}\alpha - e^T\alpha\right\} \quad avec \quad 0 \le a_i \le U, i = 1..n \quad 20$$

Où e est une vecteur de 1, $\bar{Q} = Q + D$, D est une matrice diagonale, et $Q_{ij} = y_i y_j x_i^T x_j$. Pour la fonction de perte *L1*, $U=C$ et $D_{ii}=0$ pour tout i et pour la fonction I $U=\infty$ et $D_{ii}=1/(2C)$ pour tout i.

II.5.2 Régularisation L1 ave fonction de coût L2 en classification

La régularisation *L1* génère des vecteurs w creux en résolvant le problème primal suivant :

$$\min_{w}\left\{\|w\|_1 + C\sum_{i=1}^{n}(\max(1-y_i w^T x_i,0))^2\right\} \quad 21$$

Où $\|w\|_1$ dénote la norme 1 de w.

II.6 Conclusion

Du point de vu mathématique, la fonction de décision des *SVMs* linaires est équivalente à celle d'un perceptron simple, celle d'un *SVM* avec un noyau gaussien est très similaire à un réseau *RBF*, ainsi qu'un *SVM* avec un noyau sigmoïde est comparable à un perceptron multicouches. Il est plus judicieux de parler de leurs différences, qui se situent principalement dans la phase d'apprentissage. Les règles utilisées en connexionnisme sont basées sur la rétropropagation du gradient de l'erreur ou l'apprentissage Hebbien et mettent à jours itérativement tous les paramètres du réseau. L'apprentissage dans les *SVMs* par contre est transformé en un problème d'optimisation quadratique sous contraintes de maximisation de la marge pour la couche de décision. Les paramètres des noyaux sont fixés au début de l'apprentissage (ex : les vecteurs de support, l'écart type du noyau gaussien, l'ordre du noyau polynomial …etc.).

On peut dire que les *SVMs* sont très similaires aux modèles connexionnistes du point de vue architecture, mais issues d'inspirations différentes, l'apprentissage des *SVMs* minimise le risque de surapprentissage, n'est pas sujet aux minimas locaux, est en général nettement plus rapide et ne dépend que de peu de paramètres à initialiser.

Les avantages des *SVMs* en font un concurrent très performant aux techniques connexionnistes, dans les chapitres suivants, on utilisera les *SVMs* afin de guider l'apprentissage et la génération de représentations discriminantes pour des données complexes combinés avec une technique d'optimisation inspirée du mécanisme de l'évolution des espèces.

CHAPITRE 3

Génération de représentations optimisées via une approche mimétique parallèle

III.1 Introduction

Les algorithmes évolutionnaires (*AE*) sont des approches stochastiques d'optimisation inspirées de la théorie de l'évolution naturelle de Darwin. Ces algorithmes ont suscités de nombreux travaux notamment ceux de John H. Holland et David E. Goldberg sur les algorithmes génétiques. Faire émerger un comportement intelligent à partir de l'aléatoire est une idée qui a fasciné beaucoup de chercheurs, cela a donné naissance à de multitudes de familles d'algorithmes évolutionnaires dont les algorithmes génétiques (Goldberg & Holland 1988) (Goldberg 1989), les stratégies d'évolution (Rechenberg 1989), la programmation génétique (Koza 1994) et les algorithmes à estimation de distribution (Larrañaga & Lozano 2002). Certains algorithmes très similaires peuvent être inclus dans cette famille tels que les essaims particules (Poli et al. 2007), les colonies de fourmis (Pedemonte et al. 2011) et les colonies d'abeilles artificielles (Narasimhan 2009). Ces techniques reposent sur le même principe, faire évoluer une population d'individus générés aléatoirement représentant chacun une solution potentielle au problème, grâce à une fonction de fitness qui permet d'évaluer la qualité de chaque individu en plus d'opérateurs de modification qui permettent de remplacer les individus de la population actuelle par des individus dits plus fort. Le principal avantage de ces approches est qu'ils permettent d'atteindre de meilleurs minimas que les approches de recherche locales et ceci en formulant le problème sous forme d'une simple fonction à maximiser/minimiser sans connaissances à priori du problème à traiter et sans contraintes de convergence ou de différentiabilité sur la fonction de fitness. Le principal défaut de ces approches est le cout du calcul qui peut être très lourd surtout dans le cas d'une fonction de fitness très complexe. Cependant, il est très commun de combiner un algorithme évolutionnaire avec une approche de recherche locale afin de le guider et d'accélérer la convergence (paradigme mimétiques). Dans le même contexte, de nombreuses recherches portent sur l'élaboration de modèles pour la parallélisation de ces algorithmes.

Dans ce chapitre, on aborde brièvement le principe des approches évolutionnaires tout mettant l'accent sur leurs limites, des solutions sont proposées via le mimétisme et la parallélisation. Dans la suite du chapitre, on utilise ces solutions afin de modéliser une technique d'optimisation de la représentation des données complexes via la sélection et la pondération d'attributs, cette technique sera testée et évaluée dans le chapitre suivant sur la classification des documents Web et des images aériennes.

III.2 Terminologie

Dans cette section, on présente le vocabulaire essentiel à la compréhension du mécanisme des approches évolutionnaire :

- Individus : des solutions possibles au problème d'optimisation traité.
- Population : un ensemble fini d'individus.
- Fonction de fitness : la fonction à optimiser (à minimiser ou maximiser).
- Evaluation : calcul de la fitness de chaque individu de la population.
- Génération : la population en une itération du processus évolutionnaire.

- Opérateurs de modification : mécanisme pour générer de nouveaux individus (tel que le croisement et la mutation).
- Sélection : le mécanisme de choix des individus selon leur fitness.
- Remplacement : remplacer les individus faibles par de nouveaux individus.

III.3 Principe des approches évolutionnaires

Le mécanisme évolutionnaire (Figure 33) est un processus itératif qui commence par :

1. la génération de la population POP_0 initiale composée d'un nombre fini d'individus avec une distribution uniforme.
2. Evaluation de la population.
3. Si le critère d'arrêt est atteint alors s'arrêter.
4. Sélection d'un sous ensemble d'individus.
5. Application des opérateurs de modification pour créer de nouveaux individus.
6. Remplacer certains individus de la population par ceux créés en 5.
7. Revenir à l'étape 2.

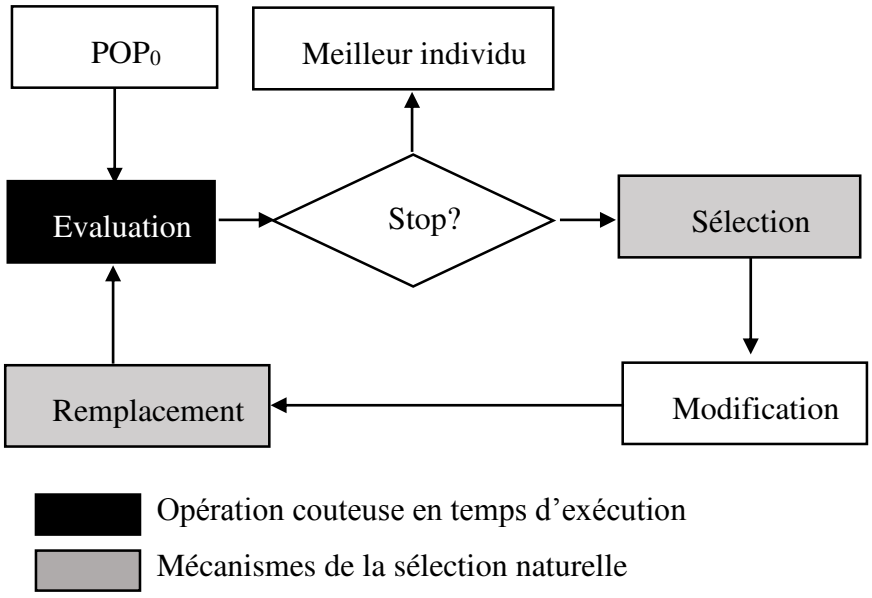


Figure 33 Les étapes du processus évolutionnaire

On note que le coût d'une application évolutionnaire dépend surtout de la complexité de sa fonction de fitness et donc de l'étape d'évaluation. A chaque étape, un compromis entre la diversité de la population et la convergence est maintenu afin d'affiner les individus sans pénaliser l'exploration de l'espace de recherche et ainsi éviter une convergence prématurée.

Dans un algorithme évolutionnaire, ne garder que les meilleurs individus dans la population risque à la longue de pénaliser la diversité de celle-ci, et donc, un espace de recherche plus réduit et plus de probabilité de stagner dans un optimum local. Il est plus judicieux de garder du bruit au sein de la population en gardant aussi quelques individus faibles, cette technique permet une meilleure diversité et un espace de recherche plus large. Ceci peut être réalisé en utilisant une stratégie de sélection adéquate.

III.4 Stratégies de sélection

L'étape de la sélection permet de simuler le principe de la sélection naturelle dans la nature, en favorisant les meilleurs individus à survivre, et donc, de générer une nouvelle génération qui hérite des caractéristiques des individus les plus forts. Ainsi, les individus les plus faibles ont une grande probabilité d'être remplacés, dans cette étape, il est possible de faire des remplacements au niveau des parents, ou bien ou niveau de l'ensemble des parents et des fils. Il existe plusieurs stratégies de sélection qui ont été proposées dans divers travaux sur les algorithmes évolutionnaires dont les plus connues sont la sélection par roulette et par tournoi.

III.4.1 La sélection par roulette

C'est une technique stochastique qui consiste à :

1. Affecter à chaque individu une probabilité de sélection en fonction de sa fitness, les meilleurs individus auront de plus grandes probabilités d'être sélectionnés.
2. Une simulation d'une lancée d'une boule sur une roulette dont les parts sont représentées par les probabilités de sélection des individus est effectuée.
3. La boule s'arrête sur l'individu à sélectionner.
4. Réitérer à partir de 2.

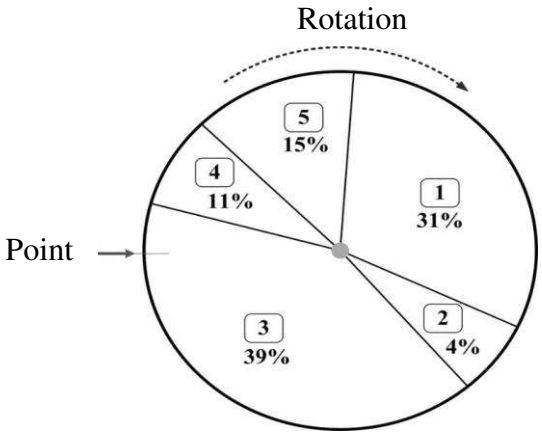


Figure 34 La sélection par roulette

Cette technique présente néanmoins quelques problèmes de réglage, par exemple un individu qui a une fitness largement supérieure aux autres individus aura une grande probabilité d'être

toujours sélectionné. Ce cas, risque de pénaliser fortement la diversité de la population et augmenter le risque de convergence prématurée. En même temps, des individus dont la fitness est proche vont avoir les mêmes probabilités de sélection, les meilleurs individus peuvent ne pas être sélectionnés. Certaines solutions ont été proposées dans la littérature, notamment donner des probabilités de sélection aux individus selon leurs classements et non selon leurs fitness.

III.4.2 La sélection par tournoi

Dans cette stratégie, la taille d'un sous ensemble *T* est fixée puis :

1. Choisir *T* individus aléatoirement à partir de notre population.
2. Un tournoi entre les *T* individus est joué et le meilleur individu est sélectionné.
3. Réitérer à partir de 1.

Dans cette stratégie le choix du paramètre *T* est important, plus *T* est grand plus la probabilité de sélectionner les meilleurs individus augmente, avec un T petit la chance de sélectionner des individus moins bon augmente et avec elle la diversité de la population.

III.5 Opérateurs de modification

Il s'agit de mécanismes qui permettent de combiner ou d'altérer les caractéristiques des individus afin de générer une nouvelle génération héritant des caractéristiques de la génération précédente. En pratique, il s'agit de chercher dans un grand espace de recherche, des solutions voisines meilleures que celles qu'on a déjà dans notre population. Plusieurs opérateurs de modification ont été abordés dans la littérature, dont les plus communs sont le croisement et la mutation inspirés de la génétique des espèces.

III.5.1 Le croisement

Il s'agit de combiner les caractéristiques de plusieurs individus. On peut du coup espérer en combinant des solutions performantes obtenir des solutions encore plus performantes. Dans un algorithme génétique, l'application du croisement peut se faire selon plusieurs critères de choix :

- Le nombre d'individus parents à choisir.
- Combien d'individus fils seront générés par ce croisement.
- Les positions d'échange des caractéristiques.
- Le nombre de positions.

La stratégie la plus simple, et sans doute la plus proche de l'inspiration biologique est décrite comme suit :

1. Choisir aléatoirement deux individus parents.
2. Combiner les deux parents selon le nombre de positions de croisement.
3. Générer deux individus fils issus de ce croisement

Cette stratégie permet de conserver une taille de population fixe, en remplaçant tous les parents par les enfants.

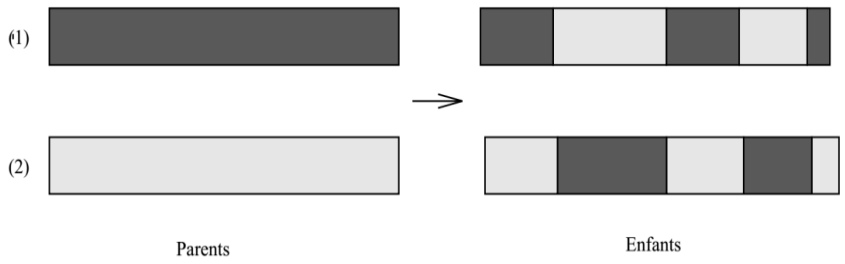


Figure 35 Croisement à multiples positions

III.5.2 La mutation

La mutation est un concept issu de la biologie, qui indique une altération dans une séquence ADN d'un individu, elle se produit généralement suite à des facteurs externes (radiation, produits chimiques …). Cette mutation peut avoir un effet visible ou invisible sur l'individu qui peut être négatif ou positif. Ce phénomène est complètement aléatoire, il est nécessaire d'effectuer plusieurs mutations avant d'obtenir une qui soit favorable.

En pratique, dans un algorithme génétique on essaie de reproduire ce mécanisme en définissant un opérateur de mutation qui permet d'introduire du bruit favorisant la diversité qui est nécessaire pour une exploration approfondie de l'espace de recherche. Il permet d'introduire des valeurs de caractéristiques qui n'étaient pas présentes dans la population initiale et réduit le risque d'une convergence prématurée.

Pour utiliser l'opérateur de mutation, on définit un taux de mutation qui indique le pourcentage des individus qui subissent une mutation pour chaque génération. Le taux de mutation peut s'appliquer sur les individus, une variante consiste à l'appliquer sur les positions (locus) dans un chromosome. Il est aussi possible de faire varier le taux de mutation durant les générations pour maintenir la diversité de la population et éventuellement faire sortir l'algorithme d'un optimum local. L'application de la mutation dépend du codage utilisé dans la modélisation du problème, la mutation dans un codage binaire consiste à inverser tout simplement un bit du chromosome, dans un codage réel on peut générer un nombre aléatoire suivant une loi uniforme sur l'intervalle des valeurs possibles. On note que les conséquences de la mutation sont complètement aléatoires, et donc on ne peut garantir l'amélioration des solutions en utilisant cet opérateur.

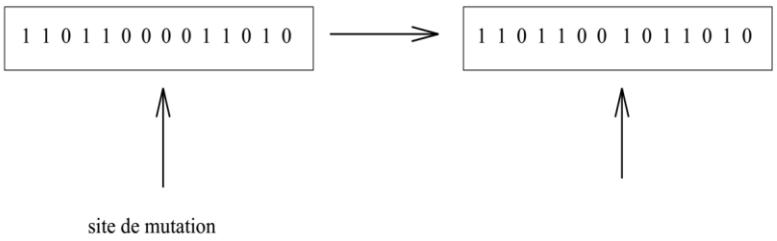


Figure 36 Exemple d'une mutation dans un codage binaire

III.6 Le paradigme mimétique

Les approches évolutionnaires dépendent fortement de l'aléatoire, optimiser les individus de la population implique une recherche aléatoire faite sur leur voisinage. Cependant cette recherche n'est guidée que par l'aléatoire et la chance, ce qui implique un temps de recherche plus important et plus de générations afin d'atteindre des solutions satisfaisantes. Dans les principes de l'évolution naturelle de Darwin et le paradigme mimétique, un individu est capable d'apprendre et de s'améliorer par lui-même en plus du patrimoine génétique qu'il a hérité de ses ancêtres. C'est dans ce cadre-là que plusieurs travaux de recherche tentent d'hybrider les algorithmes évolutionnaires classiques avec la recherche locale afin d'optimiser les individus avec une recherche guidée par des heuristique du domaine. Les individus sont optimisés à la fois par le mécanisme évolutionnaire et par l'approche locale, cette combinaison permet de réduire considérablement le temps et le nombre de générations nécessaires pour la convergence. Cette hybridation est connues sous les appellations : algorithmes évolutionnaires hybrides, algorithmes évolutionnaires locales et enfin algorithmes mimétiques (Neri & Cotta 2012).

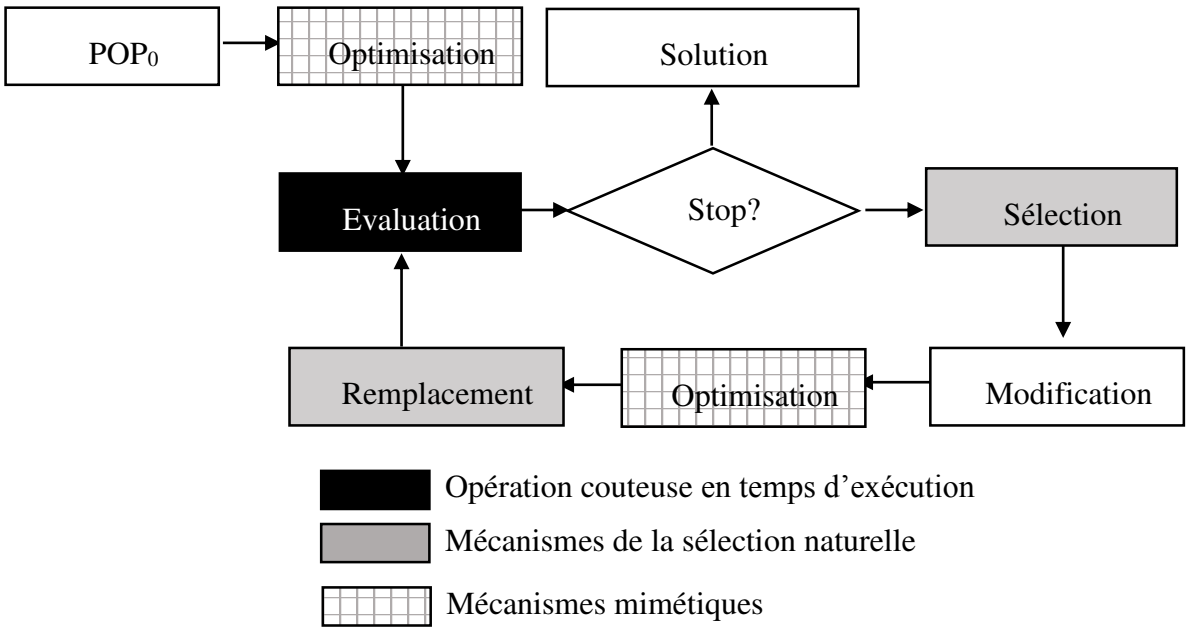


Figure 37 Les étapes d'une approche mimétique

Du point de vu général, les algorithmes mimétiques sont des métaheuristiques composés d'une technique évolutionnaire et d'un ensemble de techniques de recherche locales qui sont activées durant le processus évolutionnaire à travers les générations. Une approche mimétique peut se résumer dans les étapes suivantes (Figure 37), on indique en gras les étapes supplémentaires par rapport à une approche évolutionnaire classique :

1. la génération de la population POP_0 initiale composée d'un nombre fini d'individus avec une distribution uniforme.
2. **Optimiser chaque individu de POP_0 avec une approche de recherche locale.**
3. Evaluation de la population.
4. Si le critère d'arrêt est atteint alors s'arrêter.
5. Sélection d'un sous ensemble d'individus.
6. Application des opérateurs de modification pour créer de nouveaux individus.
7. **Optimiser les nouveaux individus par une approche de recherche locale.**
8. Remplacer certains individus de la population par ceux créés en 7.
9. Revenir à l'étape 2.

III.7 Les modèles évolutionnaires parallèles

L'application d'une approche évolutionnaire sur un problème d'optimisation est confrontée au temps d'exécution qui augmente considérablement avec l'augmentation de la taille de la population ou la complexité de la fonction de fitness utilisée. Actuellement, on a accès plus facilement à des machines de plus en plus puissantes, et de plus en plus parallélisées, la puissance de calcul de milliers de machines est à la portée de toutes les bourses via des grilles de calcul ou d'abonnements sur différentes plateformes du Cloud Computing.

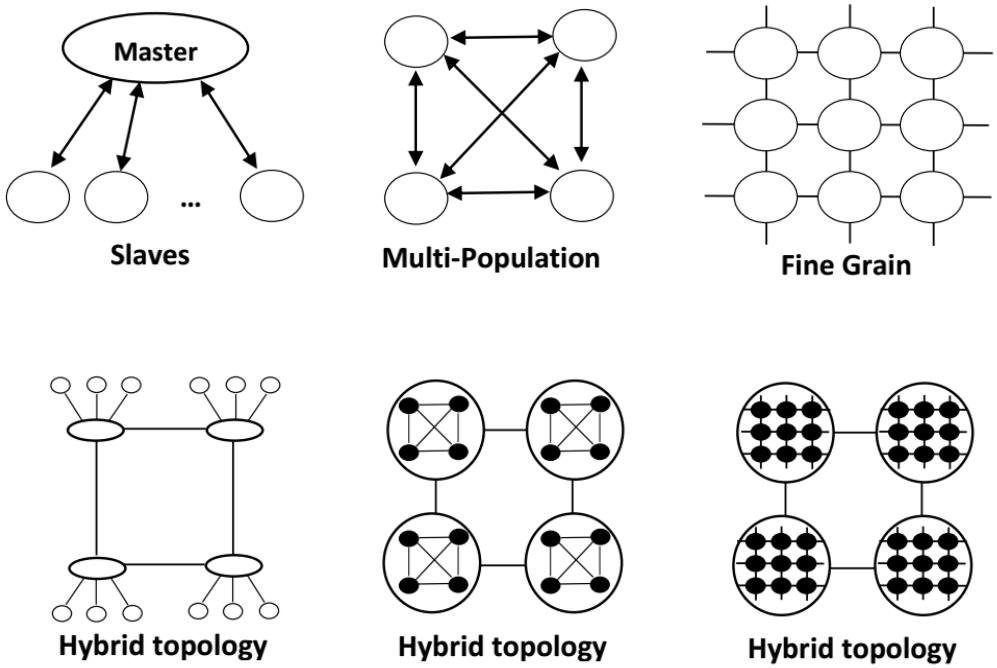


Figure 38 Exemples de topologies AE parallèles

Cette avancée technologique a suscité des recherches très actives sur la parallélisation d'algorithmes très lourds en temps de calcul, dans la famille des approches évolutionnaires on site par exemple: les algorithmes génétiques parallèles (Cantú-Paz 1998), les essaims particulaires parallèles (Vanneschi et al. 2011), les colonies artificielles d'abeilles parallèles (Narasimhan 2009), les colonies de fourmis parallèles (Pedemonte et al. 2011), les approches mimétiques parallèles (Tang et al. 2006) et les algorithmes à estimation de distribution parallèles (Lozano et al. 2002).

Plusieurs modèles ont été proposés dans la littérature afin de paralléliser les approches évolutionnaires et peuvent être classés selon plusieurs topologies dont les principales sont : les modèles Maître-esclave, les modèles d'îlots (ou à multiples populations), les modèles complètement distribués (Fine Grain) et les topologies hybrides (voir Figure 38).

III.7.1 Le modèle maître-esclave

Le plus facile à réaliser, le modèle maître-esclave vise tout simplement à distribuer l'étape de l'évaluation de la fonction de fitness (qui est l'étape la plus couteuse en temps de calcul) sur des processus esclaves, les autres étapes de l'approche évolutionnaires sont réalisées par le processus maître. D'un point de vu algorithme on peut dire que cette implémentation est identique à une implémentation séquentielle. L'un des inconvénients de cette architecture est que le processus maitre doit attendre le résultat de toutes les évaluations des processus esclaves afin de continuer vers la prochaine génération, cependant, dans une configuration matérielle hétérogène, certains processus peuvent être plus lents que d'autres et peuvent ralentir l'optimisation, ou pire tomber en panne et bloquer cette dernière. Certaines solutions peuvent être envisagées en utilisant un programme de scheduling qui balance la charge de travail sur les processus selon leurs performances et l'utilisation de processus de secours.

Le temps global de l'optimisation dépend du temps de l'évaluation à travers les processus esclaves ainsi que le temps des communications entre ces derniers et le processus maître. Un temps de communication élevé peut nuire aux performances de l'optimisation. L'accélération dans ce modèle est prouvée être linéaire au nombre de nœuds esclaves jusqu'à une limite maximale où le temps des communications devient plus important que le temps d'une évaluation de fitness, cependant, en pratique l'évaluation de la fitness est bien plus gourmande en temps de calcul que les communications.

III.7.2 Le modèle des îlots

Dans ce modèle, la population est divisée en plusieurs sous populations qui évoluent de façon indépendantes sur des îlots imaginaires (représentant des processus ou des unités de calcul). Cette configuration se rapproche beaucoup plus de l'inspiration biologique qu'on trouve au sein de groupes d'individus isolées géographiquement les uns des autres. L'un des points culminant de ce modèle est la possibilité de migration des individus d'une sous population (îlot) vers une autre, ce qui permet d'augmenter la diversité et de garantir une meilleure convergence (Madera et al. 2006), (Corcoran & Wainwright 1994).

En pratique, cela se résume à exécuter toutes les étapes d'une approche évolutionnaire sur une sous population de façon isolée par rapport aux autres sur un processus, un procédé de communication entre les processus est mis en œuvre afin de gérer la migration des individus d'une sous population à une autre. Ce modèle dépend de plusieurs paramètres :

- Le nombre et la taille des îlots.
- A quelle fréquence se fait la migration.
- Le nombre d'individus à migrer.
- La destination de chaque individu à migrer.
- La stratégie de sélection des individus à migrer.

Il est difficile de faire varier tous ces paramètres dans un processus d'optimisation, c'est pour cela que dans cette thèse on a fixé certains paramètres empiriquement, et on a choisi de ne faire varier que les stratégies de migration afin de voir leur impact sur l'optimisation, voir l'étude de Ruciński et ses collègues (Ruciński et al. 2010) pour une comparaison plus poussée sur l'impact de chaque paramètre sur les performances.

III.7.3 Le modèle complètement distribué

Dans ce modèle, la population est organisée en grille rectangulaire à 2 dimensions (voir Figure 38 topologie Fine Grain) où chaque nœud traite un sous ensemble de la population, l'évaluation de la fitness dans les nœuds est faite de façon parallèle. Les autres étapes du processus évolutionnaire sont faites par les nœuds voisins, dans la littérature on peut trouver ce modèle sous les noms : algorithmes évolutionnaires de diffusion, ou bien algorithmes évolutionnaires cellulaires ou algorithmes évolutionnaires massivement parallèles, cependant, il est très peu utilisé par rapport aux autres modèles.

III.8 L'approche mimétique proposée

Dans cette section, on présente une approche pour la génération de vecteurs de pondération optimisés via une méthode évolutionnaire (*CGEDA*) hybridée avec une méthode de sélection d'attributs (*SVM-RFE*) afin d'améliorer les performances en classification de données complexes avec les *SVM*. L'approche proposée est boostée en temps d'exécution via une implémentation parallèle basée sur les modèles d'îlots.

III.8.1 Motivations

- Les *SVM* réduisent le risque de surapprentissage (Vapnik 1995).
- Les *SVM* sont très performants sur des données complexes telles que les données textuelles (Joachims 1998).
- L'approche *SVM-RFE* est très efficace sur des données de grande dimension et elle prend en considération l'interdépendance entre les attributs (Guyon et al. 2002), (Zhang & Huang 2015).

- Les approches évolutionnaires permettent d'avoir de meilleurs minimas que les méthodes de recherche locales (Goldberg & Holland 1988).
- L'approche *CGEDA* comporte peut de paramètres et est donc facile à paramétrer (Shahraki & Tutunchy 2013).
- La combinaison entre une approche évolutionnaire et une approche locale (paradigme mimétique) permet une meilleure convergence en moins de générations (Neri & Cotta 2012).
- Les approches évolutionnaires sont facilement parallélisables (Sudholt 2015).
- Les modèles parallèles d'îlots permettent de booster la vitesse tout en gardant une bonne diversité dans la population ce qui induit une meilleure convergence (Ruciński et al. 2010).

III.8.2 Formulation évolutionnaire du problème

L'approche proposée repose sur un *SVM* linéaire dans l'étape d'évaluation de la qualité des individus de la population actuelle. Un *SVM* linéaire ne dépend que de très peu de paramètres et est donc facile à régler, il nous facilite par la même occasion la combinaison de l'approche évolutionnaire avec la sélection d'attributs basée sur l'approche *MSVM-RFE*.

Le but est de trouver un vecteur de pondération optimisé *w* qui maximise les performances en taux de classification d'un *SVM* linéaire tout en minimisant le nombre d'attributs utilisés (voir Figure 39). Dans notre approche, l'évaluation est réalisée sur chaque individu de la population (ou vecteur de pondération *w*) en utilisant *m SVMs* linéaires binaires en mode un contre tous, où *m* représente le nombre de catégories de l'ensemble d'apprentissage traité.

III.8.2.1 Représentation des individus

En classification, les attributs n'ont souvent pas la même expressivité, certains attributs ont plus de pertinence et d'autres sont plus discriminant vis-à-vis des catégories visées. La pondération des attributs permet d'équilibrer l'impact des attributs sur la classification selon leur importance sans pour autant les supprimer de façon brutale comme c'est le cas en sélection d'attributs, cependant les attributs les moins importants auront des poids qui tendent vers zéro.

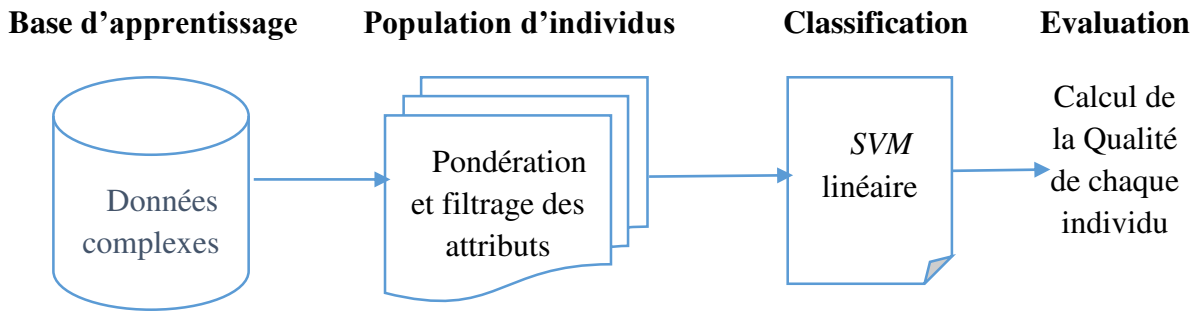


Figure 39 Schéma représentant la formulation évolutionnaire de la classification des données complexes par les SVM

Dans cette partie, il est question de trouver une formulation adéquate du chromosome/individu qui permet une bonne pondération des attributs, la pondération va se faire avec une simple multiplication entre le poids indiqué dans le chromosome et la valeur de l'attribut. Dans l'approche proposée, un individu est représenté par un vecteur de valeurs réelles dans l'intervalle [0, 1], cet individu est utilisé comme un masque pour la pondération et l'élimination des attributs. La valeur zéros est utilisée pour éliminer des attributs, la pondération maximale est à la valeur 1. La dimension d'un individu est égale au nombre d'attributs (voir équation22). X_w représente la matrice des données après pondération, X représente la matrice originale, w est le vecteur de pondération (un individu de notre population), i est l'indice des instances et j est l'indice des attributs.

$$X_w(i,j) = w(j) \times X(i,j) \qquad 22$$

III.8.2.2 Fonction de fitness

L'objectif principal de l'approche proposée est de trouver un vecteur de pondération optimisé w qui maximise les performances en classification d'un *SVM* linéaire avec un minimum d'attributs. Dans cette approche, pour chaque individu de la population (ou vecteur de pondération w) on entraine m *SVMs* linéaires binaires en mode un contre tous sur la base d'apprentissage pondérée avec w (voir équation 22), où m représente le nombre de catégories. Pour chaque *SVM* on évalue la précision (voir équation 23) et le rappel (voir équation 24) puis le *F-Score* moyen pour toutes les catégories (voir équation 25), où tp_i sont les vrai positifs (true positive) du *SVM* sur la catégorie i, fp_i et fn_i sont respectivement les faux positifs (false positive) et les faux négatifs (false négative).

$$\Pr ecision_{i=1..m}(w) = \frac{tp_i}{tp_i + fp_i} \qquad 23$$

$$\mathrm{Re}call_{i=1..m}(w) = \frac{tp_i}{tp_i + fn_i} \qquad 24$$

$$F.Score = macroF_1(w) = \frac{1}{m}\sum_{i=1}^{m} \frac{2 \times Recall_i(w) \times Precision_i(w)}{Recall_i(w) + Precision_i(w)} \qquad 25$$

Afin de réduire le risque de surapprentissage, l'évaluation a été faite en utilisant un test croisé à N parties (*N-fold* cross validation) sur la base d'apprentissage de telle façon où chaque partie est utilisée une fois pour le calcul de la $macroF_1^k$ pour $k=1..N$. La moyenne des N parties est utilisée afin de calculer le $SVMF_1$ (voir équation 26).

$$SVMF_1(w) = \frac{1}{N}\sum_{k=1}^{N} macroF_1^k(w) \qquad 26$$

On réfère à *NZ(w)* comme étant le nombre de zéros dans le vecteur de pondération *w* (voir équation 27) où *j* est l'indice sur les attributs variant de *1* jusqu'à *nf* (nombre d'attributs) et (*x==0*) est une fonction booléenne qui retourne un *1* si *x=0* sinon un *0*.

$$NZ(w) = \frac{1}{nf}\sum_{j=1}^{nf}(w(j) == 0) \qquad 27$$

Finalement, la fonction de fitness pour un vecteur de pondération w est calculée en utilisant l'équation 28 (à maximiser). Le second terme de l'équation permet de mesurer la sparsité induite par le vecteur de pondération *w* sur les données, ou pour simplifier, le nombre d'attributs éliminés. L'objectif de cette fonction est de trouver un vecteur de pondération *w* qui maximise les performances en classification avec un minimum d'attributs où α et β sont des constantes de réglage. Dans la partie expérimentale, on a utilisé les valeurs $\alpha=1$ and $\beta=1/nti$ où *nti* est le nombre d'instances dans la partie apprentissage de la base, ce qui nous permet de donner plus de priorité à l'optimisation des performances du classifieur par rapport à l'élimination des attributs.

$$fitness(w) = \alpha \times SVMF_1(w) + \beta \times NZ(w) \qquad 28$$

III.8.2.3 CGEDA : Algorithme à estimation gaussienne de distribution continue

Les algorithmes à estimation de distribution (*EDA* : *Estimation Distribution Algorithms*) ont été introduits comme des approches évolutionnaires par (Mühlenbein & Paass 1996) et sont largement discutés et comparés dans le livre de Larrañaga et Lozano (Larrañaga & Lozano 2002). Contrairement aux algorithmes génétiques, ces approches n'utilisent pas d'opérateurs de croisement ou de mutation, la population suivante est générée par l'échantillonnage d'une fonction de distribution dont les paramètres sont estimés à partir des individus sélectionnés de la génération actuelle. Tous les *EDA* sont basés sur les mêmes étapes :

1. Générer *M* individus (population initiale) à partir d'une distribution uniforme.
2. Evaluer chaque individu en utilisant la fonction de fitness.
3. Sélectionner un sous ensemble de *N* individus en utilisant une stratégie de sélection.
4. Estimer la fonction de distribution des individus sélectionnés.
5. Générer *M* individus (population suivante) par échantillonnage de la fonction de distribution estimée.
6. S'arrêter si le critère d'arrêt est atteint, sinon aller à 2.

La principale difficulté dans les *EDA* c'est l'estimation de la distribution des individus sélectionnés, plusieurs modèles d'approximation ont été proposés dans la littérature, ils sont généralement divisés en trois catégories en se basant sur le type de dépendance supposé entre les variables du modèle :

1. Les algorithmes à dépendance uni-variable, où aucune dépendance n'est supposée entre les variables. On trouve dans cette catégorie les algorithmes *UMDA : Univariate Marginal Distribution Algorithm* (Mühlenbein 1997) ou *PBIL : Population-Based Incremental Learning* (Baluja 1994).
2. Les algorithmes à dépendance bi-variables, où les variables sont supposées être dépendantes deux à deux. Comme exemples, on cite *MIMIC* : *Mutual Information Maximization for Input Clustering* (De Bonet et al. 1997).
3. Les algorithmes à dépendance multi-variables, les dépendances du modèle couvrent plusieurs variables à la fois, ces algorithmes sont souvent plus complexes et plus lents à l'estimation cependant ils se rapprochent le plus de la distribution réelle des données. Comme exemples, on cite *EBNA* : *Estimation of Bayesian Networks Algorithm* (Etxeberria & Larrañaga 1999) et *BOA* : *Bayesian Optimization Algorithm* (Pelikan et al. 1999).

Ces algorithmes sont généralement utilisés avec un codage discret, cependant plusieurs modèles utilisant un codage continu ont été proposés dans la littérature, voir l'ouvrage (Larrañaga & Lozano 2002) pour une comparaison détaillée de ces modèles.

L'algorithme à estimation gaussienne de distribution continue (*CGEDA* : *Continuous Gaussian Estimation of Distribution Algorithm*) (Shahraki & Tutunchy 2013) est un *EDA* continu à dépendance multi-variables où la distribution des meilleurs individus est estimée en utilisant la loi normale multi-variée (ou multidimensionnelle voir équation 29), où μ représente la moyenne, $\sum$ la matrice de covariance et k la dimension d'un individu. Les paramètres $\sum$ et μ sont estimés à partir des individus sélectionnés à chaque génération en utilisant l'estimateur du maximum de vraisemblance.

$$\Phi\left(x|\mu,\Sigma\right)=\frac{1}{2\pi^{k/2}\left|\Sigma\right|^{1/2}}e^{-\frac{1}{2}\left[(x-\mu)\Sigma^{-1}(x-\mu)\right]} \qquad 29$$

L'avantage principal de cet algorithme est que contrairement à d'autres approches évolutionnaires, *CGEDA* est très facile à paramétrer vu qu'il ne dépend pas de paramètres additionnels, il utilise seulement la distribution estimée afin de générer la population suivante (exemple : les algorithmes génétiques dépendent fortement de la probabilité de mutation et du taux de croisement et du type de ces opérateurs).

III.8.3 Hybridation de l'optimisation *CGEDA* avec la sélection par *MSVM-RFE*

Dans cette approche, on propose d'introduire la sélection d'attributs par *MSVM-RFE* à l'intérieur du mécanisme évolutionnaire du *CGEDA* afin d'améliorer ses performances (paradigme mimétique, voir algorithme 1) : l'avantage d'une telle combinaison est que l'apprentissage des *SVMs* linéaires qui est nécessaire pour la sélection *MSVM-RFE* a déjà été faite lors de la phase d'évaluation du *CGEDA*.

Algorithme 1 Combinaison de l'approche *CGEDA* avec la sélection *MSVM-RFE*

Définir M: la taille de la population, nf: Nombre d'attributs
Train: base d'apprentissage, *Test*: base de test
$Data_w$: une notation pour la matrice *Data* pondérée avec w *//Eq.*22

Générer M individus aléatoirement dans $\mathbb{R}^{nf}$ de valeurs dans [0,1] : $POP(1..M, 1..nf)$

Définir fit_b=0 et wb=[ ] *//Afin de stocker la meilleure fitness et le meilleur individu*
While true **do**
//étape du CGEDA

for $l \leftarrow 1, M$ **do**
$w=POP(l, :)$
Faire l'apprentissage de *N MSVM* sur $Train_w$ *//utilisant une division N-fold*
$Rank_l$: Moyenne des *MSVM-RFE Rank* des *N MSVM //Eq.*12
Calculer $fitness(w)$ *//Eq.*28
If $fitness(w) > fit_b$ **then** $fit_b = fitness(w)$; $wb=w$; **endif**
end for

If critère d'arrêt atteint **then** break; **endif**

SPOP: Sélection de *N* individus utilisant une stratégie de sélection
Estimer μ et $\sum$ à partir de *SPOP //Eq.*29
POP_{new}: générer la nouvelle population en utilisant $\Phi(x|\mu, \sum)$ *//Eq.*29
Délimiter la nouvelle population dans [0,1]

//Optimisation des individus par MSVM-RFE
for $t \leftarrow 1, nf$ **do**
//Estimer un masque d'élimination en utilisant la médiane
$\psi(t) = median(SPOP(:, t))$
$\psi(t) = 1$ if $\psi(t) \neq 0$
$POP(:, t)=POP_{new}(:, t)\times \psi(t)$
end for

$$Ranking(t) = \frac{1}{M}\sum_{l=1}^{M} Rank_l(t)$$

Rf: ensemble des attributs restants *//{t, ψ(t) ≠ 0}*
Lf: la moitié des attributs de *Rf* dont le *Ranking* est le plus faible
for $i \leftarrow 1, M$ **do**
Ef: ensemble de τ% attributs aléatoirement choisis de *Lf*
$POP(i, Ef) = 0$
end for

end while

Apprentissage d'un *MSVM* sur la base d'apprentissage $Train_{wb}$
Prédire les catégories de $Test_{wb}$ en utilisant le *MSVM*
Reporter la *macro-F1* en utilisant les catégories prédites et les vraies catégories

Dans cette approche, on propose d'optimiser la sparsité des individus de deux façons. D'abord, on estime un vecteur qu'on a nommé « masque d'élimination » en utilisant la fonction médiane sur chaque colonne d'attributs à partir de la matrice des individus sélectionnés : les attributs qui ont un poids égal à zéro dans la majorité des individus sélectionnés sont éliminés des générations futures. La deuxième méthode est de calculer un vecteur d'évaluation de pertinence moyen des attributs à travers tous les individus sélectionnés en utilisant la mesure *MSVM-RFE*, de trier les attributs des plus pertinents au moins pertinents, puis, d'éliminer un sous ensemble d'attributs sélectionné aléatoirement à partir des attributs les moins pertinents.

III.8.4 Implémentation parallèle sur un modèle d'îlots

Afin de booster la vitesse d'exécution de l'algorithme proposé, on propose aussi une parallélisation sur un cluster basée sur un modèle d'îlots. L'idée principale est de diviser la population en plusieurs sous populations réduites et de répartir le traitement sur plusieurs processus (workers) communicants entre eux (Figure 40 et Figure 41). Trois types de workers ont été implémentés :

1. *Un îlot* :
    - Exécute toutes les étapes de l'algorithme mimétique sur une sous population.
    - Peut éventuellement déléguer une partie du calcul de la fitness aux workers d'évaluation.
    - Après l'étape de sélection, les individus sélectionnés sont envoyés au « *scheduler* » pour migration, et réception d'un autre ensemble (de taille fixée par le taux de migration) d'individus migrants à partir des autres îlots choisis suivant une stratégie de migration, ce sous ensemble est ajouté aux individus sélectionnés pour l'estimation de la distribution et la génération d'une nouvelle population avec une meilleure diversité.
2. *Worker d'évaluation* : Exécute seulement l'étape d'évaluation sur une portion d'une sous population déléguée par un des îlots.
3. *Le scheduler*: Assignation des workers d'évaluation aux îlots, la gestion de la migration des individus (réception en envoi d'individus migrants entre îlots).

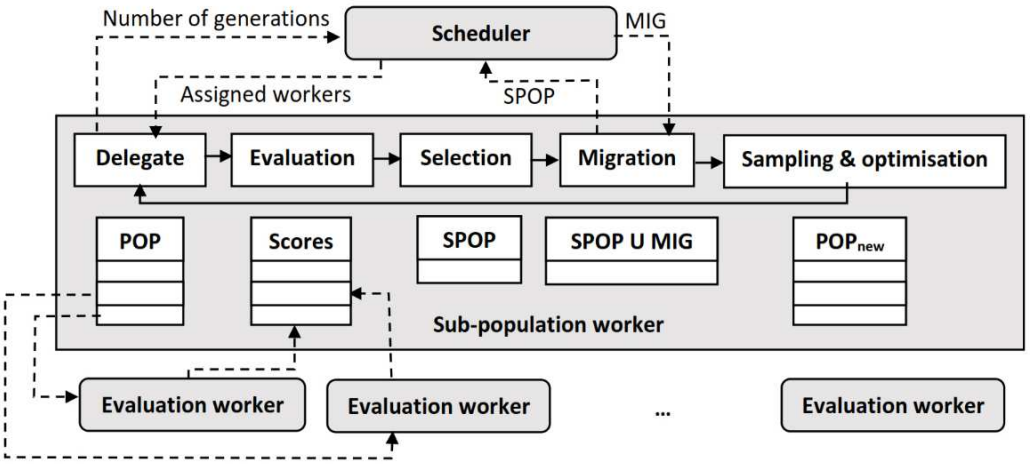


Figure 40 Les étapes de traitement d'un îlot parallèle

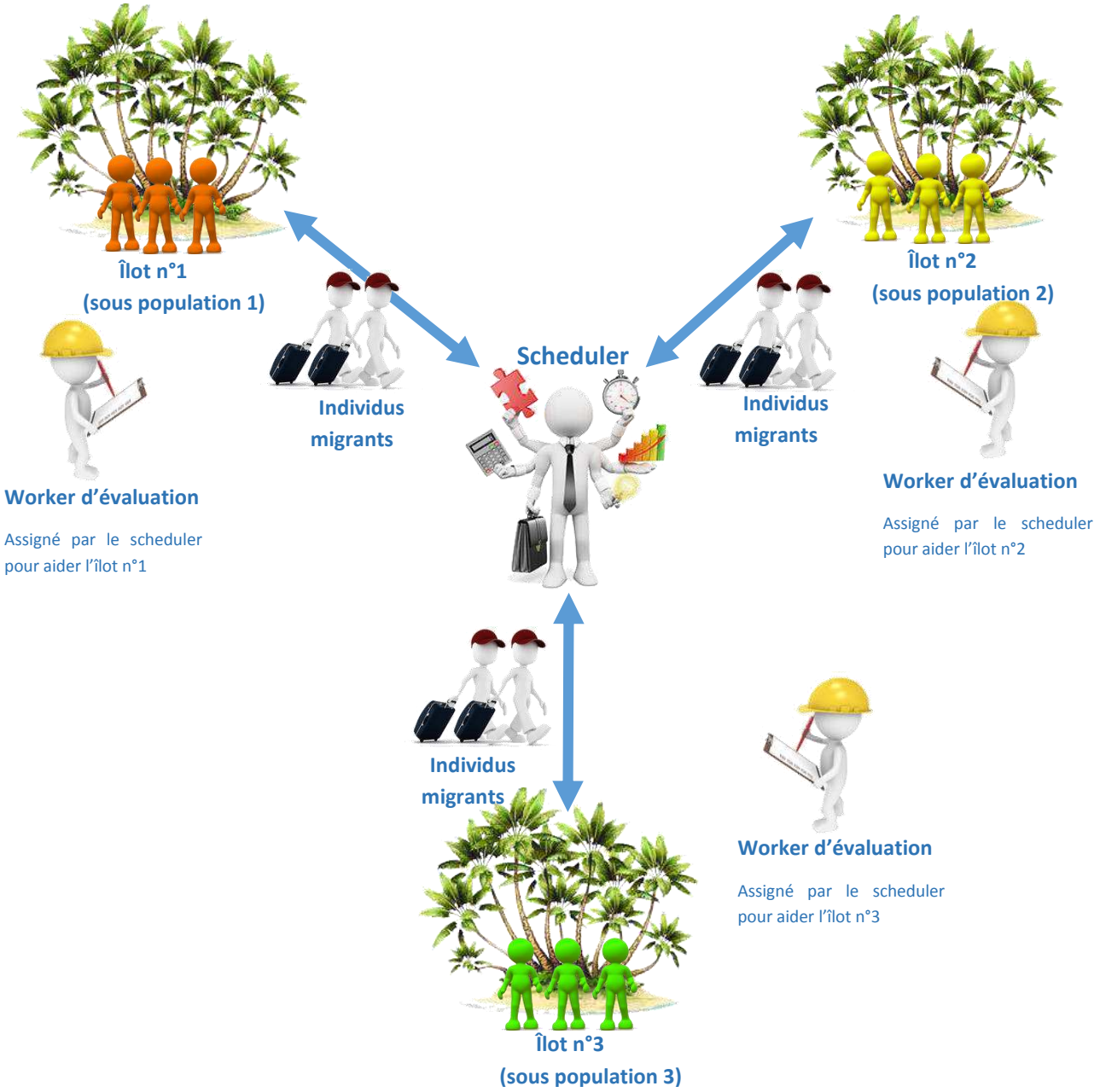


Figure 41 Représentation graphique du modèle d'îlot parallèle proposé

Dans l'approche proposée, *ns* et *ne* représentent respectivement le nombre d'îlots et le nombre de workers d'évaluation. La taille totale de la population est divisée sur *ns* donnant la taille d'une sous-population d'îlot. Chaque îlot peut déléguer l'évaluation de la fitness d'une partie de sa sous-population à un ou plusieurs workers d'évaluation qui sont assignés par le *scheduler*.

Le *scheduler* peut assigner plus de wokers d'évaluation aux îlots qui ont un retard par rapport aux autres en nombre de générations, souvent dû à l'hétérogénéité du matériel, certains îlots peuvent s'exécuter sur des processeurs moins puissants. Pour calculer le nombre de workers d'évaluation assignés au $k^{ième}$ îlot, on propose l'utilisation des équations 30 et 31, où gen_i est le nombre de générations effectuées par le $i^{ème}$ îlot. Les workers assignés sont ceux qui ont la file

d'attente la moins chargée, $assign_k$ représente le nombre de workers assignés au $k^{ième}$ îlot, *ceil* est une fonction d'arrondissement.

$$delay_k = \max_{i=1..ns}(gen_i) - gen_k + 1 \quad (30)$$

$$assign_k = ceil\left(\frac{ne \times delay_k}{\sum_{j=1}^{ns} delay_j}\right) \quad (31)$$

Afin d'introduire plus de diversité dans les îlots et ainsi améliorer la convergence, plusieurs stratégies de migration ont été adoptées et comparées dans la partie expérimentation :

1. La stratégie des meilleurs (*Best policy*) : le *scheduler* choisi toujours les meilleurs individus pour la migration vers d'autres îlots.
2. La stratégie de la diversité (*Different policy*) : le *scheduler* calcul le centroïde de chaque sous population d'îlot et choisi comme individus migrants vers cet îlot, les individus les plus différents en terme de distance par rapport à ce centroïde afin d'améliorer la diversité de cette sous-population.
3. La stratégie aléatoire (*Random policy*) : le *scheduler* choisi les individus migrants vers un îlot de façon aléatoire à partir des autres îlots.

III.9 Conclusion

On a présenté dans ce chapitre un état de l'art sur les approches évolutionnaire ainsi que le paradigme mimétique. Après avoir introduit des notions préliminaires sur les algorithmes évolutionnaires, on a passé en revue les principales étapes utilisée en optimisation évolutionnaire, ainsi que les limites de ces algorithmes en relation avec le temps de calcul nécessaire lors de la phase d'optimisation. Par la suite le paradigme mimétique a été introduit comme une proposition de solution afin de pallier aux limites des approches évolutionnaires, l'état de l'art indique qu'il est possible d'accélérer et d'améliorer la convergence via des hybridations avec des méthodes de recherche locale. Une deuxième proposition de solution a été abordée via la parallélisation du processus évolutionnaire. Motivés par la puissance du calcul parallèle qui est actuellement à portée de toutes les bourses, on a détaillé et comparé certains modèles parallèles proposés dans la littérature pour booster la vitesse de l'optimisation parallèle. D'après l'étude théorique, il est clair que le modèle d'îlots offre plus de diversité à la population et donc une meilleure convergence. Ce modèle offre aussi l'avantage d'être plus souple et plus tolérant aux pannes.

La suite du chapitre détaille l'approche mimétique proposée en optimisation de la représentation des données complexes. La représentation proposée repose sur un système de pondération et de sélection d'attributs optimisée à la fois par une approche évolutionnaire (*CGEDA*) et une approche de sélection d'attributs très performante (*MSVM-RFE*). Après avoir énoncé les motivations qui nous ont menés à ces choix, on a présenté la formulation évolutionnaire du problème ainsi que le calcul de la fonction de fitness.

L'introduction de la méthode *MSVM-RFE* dans le mécanisme évolutionnaire de l'approche *CGEDA* est détaillée algorithmiquement et permet à la fois d'optimiser le taux de la classification du *SVM* tout en réduisant la dimensionnalité des données suivant plusieurs mécanismes qu'on a intégrés. La complexité temporelle induite par les multiples apprentissages des *SVMs* nécessaires a été traitée via la parallélisation du mécanisme sur un modèle d'îlots prévu pour s'exécuter sur un cluster.

Le modèle parallèle proposé comprend plusieurs types de workers, cette division permet de répartir le traitement sur plusieurs unités de calcul, de gérer la charge de travail, ainsi que la gestion des communications et de la migration des individus d'un îlot vers un autre. Dans le chapitre suivant, cette approche sera expérimentée et comparée selon différents paramètres afin de montrer son efficacité du point de vue performances et temps d'exécution.

CHAPITRE 4

Expérimentations, résultats et discussion

IV.1 Introduction

Dans ce chapitre, on présente la partie expérimentale de cette thèse, les données choisies pour nos expérimentations sont des documents issus du web ainsi que des images aériennes.

Après une brève description des benchmarks utilisés et du prétraitement appliqués, les expérimentations seront divisées en plusieurs parties :

- La comparaison des approches usuelles de représentation, de pondération et de sélection d'attributs pour la classification.
- L'intégration de l'approche proposée avec et sans sélection *MSVM-RFE* dans un contexte non parallélisé afin de valider son efficacité par rapport aux approches usuelles.
- Une expérience finale montrant le gain en temps de calcul pour l'apprentissage du classifieur après la parallélisation du processus ainsi que l'influence des différents paramètres sur les performances.

L'évaluation avec les *SVM* est faite en utilisant la librairie LIBLINEAR[2] en mode un contre tous pour la classification multi-classes avec le paramètre de régularisation fixé empiriquement à *C=0.5* pour la classification des documents et à *100* pour la classification des images aériennes. La régularisation *L2* avec fonction de coût *L2* pour la classification (voir II.5.1) est utilisée pour l'apprentissage du *SVM* linéaire vu sa rapidité même sur de grandes bases d'apprentissage. Les vecteurs d'apprentissage et de test sont normalisés avant la phase d'apprentissage du classifieur.

Les expériences sont réalisées sur un cluster fait maison composé de 9 unités personnelles équipées d'un processeur INTEL core i5 à 3.20 GHz et 4Go de RAM connectées via un réseau LAN câblé et gérées par le *Toolbox Parrallel Computing* sous *Matlab*. Une unité va servir de serveur afin d'héberger le « *scheduler* » et le reste des unités seront des machines clientes qui hébergeront les îlots et les workers d'évaluation.

IV.2 Expérimentations sur la classification automatique des documents

Dans cette partie on expérimente l'approche proposée sur la classification du contenu textuel de plusieurs benchmarks dédiés.

IV.2.1 Description des bases de données et du prétraitement

Pour la partie expérimentation de cette thèse, on a choisi trois bases très populaires dans le domaine de la classification des documents web :

[2] https://www.csie.ntu.edu.tw/~cjlin/liblinear/

1. *Reuters-21578* (ou *R8*) : récupérée à partir de la page[3] de David Lewis et en utilisant la division standard en *modApté*. A partir de cette base on a utilisé un ensemble de 8 catégories ayant le plus d'exemples d'apprentissage positif : « acq », « crude », « earn », « grain », « interest », « money-fx », « ship » et « trade ». Les documents appartenant à aucune classe ou à plusieurs classes sont éliminés ce qui laisse un total de 7674 documents.
2. *7Sectors* : issue du projet CMU WebKB[4], elle contient 4581 pages html, chaque page est étiquetée avec sa catégorie principale « basic», « energy », « financial », « health », « transportation », « technology » et « utilities ».
3. *WebKB* : aussi issue du projet CMU WebKB, elle contient des pages html téléchargées à partir de sites web de plusieurs universités. Tel que suggéré par les nombreux travaux sur cette base, on a utilisé les catégories « project », « faculty », « student » et « course » ce qui fait un total de 4199 documents.

Dans nos expérimentations on a utilisé 70% de chaque base afin de réaliser l'apprentissage du classifieur avec un test croisé sur cette partie (*4 fold cross validation*) afin d'optimiser les paramètres du l'approche proposée. Les 30% restantes sont utilisées afin d'évaluer les performances du classifieur obtenu via le calcul du *F-Score* (*MacroF1*).

La partie prétraitement inclue :

- L'extraction des termes à partir du texte brute avec l'expression régulière :
$$[a\text{-}z]([a\text{-}z], [0\text{-}9], \text{-})^{+}$$
- Elimination des mots vides.
- La racinisation des termes (*stemming*) en utilisant l'algorithme Porter II.
- Elimination des termes rares (qu'on trouve dans moins de 10 documents).

Après prétraitement, on obtient une liste de de 5763 termes de la base *R8*, 4369 termes de la base *7Sectors* et 4315 termes de la base *WebKB*. Ce traitement nous permet d'obtenir des matrices avec les dimensions suivantes : *R8*(7674 × 5763), *7Sectors*(4581 × 4369) et *WebKb*(4199 × 4315).

IV.2.2 Performances des Schémas de pondération usuels

On commence nos expérimentations par une comparaison pratique des schémas de pondération standards (vus en chapitre I) sur les corpus choisis en utilisant un *SVM* linéaire sur l'ensemble complet des attributs pour chaque corpus.

[3] http://www.daviddlewis.com/resources/testcollections

[4] https://www.cs.cmu.edu/ webkb/

Le Tableau 2 montre les résultats de cette comparaison, il est clair que le choix du bon schéma est crucial sur les performances de la classification. Le *F-Score* varie de 85.7% à 91.8% pour la base *R8*, de 84.8% à 92.6% pour la base *7Sectors* et de 85.5% à 92.8% pour la base *Webkb*. Alors que le schéma $logTF^{IDF}$ semble plus performant sur les bases *R8* et *7Sectors*, le codage binaire donne de meilleurs résultats sur la base *Webkb*. Le meilleur schéma de pondération dépend des données : dans nos expérimentations le codage $logTF^{IDF}$ a la meilleure performance moyenne sur les trois corpus.

On a constaté que la fonction *log* est bénéfique sur les performances de la classification, et on a retenu le $logTF^{IDF}$ comme schéma d'initialisation dans nos expériences avec l'approche mimétique, de telle façon à pouvoir optimiser des résultats déjà très performants.

Tableau 2 F-Score sur les trois corpus en utilisant les schémas de pondération standards

Schémas de pondération	*TF*	*Binaire*	TF^{IDF}	$logTF^{IDF}$	*ITF*
R8	*0.857*	***0.918***	*0.883*	***0.918***	*0.909*
7Sectors	*0.875*	*0.848*	*0.915*	***0.926***	*0.873*
Webkb	*0.855*	***0.928***	*0.880*	*0.904*	*0.919*
Moyenne	*0.862*	*0.898*	*0.893*	***0.915***	*0.900*

IV.2.3 Performances des méthodes de sélection d'attributs usuelles

La deuxième expérience consiste à comparer la stabilité et les performances des approches de sélection d'attributs standards (*MI*, *CHI*, *IG*) par rapport à l'approche *MSVM-RFE* après les avoir combinées avec différents schémas de pondération (*BIN*, *TF*, TF^{IDF}, $logTF^{IDF}$, *ITF*).

Le début de chaque expérience commence par la mesure du *F-Score* obtenu avec une classification par un *SVM* linéaire multi-classes sur l'ensemble complet des attributs de chaque corpus, puis itérer :

1. Mesurer la pertinence de chaque attribut restant du corpus.
2. Trier les attributs des plus pertinents aux moins pertinents.
3. Eliminer un sous ensemble des attributs les plus faibles en pertinence parmi les attributs restants.
4. Refaire l'apprentissage du *SVM* multi-classes avec l'ensemble réduit des attributs.
5. Noter le *F-Score* obtenu.
6. revenir à 1.

L'estimation de la pertinence des attributs pour les mesures *MI*, *CHI* et *IG* est faite une seule fois au début de chaque expérience, vu qu'il n y a pas de supposition de dépendance entre les

attributs, la réduction de l'ensemble des caractéristiques n'affecte pas la mesure de pertinence des attributs restants. Cependant, pour l'approche *MSVM-RFE*, l'interdépendance entre les attributs est prise en considération vu que la réduction de l'ensemble de ces derniers affecte les poids du *SVM* entrainé et donc l'évaluation des attributs restants. Il est donc nécessaire de ré-estimer l'évaluation des attributs à chaque itération à partir des poids du *SVM* obtenu.

Dans cette expérience, on a essayé deux variantes de l'approche *MSVM-RFE* :

- *MSVM-RFE1*, on élimine à chaque itération 5% du total du nombre attributs
- *MSVM-RFE2*, on élimine à chaque itération 1% du total du nombre attributs

Figure 42, Figure 43 et Figure 44 montrent les courbes *F-Score* des meilleures combinaisons pondération - sélection d'attributs pour chaque corpus en fonction du nombre d'attributs éliminés. Cette expérience montre des courbes très rapprochées pour un sous ensemble d'attributs réduit à moins de 50%, on note une légère supériorité et plus de stabilité de la variante *MSVM-RFE2* par rapport aux autres courbes où les performances restent bonnes même après avoir éliminé 70% des attributs.

Après avoir éliminé plus de 50% des attributs, les performances chutent pour les approches *MI*, *CHI* et *IG* spécialement sur les corpus *7Sectors* et *Webkb*, montrant l'importance de l'interdépendance entre les variables dans la sélection d'attributs.

Cette expérience montre aussi que le nombre d'attributs éliminés à chaque itération peut affecter les performances et la stabilité de l'approche *MSVM-RFE* de manière significative, les courbes montrent que la variante *MSVM-RFE2* est meilleure que la variante *MSVM-RFE1* sur tous les corpus. Cependant *MSVM-RFE2* requière plus de temps de calcule vu qu'elle nécessite plus d'itérations.

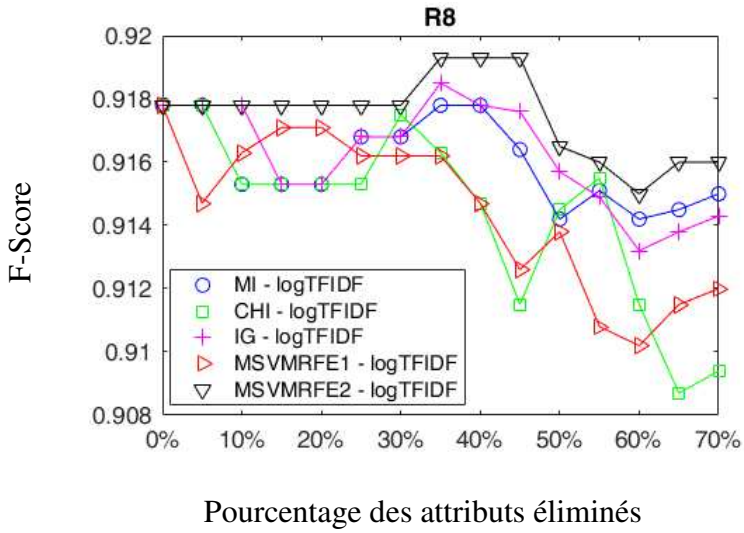


Figure 42 les courbes F-Score des meilleures combinaisons (pondération – sélection) d'attributs obtenues sur la base R8 en fonction du nombre d'attributs éliminés

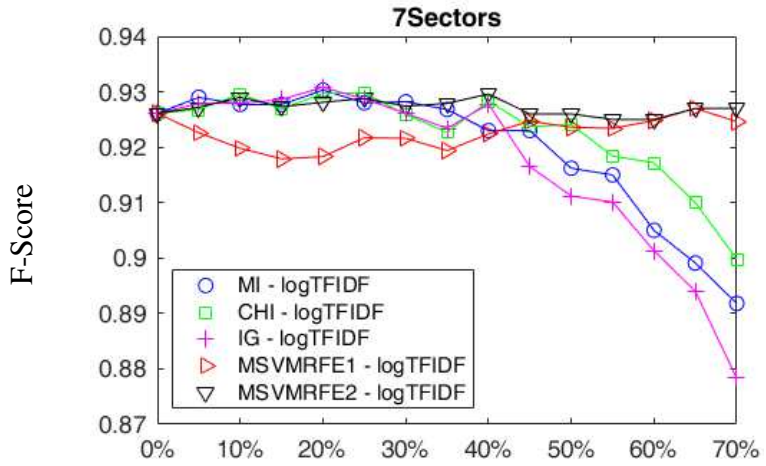


Figure 43 Les courbes F-Score des meilleures combinaisons (pondération – sélection) d'attributs obtenues sur la base 7Sectors en fonction du nombre d'attributs éliminés

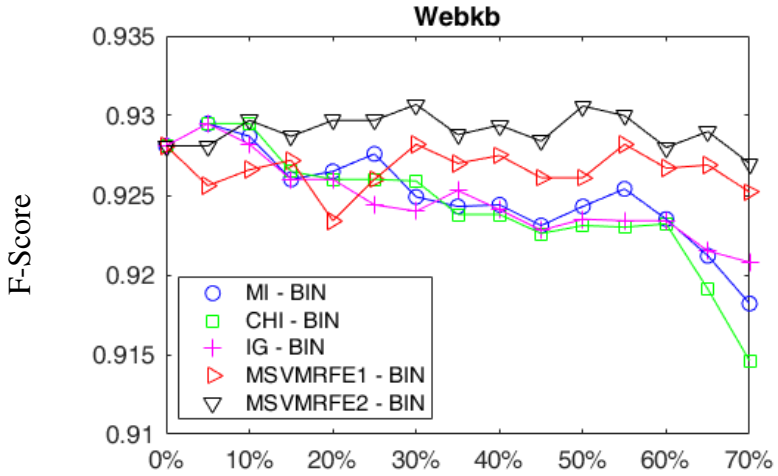


Figure 44 Les courbes F-Score des meilleures combinaisons (pondération – sélection) d'attributs obtenues sur la base Webkb en fonction du nombre d'attributs éliminés

IV.2.4 Intégration de l'approche mimétique proposée

L'expérience suivante consiste à montrer l'apport de l'intégration de l'approche *MSVM-RFE* dans le mécanisme évolutionnaire des *CGEDA* pour la pondération et la sélection d'attributs en termes de performances en classification ainsi qu'en taux de réduction des attributs.

Afin de maintenir un temps d'exécution raisonnable, on a fixé certains paramètres :

- La taille de la population a été fixée à 100 individus.

- Le taux de sélection de l'approche évolutionnaire est fixé à 25%.
- Après chaque génération, 25% des individus sélectionnés peuvent migrer vers d'autres îlots.
- Le nombre d'îlots est fixé à 4 (25 individus par îlot).
- Le nombre maximum de générations est fixé à 100.
- Le taux d'élimination des attributs τ=1% (voir algorithme 1).

Dans cette expérience, les performances sont évaluées en termes de *F-Score* ainsi qu'en taux de réduction (nombre d'attributs éliminés). Quatre lancements ont été effectués sur chaque corpus et pour chaque expérience et les moyennes des résultats des lancements sont reportées afin de réduire l'influence de l'aléatoire sur la présentation de ces derniers.

Le Tableau 3 montre la moyenne des performances du *CGEDA* avec et sans combinaison avec la sélection *MSVM-RFE* :

- Une première remarque qu'on constate est que les performances en termes de *F-Score* de l'approche *CGEDA* toute seule sont nettement meilleures que les approches de pondération et de sélection d'attributs standards (voir Tableau 2, Figure 42, Figure 43 et Figure 44 pour comparer).
- On constate aussi une légère amélioration du *F-Score* avec la combinaison *CGEDA+MSVM-RFE* par rapport à l'approche *CGEDA* toute seule.
- Le gain principal de cette combinaison est très prononcé en termes de taux de réduction de l'ensemble des attributs, alors que l'approche *CGEDA* n'arrive pas à réduire plus de 10% de l'ensemble des attributs, la combinaison *CGEDA* et *MSVM-RFE* arrive à réduire l'ensemble des attributs de plus de 70% sur *R8* et *Webkb* et de 67% sur *7Sectors* sans faire chuter les performances en classification.

Tableau 3 Performances en F-Score et en taux de réduction de la combinaison CGEDA+MSVM-RFE par rapport à CGEDA

Expérience	*CGEDA*		*CGEDA+MSVM-RFE*	
	F-Score	*Attributs éliminés*	*F-Score*	*Attributs éliminés*
R8	*0.970*	*7%*	***0.976***	*74%*
7Sectors	*0.972*	*6%*	***0.975***	*67%*
Webkb	0.963	*7%*	**0.966**	*72%*

IV.2.5 Influence des stratégies de migration sur les performances

Dans cette expérience, on compare les performances de l'implémentation parallèle proposée de la combinaison *CGEDA+MSVM-RFE* en variant les stratégies de migration (*Best* : la stratégie des meilleurs, *Different* : la stratégie de la diversité, *Random* : la stratégie aléatoire, *None* : sans migration). Les performances sont évaluées en termes de *F-Score* et en termes de diversité de la population.

Les îlots peuvent partager leurs solutions selon plusieurs schémas de coopération, l'expérience réalisée consiste à mesurer le *F-Score* et la diversité de la population pour chaque génération en faisant varier les stratégies migration pour chaque corpus. Chaque expérience est répétée 4 fois et la moyenne du *F-Score* et de la diversité est calculée pour chaque génération afin de réduire l'influence de l'aléatoire de l'approche évolutionnaire sur les résultats présentés. Pour une meilleure clarté des résultats, on a aussi calculé la moyenne des courbes à travers les corpus afin de ne garder qu'une seule courbe par stratégie de migration et avoir une vision globale des performances moyennes de chaque stratégie, ces courbes moyennes sont présentées en Figure 45 et Figure 46.

Une première constatation à partir des courbes est l'importance de la migration dans les modèles évolutionnaires parallèles, on note que les courbes de la stratégie « *None* » (ou sans migration) ont les performances les plus faibles en termes de *F-Score* et de diversité.

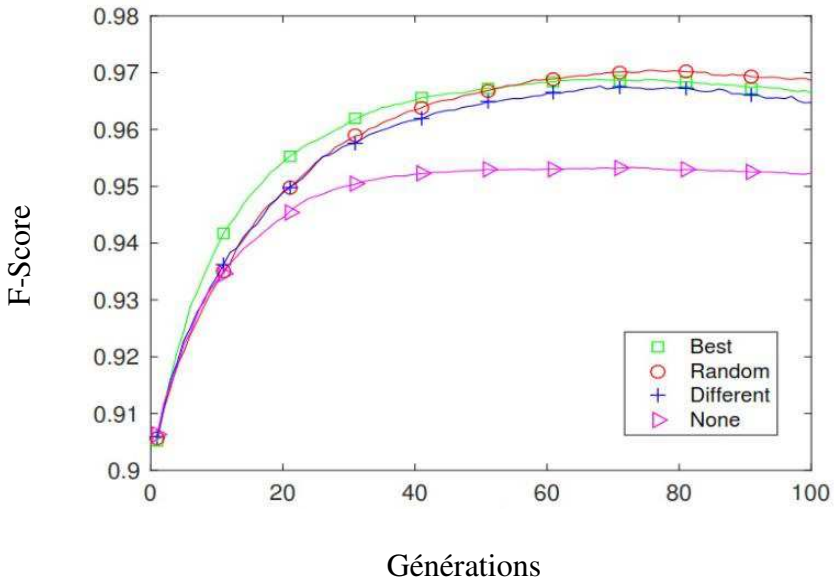


Figure 45 Moyennes des courbes du F-Score (sur tous les corpus) par génération pour chaque stratégie de migration

Les courbes en Figure 45 indiquent des résultats très rapprochés entre les trois stratégies de migration, on remarque que la courbe de la stratégie « *Best* » converge plus rapidement au début des générations mais stagne par la suite à cause d'une baisse en diversité de la population (Figure 46). Les stratégies « *Random* » et « *Different* » maintienne une meilleure diversité tout au long des générations, cependant, nos expériences ont montré une légère supériorité de la

stratégie « *Random* » par rapport à toutes les autres stratégies en terme de *F-Score* et de diversité.

Enfin, il serait intéressant d'étudier la combinaison de plusieurs stratégies de migration afin de booster la vitesse de convergence et améliorer la diversité de la population en même temps.

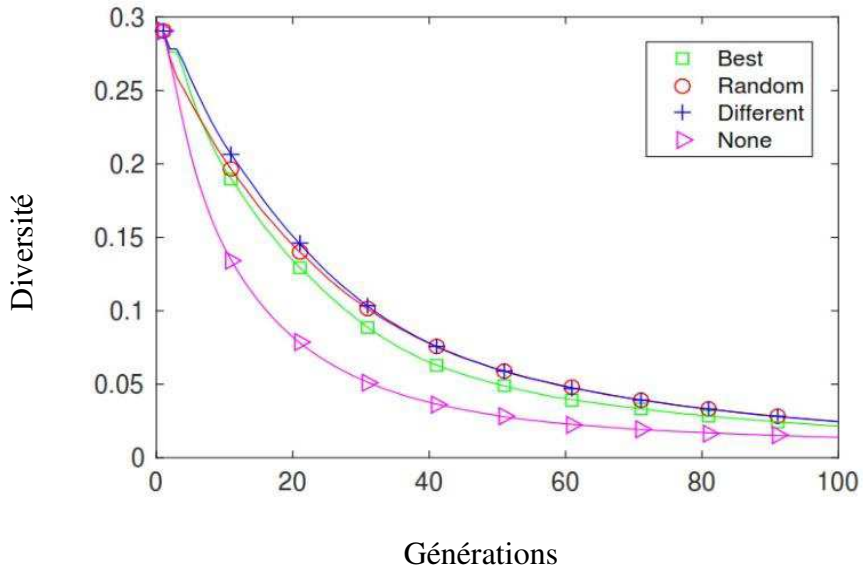


Figure 46 Moyennes des courbes de la diversité (sur tous les corpus) par génération et pour chaque stratégie de migration

IV.2.6 Etude de l'accélération de l'implémentation parallèle proposée

L'objectif de cette expérience est de montrer le gain en temps d'exécution obtenu grâce au modèle parallèle proposé ainsi que l'accélération obtenue en ajoutant plus de ressources matériels (workers).

Le Tableau 4 montre les résultats obtenus en temps d'exécution sur l'implémentation parallèle proposée en variant le nombre de workers sur les 3 corpus sans compter le *scheduler*. Ici le nombre de workers comprend le nombre d'îlots plus le nombre de workers d'évaluation. Pour les expériences avec 1, 2 et 4 workers on a utilisé que des îlots, pour les expériences avec 6 et 8 workers on a utilisé 4 îlots et respectivement 2 et 4 workers d'évaluation.

D'après nos expérimentations, avec une population de 100 individus, utiliser plus de 4 îlots peut dégrader les résultats obtenus. En effet, le nombre d'individus par sous population est obtenu par une division de la taille totale de la population sur le nombre d'îlots, augmenter le nombre d'îlots risque de réduire fortement la densité des sous populations et ainsi dégrader leurs capacités de recherche et d'optimisation.

Les résultats présentés en Tableau 4 (en secondes par génération) montrent que le temps d'exécution est visiblement réduit avec l'augmentation du nombre de workers. Le temps d'exécution est réduit de 1233 à 264 secondes par génération en passant de 1 worker à 8 workers

pour le corpus *R8*. Cette réduction nous a permis de passer de 35h en temps d'exécution pour 100 générations à seulement 7h et 30min pour le corpus *R8*, ce qui est très convenant pour un corpus aussi grand. On note des réductions similaires pour les autres corpus : de 206 à 42 secondes par génération pour la base *7Sectors* et de 120 à 28 secondes par génération pour la base *Webkb*. Cela réduit le temps d'exécution total sur 100 générations de 6h à 1h pour la base *7Sectors* et de 3h30min à 45min pour la base *Webkb*.

Tableau 4 Temps d'éxecution moyen (en secondes) par génération pour chaque corpus en variant le nombre de workers

	Nombre de workers				
Corpus	*1*	*2*	*4*	*6*	*8*
R8	*1233 sec*	*725 sec*	*535 sec*	*380 sec*	*264 sec*
7Sectors	*206 sec*	*129 sec*	*84 sec*	*57 sec*	*42 sec*
Webkb	120 sec	*84 sec*	52 sec	*35 sec*	*28 sec*

L'accélération (speedup) est un facteur qui mesure les performances en temps d'exécution relatives (par division, équation 32) entre une exécution parallèle sur un seul worker (ou exécution non parallèle) et une exécution sur plusieurs workers.

$$Speedup(k) = \frac{temps\ d'exécution\ sur\ 1\ worker}{temps\ d'exécution\ sur\ k\ workers} \qquad 32$$

La Figure 47 montre les courbes d'accélération de l'approche proposée sur chaque corpus en variant le nombre de workers. Les courbes montrent une accélération pseudo linéaire avec l'augmentation du nombre de workers qu'on a estimé avec l'équation 33.

Théoriquement, on peut assumer qu'on peut réduire encore plus le temps d'exécution avec l'ajout de plus de workers (ou unités de calcul), cependant, le temps des communications entre les workers augmente avec l'augmentation du nombre de ces derniers, arrivé à un certain point critique on arrive à une accélération maximale qui n'augmente plus avec l'ajout de plus de workers. On rappelle que dans nos expériences, chaque worker s'exécute sur une machine individuelle et que les communications passent à travers un réseau LAN câblé, limités par le matériel disponible on n'a pas pu réaliser des expérimentations avec un nombre plus important de workers afin d'estimer l'accélération maximale.

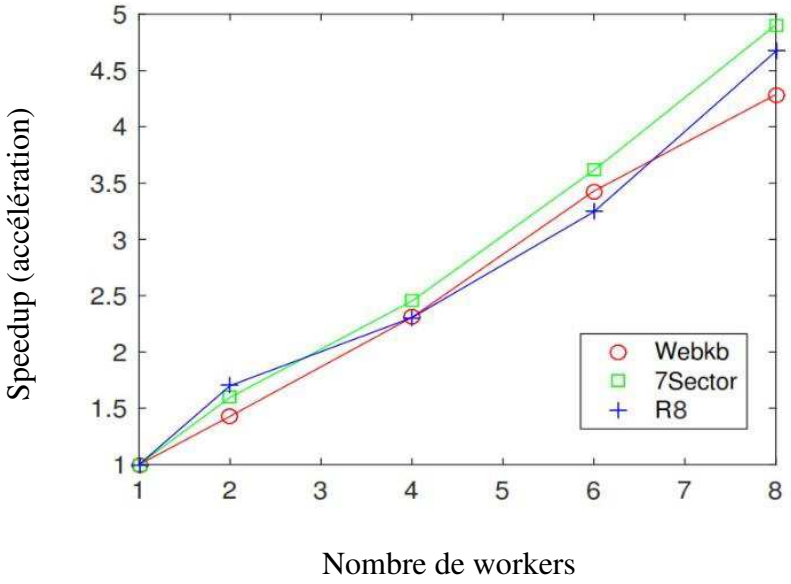


Figure 47 Accélération (Speedup) de l'implémentation parallèle proposée pour chaque corpus en variant le nombre de workers

$$Speedup(k) \approx 0.5 \times (k-1) + 1 \qquad 33$$

On note qu'on a pu réduire avec succès le temps d'exécution de l'approche proposée avec une implémentation parallèle basé sur le modèle d'îlots, ainsi qu'on a pu générer avec succès une représentation réduite et optimisée des données qui améliore les performances en classification sur les 3 corpus.

IV.3 Expérimentations sur la classification des images aériennes

Dans cette partie on expérimente l'approche proposée sur la classification des images aériennes issues du benchmark *UCMerced LandUse*.

IV.3.1 Description de la base de données et du prétraitement

UCMerced LandUse[5] est un benchmark composé de 21 classes d'images aériennes comportant 100 images par classe d'une dimension de 256x256 pixels en *RVB* avec une résolution de un pied issues du centre géologique des Etats Unis d'Amérique USGS (Yang & Newsam 2010). Les 21 classes représentées sont : *agricultural*, *airplane*, *baseball diamond*, *beach*, *buildings*, *chaparral*, *dense residential*, *forest*, *freeway*, *golf course*, *harbor*, *intersection*, *medium density residential*, *mobile home park*, *overpass*, *parking lot*, *river*, *runway*, *sparse residential*, *storage tanks*, *tennis courts* (voir Figure 48).

Dans nos expérimentations on a utilisé 70% des images de la base (distribuées uniformément sur les catégories) afin de réaliser l'apprentissage du classifieur, un test croisé (*4 fold cross*

[5] *http://weegee.vision.ucmerced.edu/datasets/landuse.html*

validation) est réalisé sur cette partie pour optimiser les paramètres du classifieur. Les 30% restantes sont utilisées afin d'évaluer les performances du classifieur obtenu via le calcul du *F-Score* (MacroF1).

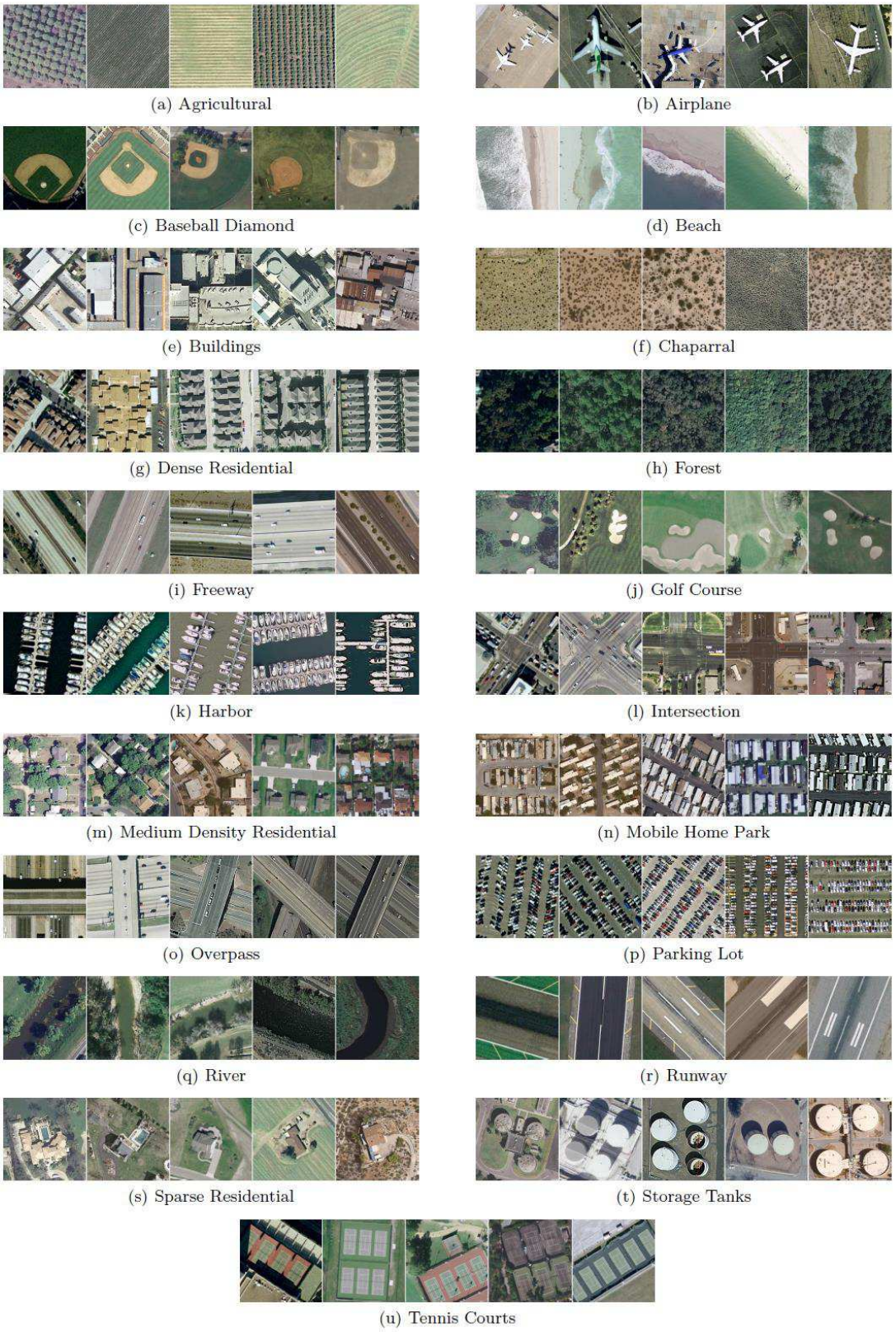


Figure 48 Exemples d'images du benchmark UCMerced LandUse (Yang & Newsam 2010)

Les codeurs utilisés pour l'extraction et la représentation des caractéristiques des images du benchmark sont :

- L'histogramme en *YCbCr* : 256 valeurs de l'histogramme de l'image représentée dans l'espace des couleurs : luminance chrominance.
- L'histogramme en *HSL* : 256 valeurs de l'histogramme de l'image représentée dans l'espace des couleurs : teinte saturation luminosité.
- Le codeur *LBP* avec un voisinage de 32x32: 102 attributs, 34 pour chaque couleur (rouge, vert, bleu).
- La matrice de cooccurrence : 8x8 attributs plus 4 attributs additionnels (contraste, corrélation, énergie, homogénéité) pour chaque couleur ce qui fait un total de 204 attributs.
- Le codeur à convolution pré-entrainé *GoogLeNet* (Szegedy et al. 2015): permet d'extraire 1024 attributs.
- Le codeur à convolution pré-entrainé *Caffe* (Jia et al. 2014): permet d'extraire 4096 attributs.
- Le codeur à convolution pré-entrainé *AlexNet* (Krizhevsky et al. 2012): permet d'extraire 4096 attributs.
- Le codeur à convolution pré-entrainé *ResNet* (He et al. 2016) : permet d'extraire 2048 attributs dans sa version *ResNet50*.
- *AllNet* : c'est une proposition de concaténation des attributs obtenus à partir des codeurs *GoogLeNet, Caffe, AlexNet et ResNet*, ce qui fait 11264 attributs (voir II.4.4).

L'extraction des caractéristiques avec les codeurs à convolution pré-entrainés a été faite avec le toolbox *MatConvNet*[6] sous *Matlab*.

IV.3.2 Performances des codeurs sans sélection d'attributs

Cette expérience consiste à comparer la qualité des attributs extraits des différents codeurs utilisés sans sélection, la qualité est mesurée en termes de performances en classification (*F-Score*) sur 4 classifieurs :

- Un *K-NN* en variant le k de 1 à10, en pratique, les meilleurs résultats ont été obtenus pour k=1.
- Un *MLP* avec 21 neurones en couche cachée utilisant la règle d'apprentissage de *Levenberg Marquardt*.
- Un *SVM* linéaire utilisant la régularisation *L2* avec fonction de coût *L2* (voir II.5.1).
- Un *SVM* avec un noyau gaussien dont l'écart type varie de 0.01 jusqu'à 1 avec un pas de 0.05, les meilleurs résultats sont reportés.

[6] http://www.vlfeat.org/matconvnet/

Les résultats obtenus montrent clairement (voir Figure 49) que les codeurs à convolution pré-entrainés surpassent de loin les codeurs d'images classiques (Histogramme en *YCbCr*, Histogramme en *HSL*, *LBP* et matrice de cooccurrence) avec un *F-Score* qui dépasse les 90% sur tous les classifieurs. On note aussi que les performances des *SVM* sont meilleures par rapport au *K-NN* et au *MLP*, avec une légère supériorité du *SVM* linéaire par rapport au *SVM* avec un noyau gaussien. Le meilleur résultat a été obtenu avec un *SVM* linéaire en utilisant les attributs du codeur *AllNet* (*F-Score* = 97.46%). Ceci montre l'apport positif de la concaténation de plusieurs codeurs, cependant, le gain est de moins de 1% pour un nombre d'attributs beaucoup plus élevé (11264) par rapport au codeur *ResNet* seul (2048 attributs avec un *F-Score* = 96.66%).

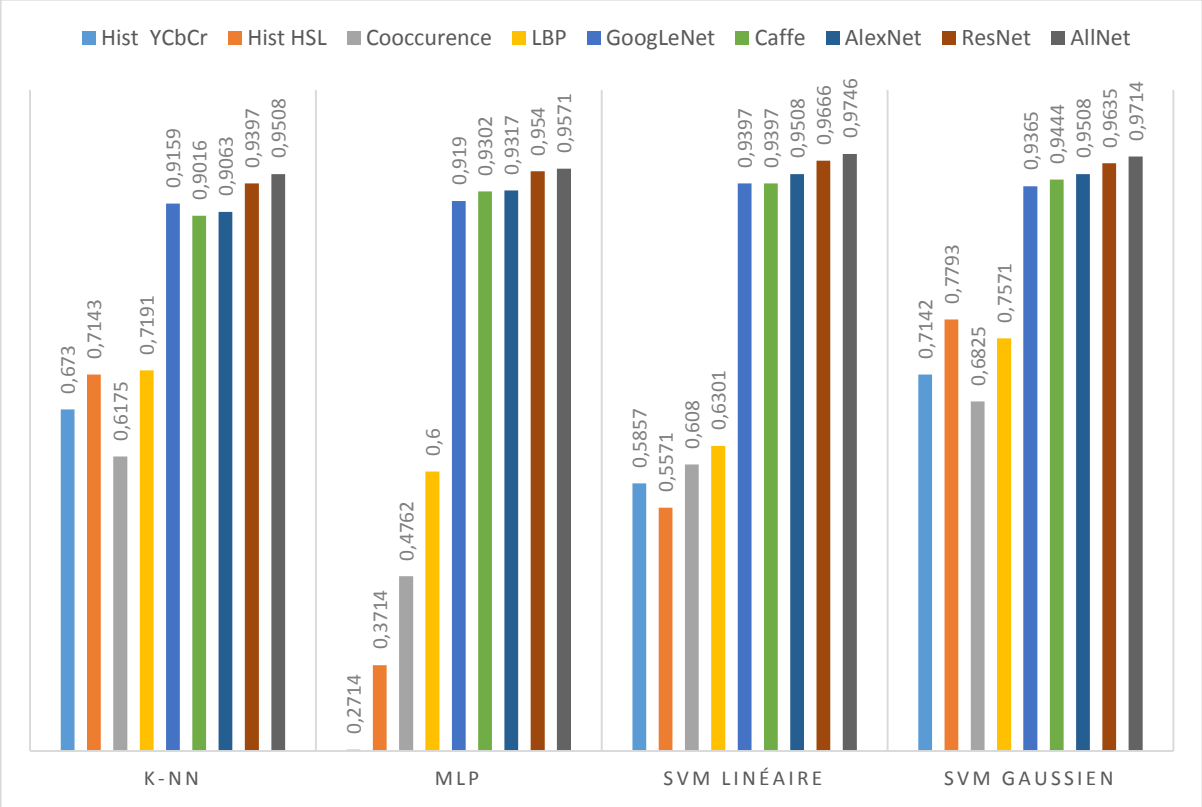


Figure 49 F-Score des codeurs classiques et convolutionels sur différents classifieurs appliqués au benchmark UCMerced LandUse

IV.3.3 Performances des codeurs avec la sélection *MSVM-RFE*

Dans cette expérience on tente de réduire le nombre d'attributs du codeur *AllNet* en utilisant l'élimination par *MSVM-RFE*, l'objectif est d'éliminer un maximum d'attributs sans dégrader les performances du codeur. On rappelle que le codeur *AllNet* est obtenu par concaténation des codeurs *GoogleNet*, *AlexNet*, *Caffe* et *ResNet*. On voudrait éventuellement mesurer la concentration des attributs restants (les plus pertinents) dans chacun de ces codeurs.

On commence l'expérience par la mesure du *F-Score* obtenu avec un *SVM* linaire sur l'ensemble complet des attributs du codeur *AllNet* puis :

1. Mesurer la pertinence de chaque attribut restant (par la mesure *MSVM-RFE*).
2. Trier les attributs du plus pertinent aux moins pertinent.
3. Eliminer un sous ensemble d'attributs (10 attributs dans notre expérience) dont la mesure de pertinence est la plus faible.
4. Refaire l'apprentissage du *SVM* linaire avec l'ensemble réduit des attributs.
5. Noter le *F-Score* obtenu ainsi que la distribution des attributs restants sur chacun des codeurs *GoogleNet*, *AlexNet*, *Caffe* et *ResNet*.
6. revenir à 1.

Les résultats de l'expérience (voir les courbes de la Figure 50) montrent que la sélection par *MSVM-RFE* a réussi à réduire le nombre d'attributs du codeur *AllNet* sans dégradation du *F-Score* à 2300 attributs, et avec une amélioration du *F-Score* (97,7%) pour 3400 attributs. Les courbes montrent que parmi les 3400 attributs restants du *AllNet* : 1250 appartiennent au codeur *AlexNet*, 1100 à *Caffe*, 550 à *GoogLeNet* et 500 à *ResNet*, et donc l'expérience n'a pas permis d'isoler un codeur majoritairement pertinent par rapport aux autres codeurs.

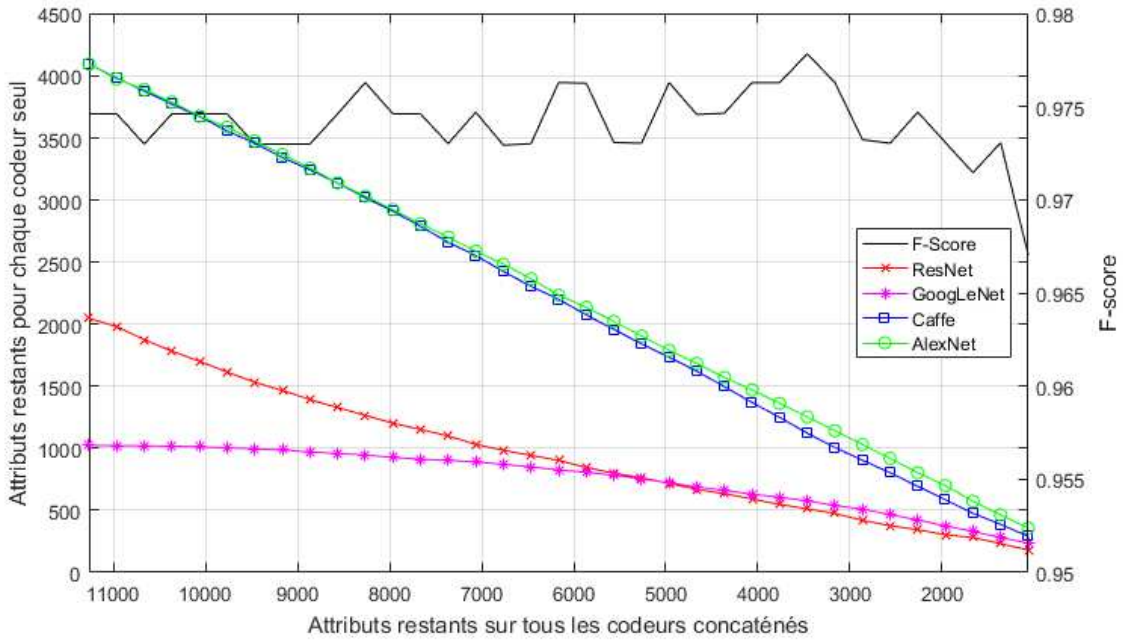


Figure 50 Courbe F-Score sur le benchmark UCMerced LandUse en fonction du nombre d'attributs restants du codeur AllNet après une sélection par MSVM-RFE

IV.3.4 Intégration de l'approche proposée

L'expérience suivante consiste à montrer l'apport de l'approche proposée pour la pondération et la sélection des attributs des différents codeurs à convolution en termes de performances de classification ainsi qu'en taux de réduction.

Afin de maintenir un temps d'exécution raisonnable, on a fixé empiriquement certains paramètres :

- La taille de la population a été fixée à 100 individus.
- Le taux de sélection de l'approche évolutionnaire est fixé à 25%.
- Après chaque génération, 25% des individus sélectionnés peuvent migrer vers d'autres îlots.
- Le nombre d'îlots est fixé empiriquement à 2 (50 individus par îlot).
- Le nombre maximum de générations est fixé à 100.
- Le taux d'élimination des attributs τ=1% (voir algorithme 1).

Dans cette expérience, les performances sont évaluées en termes de *F-Score* ainsi qu'en taux de réduction et en temps d'exécution. Quatre lancements ont été effectués pour chaque expérience et les moyennes des résultats des lancements sont reportées afin de réduire l'influence de l'aléatoire.

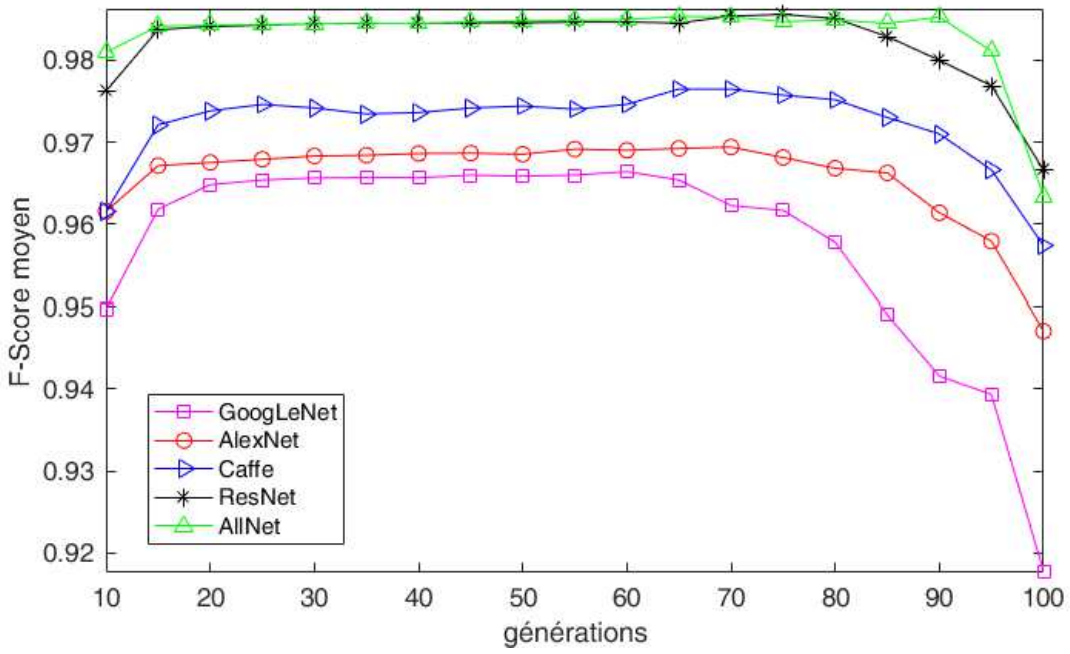


Figure 51 Moyennes des courbes du F-Score par génération pour chaque codeur avec une sélection/pondération par CGEDA+MSVM-RFE

L'expérience montre clairement (voir Figure 51) que les codeurs *ResNet* et *AllNet* surpassent en terme de *F-Score* les autres codeurs. Les courbes montrent une stabilité du *F-score* sur tous les codeurs jusqu'à 80 génération sachant qu'à chaque génération à partir de la 10^{ième}, 1% des attributs sont éliminés par la sélection *MSVM-RFE*. Ceci implique que l'approche proposée arrive à préserver un bon *F-Score* avec seulement 30% des attributs initiaux. Cependant, le codeur *GoogLeNet* initialement ne comprenant que 1024 attributs, à 30% il est réduit à seulement 306 attributs, on constate une dégradation rapide des performances au-delà des 80

générations. Les codeurs *AlexNet* et *Caffe* offrent une meilleure stabilité au-delà des 80 générations avec 4096 attributs au départ. On a constaté une légère supériorité du codeur *AllNet* par rapport à *ResNet* après 80 générations, vu que le codeur *AllNet* comprend 11264 attributs au départ il est moins affecté par l'élimination des attributs et offre le meilleur *F-Score* (98.74%) avec seulement 1492 attributs (voir Tableau 5).

Tableau 5 La moyenne des performances en F-Score, en taux de réduction et en temps d'exécution de l'approche proposée sur la classification des images aériennes

Codeur	*GoogLeNet*	*Caffe*	*AlexNet*	*ResNet*	*AllNet*
attributs	*1024*	*4096*	*4096*	*2048*	*11264*
Résultats des expériences : F-Score, Nombre d'attributs restants, temps d'exécution					
CGEDA Sans sélection	*0.9747 25min*	*0.9843 43min*	0.9764 *37min*	*0.9860 58min*	**0.9921** *165min*
CGEDA Avec sélection	*0.9381 965 25min*	*0.9575 3911 43min*	*0.9585 3894 37min*	*0.9682 1930 58min*	***98.09*** *10740 165min*
CGEDA avec sélection MSVM-RFE	*0.9719 434 17min*	*0.9796 1116 32min*	*0.9714 1205 28min*	*0.9858 563 38min*	***0.9874*** ***1492*** *112min*

Afin de montrer l'apport de la combinaison de l'optimisation *CGEDA* avec la sélection *MSVM-RFE*, on a effectué des lancements de l'optimisation *CGEDA* sans sélection d'attributs et *CGEDA* avec une sélection d'attributs basée seulement sur la fonction de fitness (voir Tableau 5). Cette expérience montre clairement un taux d'élimination nettement plus élevé avec la combinaison *CGEDA+MSVM-RFE* par rapport à la *CGEDA* seule sur tous les codeurs. On constate aussi un temps d'exécution plus réduit qui est dû à une réduction plus importante des attributs et donc du temps de calcul de la fitness basé sur l'apprentissage d'un *SVM* linéaire. Le meilleur *F-Score* (99.21%) a été obtenu avec une optimisation *CGEDA* sans sélection en utilisant les 11264 attributs (sur le codeur *AllNet*), cependant, la combinaison *CGEDA+MSVM-RFE* a permis d'obtenir un *F-Score* (98.74%) légèrement inférieur mais avec seulement 1492 attributs.

IV.3.5 Influence des stratégies de migration sur les performances

Dans cette expérience, on tente d'évaluer l'influence des stratégies de migration proposées sur les performances de l'implémentation parallèle de l'approche proposée en classification des images aériennes. On calcule pour chaque stratégie (*Best, Different, Radom, None*) et pour chaque génération, la moyenne du F-Score (Figure 52) obtenu ainsi que la diversité (Figure 53) sur tous les codeurs à convolution pré-entrainés.

La première constatation est que la stratégie de migration *None*, ou sans migration obtient les plus mauvais résultats en *F-Score* et en diversité. En termes de *F-Score*, les courbes des stratégies *Best* et *Random* sont les plus dominantes, en termes de diversité, les trois stratégies *Best*, *Random* et *Different* sont équivalentes avec une légère supériorité de la stratégie *Random*.

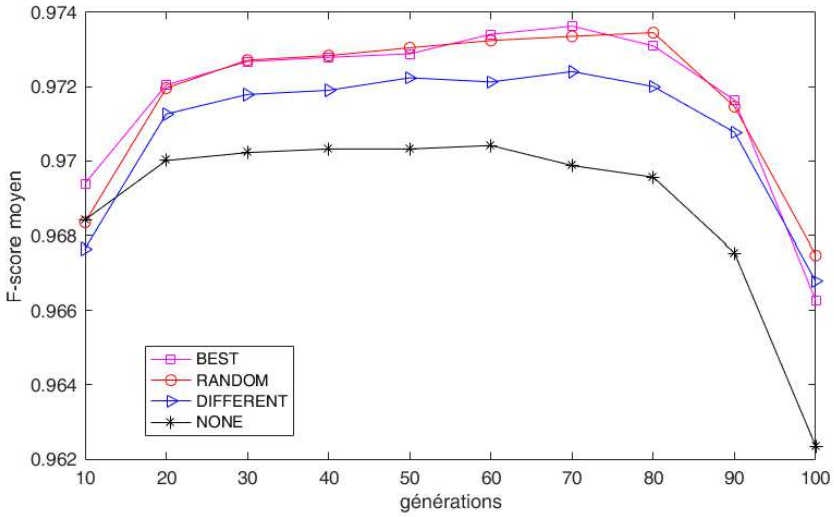


Figure 52 Moyennes des courbes du F-Score (sur tous les codeurs) par génération pour chaque stratégie de migration

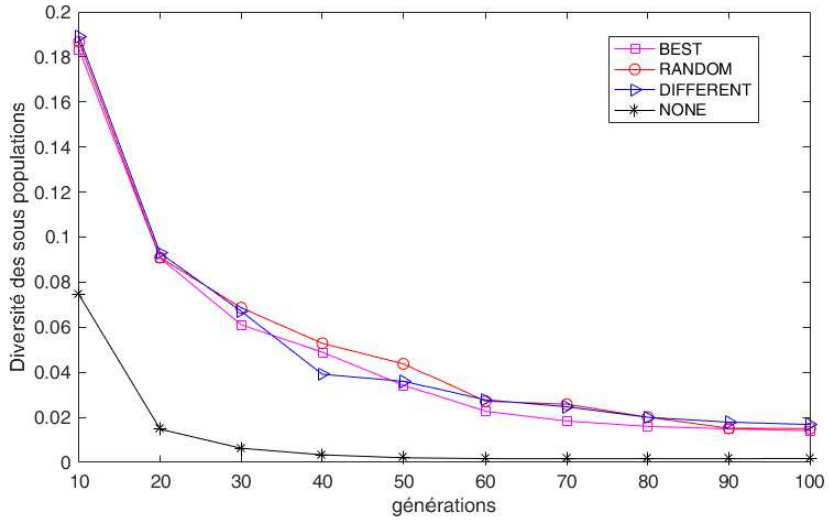


Figure 53 Moyennes des courbes de la diversité (sur tous les codeurs) par génération et pour chaque stratégie de migration

IV.3.6 Etude de l'accélération de l'implémentation parallèle proposée

L'objectif de cette étude est de déterminer le gain de temps obtenu avec la parallélisation de l'approche sur la classification des images aériennes, ainsi que l'accélération obtenue par rapport aux ressources matérielles allouées.

L'expérience (Tableau 6) consiste à exécuter l'approche proposée pour les 5 codeurs à convolution pré-entrainés en variant le nombre de workers (sans compter le *scheduler*) alloués de 1 (exécution non parallèle), 2, 4, 6 et 9 et de mesurer l'accélération obtenue. Les résultats montrent une nette réduction des temps d'exécutions pour tous les codeurs par rapport à une exécution non parallèle : de 104min à 17min pour *GoogLeNet*, de 198min à 32min pour *Caffe*, de 179min à 29min sur *AlexNet*, de 245min à 39min pour *ResNet* et de 680min à 112min sur *AllNet*.

Tableau 6 Temps d'éxecution moyen (en minutes) de l'approche proposée pour chaque codeur en fonction du nombre de workers aloués

		Nombre de workers				
Codeur	*Attributs*	*1*	*2*	*4*	*6*	*9*
GoogLeNet	*1024*	*104min*	*61min*	*40min*	*23min*	*17min*
Caffe	*4096*	*198min*	*115min*	*71min*	*44min*	*32min*
AlexNet	4096	179min	*105min*	56min	*40min*	*29min*
ResNet	2048	245min	*140min*	94min	*55min*	*39min*
AllNet	*11264*	680min	*389min*	277min	*166min*	*112min*

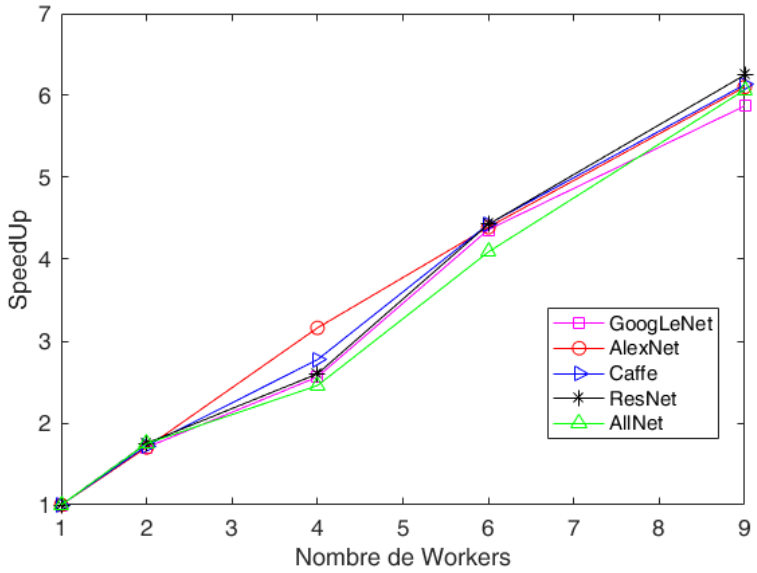


Figure 54 Accélération (Speedup) de l'implémentation parallèle proposée sur chaque codeur sur la base UCMerced LandUse en variant le nombre de workers

Les courbes de l'accélération (SpeedUp, Figure 54) de l'approche proposée pour chaque codeur montrent une accélération pseudo linéaire en fonction du nombre de workers alloués, qu'on a estimé avec l'équation 34. Cependant, cette estimation peut se révéler fausse pour un grand nombre de workers où le temps consommé en communications réseau risque d'accroitre et de réduire les performances. Les limites du matériel à disposition ne nous ont pas permis des tests avec plus de workers.

$$Speedup(k) \approx 0.625 \times (k-1) + 1 \qquad 34$$

Pour finir, une comparaison (Tableau 7) avec l'état de l'art de la base *UCMerced LandUse* montre que l'approche proposée est très compétitive par rapport aux résultats d'articles publiés récemment.

Tableau 7 Comparaison de l'approche proposée avec les résultats de l'état de l'art sur la base UCMerced LandUse

Article	*Meilleur Taux obtenu*	*Deep Learning ?*
(Yang & Newsam 2010)	77.71%	Non
(Negrel et al. 2014)	94.30%	Non
(Cheng et al. 2015)	91.46	Oui
(Luus et al. 2015)	93.48%	Oui
(Penatti et al. 2015)	**99.43%**	Oui
(Marmanis et al. 2016)	92.4%	Oui
(Wu et al. 2016)	92.7%	Oui
(Zou et al. 2016)	95.48%	Non
(Avramović & Risojević 2016)	92.35%	Non
(Cheng et al. 2016)	98.33%	Oui
(Scott et al. 2017)	98.50%	Oui
(Weng et al. 2017)	95.62	Oui
***CGEDA* sans sélection**	**99.21%**	Oui
CGEDA+SVMRFE	**98.74%**	Oui

IV.4 Conclusion

Dans ce chapitre, on a expérimenté l'approche mimétique parallèle proposée en sélection et pondération d'attributs pour la classification des documents Web et des images aériennes. On a utilisé un algorithme évolutionnaire *CGEDA*, en combinaison avec une approche de sélection locale *MSVM-RFE* afin de guider et d'accélérer la convergence. Les résultats montrent un gain considérable en performances de la classification comparés aux techniques classiques sur les benchmarks : *Reuters-21578*, *7Sectors*, *Webkb* et *UCMerced LandUse* tout en utilisant une représentation plus réduite des données. Cependant cette approche est très couteuse en temps de calcul à cause des multiples apprentissages du *SVM* nécessaires pour la convergence. Afin de résoudre cette problématique, on a réalisé une parallélisation de l'approche proposée via un modèle d'îlots. En divisant le calcul sur plusieurs workers dans un super calculateur fait maison, on a pu réduire avec succès le temps d'exécution. On a aussi étudié l'impact des différentes stratégies de migration sur la diversité des îlots et les performances obtenues.

Conclusion générale

Afin de conclure cette étude, on revient sur la notion d'apprentissage et de génération automatique de représentations discriminatives pour la classification. Comme nous l'avons signalé, une représentation adéquate des données est nécessaire afin d'optimiser les performances des classifieurs, surtout dans le cas de données complexes.

Dans ce contexte, on a commencé par montrer que derrière chaque donnée, il subsiste une structure cachée responsable de sa génération, et dont la découverte permet de mieux comprendre l'information contenue dans ces données et facilite son extraction. Cette structure cachée peut se formuler sous forme vectorielle, de séquences, d'arbres ou de graphes etc. La suite du chapitre est consacrée aux techniques standards de représentation des données textuelles ainsi qu'à la représentation des textures. L'une des limites principales des représentations usuelles est liée à sa dimensionnalité qui augmente rapidement avec le nombre de données disponibles. Des solutions sont discutées par la suite via les techniques classiques de réduction de dimensionnalité à base de variables latentes, ainsi que la sélection et la pondération des attributs. Un état de l'art des travaux récents en classification de documents Web et d'images aériennes est présenté à la fin du chapitre.

Dans le deuxième chapitre du manuscrit, on passe en revu différentes techniques de classification issues de l'apprentissage machine et appartenant à la famille des méthodes connexionnistes. Ce chapitre, nous a permis de comparer et de tirer les avantages et les inconvénients de chaque architecture, et de faire le lien entre ces techniques déjà très performantes avec les *SVMs* du point de vu structurelle. Les *SVMs*, permettent de pallier à plusieurs inconvénients qu'on trouve dans les modèles connexionnistes en imposant des contraintes sur les fonctions séparatrices, et de ce fait, constituent un choix parfait en tant que classifieur d'évaluation pour les représentations qu'on souhaite optimiser.

Le troisième chapitre aborde la problématique du point de vue optimisation, où on présente des techniques bio-inspirées puissantes qui permettent de trouver des solutions performantes dans un espace de recherche très complexe. Cependant, ces techniques (dites évolutionnaire en analogie avec l'évolution des espèces) sont très gourmandes en ressources matérielles. Afin de booster ces techniques, des solutions sont proposées via le paradigme mimétique et les modèles évolutionnaires parallèles. Ces techniques constituent des outils de choix afin d'optimiser la représentation des données complexes. La fin du chapitre est consacrée à la description algorithmique détaillée de la solution proposée, via l'hybridation d'une technique de sélection d'attribut locale (*MSVM-RFE*) avec une approche évolutionnaire simple de la famille des algorithmes à estimation de distribution (*CGEDA*), puis la parallélisation du processus via les modèles parallèles d'îlots.

Le quatrième chapitre montre l'efficacité de l'approche proposée, en premier par rapport aux techniques classiques, en deuxième par rapport aux performances en classification et en taux de réduction, et pour finir en rapport à la vitesse de l'algorithme et l'accélération obtenue.

Enfin, pour conclure cette thèse, on peut dire que, bien que l'approche proposée c'est révélée très compétitive par rapport aux techniques qu'on a pu lire en état de l'art, il reste beaucoup à faire afin de l'optimiser, notamment permettre la génération de représentation latente hiérarchiques, la comparaison des performances obtenues par les *SVM* avec d'autres classifieurs où bien tenter une parallélisation du côté GPU.

Références

Abdel Fattah, M., 2015. New term weighting schemes with combination of multiple classifiers for sentiment analysis. *Neurocomputing*, 167, pp.434–442.

Aghdam, M.H. & Heidari, S., 2015. Feature selection using Particle Swarm Optimization in text categorization. *Journal of Artificial Intelligence and Soft Computing Research*, 5(4), pp.231–238.

Akadi, A. El et al., 2009. A New gene selection approach based on Minimum Redundancy-Maximum Relevance (MRMR) and Genetic Algorithm (GA). In *IEEE/ACS International Conference on Computer Systems and Applications*. IEEE, pp. 69–75.

Amayri, O. & Bouguila, N., 2010. A study of spam filtering using support vector machines. *Artificial Intelligence Review*, 34(1), pp.73–108.

Anderson, C.W. et al., 1999. *Recurrent Neural Networks Design and Applications* L. R. Medsker & L. C. Jain, eds., CRC Press.

Avramović, A. & Risojević, V., 2016. Block-based semantic classification of high-resolution multispectral aerial images. *Signal, Image and Video Processing*, 10(1), pp.75–84.

Baluja, S., 1994. Population-based incremental learning. a method for integrating genetic search based function optimization and competitive learning, Technical Report *Carnegie Mellon University*.

Bengio, Y., Courville, A. & Vincent, P., 2013. Representation Learning: A Review and New Perspectives. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(8), pp.1798–1828.

Blei, D.M., Ng, A.Y. & Jordan, M.I., 2003. Latent dirichlet allocation. *The Journal of Machine Learning Research*, 3, pp.993–1022.

De Bonet, J.S., Isbell, C.L. & Viola, P., 1997. MIMIC: Finding Optima by Estimating Probability Densities. In *Advances in Neural Information Processing Systems*. pp. 424–430.

Brown, M.P. et al., 2000. Knowledge-based analysis of microarray gene expression data by using support vector machines. In *Proceedings of the National Academy of Sciences of the United States of America*. pp. 262–267.

Cantú-Paz, E., 1998. A survey of parallel genetic algorithms. *Calculateurs Paralleles, Reseaux et Systems Repartis*, 10(2), pp.141–171.

Chen, S., Billings, S.A. & Luo, W., 1989. Orthogonal least squares methods and their application to non-linear system identification. *International Journal of Control*, 50(5), pp.1873–1896.

Cheng, G. et al., 2015. Learning Coarse-to-Fine Sparselets for Efficient Object Detection and Scene Classification. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Cheng, G. et al., 2016. Scene classification of high resolution remote sensing images using convolutional neural networks. In *2016 IEEE International Geoscience and Remote Sensing Symposium (IGARSS)*. pp. 767–770.

Corcoran, A.L. & Wainwright, R.L., 1994. A parallel island model genetic algorithm for the multiprocessor scheduling problem. *Proceedings of the 1994 ACM symposium on Applied*

computing - SAC '94, 1, pp.483–487.

Costantea, I., Bot, R.I. & Wanka, G., 2008. Patent document classification based on mutual information feature selection. *Italian Journal of Pure and Applied Mathematics*, 23, pp.121–136.

Debole, F. & Sebastiani, F., 2003. Supervised term weighting for automated text categorization. In *Proceedings of the 2003 ACM symposium on Applied computing - SAC '03*. ACM Press, pp. 784–788.

Domeniconi, G. et al., 2016. A Comparison of Term Weighting Schemes for Text Classification and Sentiment Analysis with a Supervised Variant of tf.idf. In M. Helfert et al., eds. *Data Management Technologies and Applications: 4th International Conference, DATA 2015, Colmar, France, July 20-22, 2015,* Cham: Springer International Publishing, pp. 39–58.

Elman, J.L., 1990. Finding Structure in Time. *Cognitive Science*, 14(2), pp.179–211.

Escalante, H.J. et al., 2015. Term-weighting learning via genetic programming for text classification. *Knowledge-Based Systems*, 83, pp.176–189.

Etxeberria, R. & Larrañaga, P., 1999. Global optimization with Bayesian networks. In *In II Symposium on Artificial Intelligence. CIMAF99. Special Session on Distributions and Evolutionary Optimization*. Cuba, pp. 332–339.

Fine, S., Singer, Y. & Tishby, N., 1998. The Hierarchical Hidden Markov Model: Analysis and Applications. *Machine Learning*, 32(1), pp.41–62.

Forman, G., 2004. A pitfall and solution in multi-class feature selection for text classification. In *Twenty-first international conference on Machine learning - ICML '04*. New York, New York, USA: ACM Press, p. 38.

Furey, T.S. et al., 2000. Support vector machine classification and validation of cancer tissue samples using microarray expression data. *Bioinformatics*, 16(10), pp.906–914.

Garson, J., 2016. Connectionism. In E. N. Zalta, ed. *The Stanford Encyclopedia of Philosophy*. Metaphysics Research Lab, Stanford University.

Ghareb, A.S., Bakar, A.A. & Hamdan, A.R., 2016. Hybrid feature selection based on enhanced genetic algorithm for text categorization. *Expert Systems with Applications*, 49, pp.31–47.

Goldberg, D.E., 1989. *Genetic Algorithms in Search, Optimization, and Machine Learning* Addison-Wesley, ed., Addison-Wesley.

Goldberg, D.E. & Holland, J.H., 1988. Genetic Algorithms and Machine Learning. *Machine Learning*, 3(2–3), pp.95–99.

Grabczewski, K. & Jankowski, N., 2005. Feature selection with decision tree criterion. In *Fifth International Conference on Hybrid Intelligent Systems*. IEEE, pp. 212–217.

Guyon, I. et al., 2002. Gene Selection for Cancer Classification using Support Vector Machines. *Machine Learning*, 46(1–3), pp.389–422.

Hagenbuchner, M., Sperduti, A. & Tsoi, A.C., 2003. A self-organizing map for adaptive processing of structured data. *IEEE Transactions on Neural Networks*, 14(3), pp.491–505.

Hardy, R.L., 1971. Multiquadric Equations of Topography and Other Irregular Surfaces. *J. Geophys. Res.*, 76, pp.1905–1915.

Hassan, S., Mihalcea, R. & Banea, C., 2007. Random-Walk Term Weighting for Improved Text Classification. *International Journal of Semantic Computing*, 1(4), pp.421–439.

He, K. et al., 2016. Deep Residual Learning for Image Recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, pp. 770–778.

Hinton, G.E., Osindero, S. & Teh, Y.-W., 2006. A fast learning algorithm for deep belief nets. *Neural computation*, 18(7), pp.1527–54.

Hochreiter, S. & Schmidhuber, J., 1997. Long Short-Term Memory. *Neural Computation*, 9(8), pp.1735–1780.

Hofmann, T., 1999. Probabilistic latent semantic analysis. In *Proceedings of the Fifteenth Conference on Uncertainty in Artificial Intelligence*. Stockholm, Sweden: Morgan Kaufmann Publishers Inc., pp.289–296.

Hornik, K., 1991. Approximation capabilities of multilayer feedforward networks. *Neural Networks*, 4(2), pp.251–257.

Javed, K., Maruf, S. & Babri, H.A., 2015. A two-stage Markov blanket based feature selection algorithm for text classification. *Neurocomputing*, 157, pp.91–104.

Jia, Y. et al., 2014. Caffe: Convolutional Architecture for Fast Feature Embedding. *arXiv preprint arXiv:1408.5093*.

Joachims, T., 2002. *Learning to Classify Text Using Support Vector Machines: Methods, Theory and Algorithms*, Kluwer.

Joachims, T., 1998. Text categorization with Support Vector Machines: Learning with many relevant features. In C. Nédellec & C. Rouveirol, eds. *Machine Learning: ECML-98*. Lecture Notes in Computer Science. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 137–142.

Jordan, M.I., 1997. Chapter 25 - Serial Order: A Parallel Distributed Processing Approach. In J. W. Donahoe & V. P. Dorsel, eds. *Neural-Network Models of CognitionBiobehavioral Foundations*. Advances in Psychology. North-Holland, pp. 471–495.

Karaboga, D. & Basturk, B., 2007. A powerful and efficient algorithm for numerical function optimization: Artificial bee colony (ABC) algorithm. *Journal of Global Optimization*, 39(3), pp.459–471.

Kaumanns, F.D., 2016. *Assessment and Analysis of the Applicability of Recurrent Neural Networks to Natural Language Understanding with a Focus on the Problem of Coreference Resolution*. Thèse pour l'obtention du doctorat en philosophie à l'université Ludwig Maximilian de Munich.

Koza, J.R., 1994. Genetic programming as a means for programming computers by natural selection. *Statistics and Computing*, 4(2), pp.87–112.

Krizhevsky, A., Sutskever, I. & Hinton, G.E., 2012. ImageNet Classification with Deep Convolutional Neural Networks. In *Proceedings of the 25th International Conference on Neural Information Processing Systems - Volume 1*. NIPS'12. USA: Curran Associates Inc., pp. 1097–1105.

Lamirel, J.-C. et al., 2015. Optimizing text classification through efficient feature selection based on quality metric. *Journal of Intelligent Information Systems*, 45(3), pp.379–396.

Larrañaga, P. & Lozano, J.A., 2002. *Estimation of Distribution Algorithms: A New Tool for Evolutionary Computation*, Kluwer Academic Publishers.

Lawrence, S., Tsoi, A.C. & Back, A.D., 1996. The Gamma MLP for Speech Phoneme Recognition. In D. S. Touretzky, M. C. Mozer, & M. E. Hasselmo, eds. *Advances in Neural*

Information Processing Systems 8. MIT Press, pp. 785–791.

Lecun, Y. et al., 1998. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11), pp.2278–2324.

Lozano, J.A., Sagarna, R. & Larra Naga, P., 2002. Parallel estimation of distribution algorithms. In P. Larrañaga & J. A. Lozano, eds. *Towards a New Evolutionary Computation*. Kluwer Academic Publishers, pp. 125–142.

Luus, F.P.S. et al., 2015. Multiview Deep Learning for Land-Use Classification. *IEEE Geoscience and Remote Sensing Letters*, 12(12), pp.2448–2452.

Madera, J., Alba, E. & Ochoa, a, 2006. A Parallel Island Model for Estimation of Distribution Algorithms. *Towards a New Evolutionary Computation*, 186(2006), pp.159–186.

Marmanis, D. et al., 2016. Deep Learning Earth Observation Classification Using ImageNet Pretrained Networks. *IEEE Geoscience and Remote Sensing Letters*, 13(1), pp.105–109.

Mary Amala Bai, V. & Manimegalai, D., 2013. A Document Level Measure for Text Categorization. *International Review on Computers and Software (IRECOS)*, 8(6), pp.1374–1381.

Minsky, M.L. & Papert, S., 1972. *Perceptrons: An Introduction to Computational Geometry*, Mit Press.

Mühlenbein, H., 1997. The Equation for Response to Selection and Its Use for Prediction. *Evolutionary Computation*, 5(3), pp.303–346.

Mühlenbein, H. & Paass, G., 1996. From Recombination of Genes to the Estimation of Distributions I. Binary Parameters. In *Proceedings of the 4th International Conference on Parallel Problem Solving from Nature*. Springer-Verlag, pp. 178–187.

Narasimhan, H., 2009. Parallel artificial bee colony (PABC) algorithm. In *World Congress on Nature and Biologically Inspired Computing, NABIC 2009 - Proceedings*. pp. 306–311.

Negrel, R., Picard, D. & Gosselin, P.H., 2014. Evaluation of second-order visual features for land-use classification. In *2014 12th International Workshop on Content-Based Multimedia Indexing (CBMI)*. pp. 1–5.

Neri, F. & Cotta, C., 2012. Memetic algorithms and memetic computing optimization: A literature review. *Swarm and Evolutionary Computation*, 2, pp.1–14.

Ojala, T., Pietikäinen, M. & Mäenpää, T., 2002. Multiresolution gray-scale and rotation invariant texture classification with local binary patterns. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(7), pp.971–987.

Paul, S. & Das, S., 2015. Simultaneous feature selection and weighting - An evolutionary multi-objective optimization approach. *Pattern Recognition Letters*, 65, pp.51–59. Available at: http://dx.doi.org/10.1016/j.patrec.2015.07.007.

Pedemonte, M., Nesmachnow, S. & Cancela, H., 2011. A survey on parallel ant colony optimization. *Applied Soft Computing Journal*, 11(8), pp.5181–5197.

Pelikan, M., Goldberg, D.E. & Cantú-Paz, E., 1999. BOA: The Bayesian Optimization Algorithm. *Genetic and Evolutionary Computation*, 1, pp.525–532.

Penatti, O.A.B., Nogueira, K. & dos Santos, J.A., 2015. Do deep features generalize from everyday objects to remote sensing and aerial scenes domains? In *2015 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*. pp. 44–51.

Pérez-Rodríguez, J., Arroyo-Peña, A.G. & García-Pedrajas, N., 2015. Simultaneous instance and feature selection and weighting using evolutionary computation: Proposal and study. *Applied Soft Computing Journal*, 37, pp.416–443.

Perkins, S., Lacker, K. & Theiler, J., 2003. Grafting: fast, incremental feature selection by gradient descent in function space. *The Journal of Machine Learning Research*, 3, pp.1333–1356.

Poli, R., Kennedy, J. & Blackwell, T., 2007. Particle swarm optimization. *Swarm Intelligence*, 1(1), pp.33–57.

Powell, M.J.D., 1985. Radial Basis Functions for Multivariable Interpolation: A Review. In *IMA conference Algorithms for approximation of functions and data*. Oxford: Clarendon Press, pp. 143–167.

Rechenberg, I., 1989. Evolution strategy: Nature's way of optimization. In H. W. Bergmann, ed. *Optimization: Methods and applications, possibilities and limitations*. Springer, pp. 106–126.

Ruciński, M., Izzo, D. & Biscani, F., 2010. On the impact of the migration topology on the Island Model. *Parallel Computing*, 36(10–11), pp.555–571.

Rumelhart, D.E., McClelland, J.L. & PDP Research Group, C. eds., 1986. *Parallel Distributed Processing: Explorations in the Microstructure of Cognition, Vol. 1: Foundations*, Cambridge, MA, USA: MIT Press.

Salton, G. & Buckley, C., 1988. Term-weighting approaches in automatic text retrieval. *Information Processing & Management*, 24(5), pp.513–523.

Scott, G.J. et al., 2017. Training Deep Convolutional Neural Networks for Land-Cover Classification of High-Resolution Imagery. *IEEE Geoscience and Remote Sensing Letters*, 14(4), pp.549–553.

Senliol, B. et al., 2008. Fast Correlation Based Filter (FCBF) with a different search strategy. In *23rd International Symposium on Computer and Information Sciences*.

Shahraki, S. & Tutunchy, M.R.A., 2013. Continuous Gaussian Estimation of Distribution Algorithm. In R. Kruse et al., eds. *Synergies of Soft Computing and Statistics for Intelligent Data Analysis*. Advances in Intelligent Systems and Computing. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 211–218.

Shi, Y. et al., 2010. Fast implementation of string-kernel-based support vector classifiers by GPU computing. In *Neural Information Processing. Models and Applications: 17th International Conference, ICONIP 2010*. Springer Berlin Heidelberg, pp. 143–151.

Song, F., Liu, S. & Yang, J., 2005. A comparative study on text representation schemes in text categorization. *Pattern Analysis and Applications*, 8(1–2), pp.199–209.

Specht, D.F., 1990. Probabilistic neural networks. *Neural Networks*, 3(1), pp.109–118.

Srivastava, N. et al., 2014. Dropout: A Simple Way to Prevent Neural Networks from Overfitting. *J. Mach. Learn. Res.*, 15(1), pp.1929–1958.

Stoppiglia, H. et al., 2003. Ranking a Random Feature for Variable and Feature Selection. *Journal of Machine Learning Research*, 3, pp.1399–1414.

Sudholt, D., 2015. Parallel Evolutionary Algorithms. In *Springer Handbook of Computational Intelligence*. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 929–959.

Sun, R., 2001. Introduction to sequence learning. *Sequence Learning*, pp.1–10.

Szegedy, C. et al., 2015. Going deeper with convolutions. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. pp. 1–9.

Tan, J., Li, W. & Li, H., 2014. Automated text categorization by generalized kernel machines. In *IEEE International Conference on Information and Automation (ICIA)*. IEEE, pp. 376–381.

Tang, J., Lim, M.H. & Ong, Y.S., 2006. Diversity-adaptive parallel memetic algorithm for solving large scale combinatorial optimization problems. *Soft Computing*, 11(9), pp.873–888.

Vanneschi, L., Codecasa, D. & Mauri, G., 2011. A comparative study of four parallel and distributed PSO methods. *New Generation Computing*, 29(2), pp.129–161.

Vapnik, V.N., 1995. *The Nature of Statistical Learning Theory*, Springer.

de Vries, B. & Principe, J.C., 1992. The gamma model—A new neural model for temporal processing. *Neural Networks*, 5(4), pp.565–576.

Waibel, A. et al., 1989. Phoneme recognition using time-delay neural networks. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 37(3), pp.328–339.

Wang, T. et al., 2015. Entropy-Based Term Weighting Schemes for Text Categorization in VSM. In *2015 IEEE 27th International Conference on Tools with Artificial Intelligence (ICTAI)*. pp. 325–332.

Weng, Q. et al., 2017. Land-Use Classification via Extreme Learning Classifier Based on Deep Convolutional Features. *IEEE Geoscience and Remote Sensing Letters*, 14(5), pp.704–708.

Wu, H. et al., 2016. Deep Filter Banks for Land-Use Scene Classification. *IEEE Geoscience and Remote Sensing Letters*, 13(12), pp.1895–1899.

Yang, S. & Ramanan, D., 2015. Multi-scale recognition with DAG-CNNs. *Proceedings of the IEEE International Conference on Computer Vision*, 2015 Inter, pp.1215–1223.

Yang, Y. & Newsam, S., 2010. Bag-of-visual-words and Spatial Extensions for Land-use Classification. In *Proceedings of the 18th SIGSPATIAL International Conference on Advances in Geographic Information Systems*. GIS '10. New York, NY, USA: ACM, pp. 270–279.

Yang, Y. & Pedersen, J.O., 1997. A Comparative Study on Feature Selection in Text Categorization. In *ICML '97 Proceedings of the Fourteenth International Conference on Machine Learning*. Morgan Kaufmann Publishers Inc., pp. 412–420.

Yu, F. & Koltun, V., 2015. Multi-Scale Context Aggregation by Dilated Convolutions. *ArXiv e-prints*.

Zhang, L. & Huang, X., 2015. Multiple SVM-RFE for multi-class gene selection on DNA Microarray data. In *International Joint Conference on Neural Networks (IJCNN)*. pp. 1–6.

Zhang, S.X. et al., 2016. Recurrent support vector machines for speech recognition. *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*.

Zou, J. et al., 2016. Scene classification using local and global features with collaborative representation fusion. *Information Sciences*, 348, pp.209–226.

Résumé

Dans cette thèse, on propose une approche mimétique pour l'apprentissage et la génération de représentations optimisées de données complexes à travers la sélection et la pondération simultanées des attributs basées sur une hybridation entre une approche évolutionnaire et l'apprentissage sous contraintes des machines à vecteur de support. Cette technique a été expérimentée sur l'optimisation de la classification des documents web ainsi que des images aériennes. Cependant, les représentations usuelles des données complexes engendrent des matrices de très grandes dimensionnalités dont le traitement par une approche mimétique peut s'avérer très lourd en temps de calcul lors de la phase d'apprentissage. Dans ce travail, on propose une implémentation parallèle de l'algorithme proposé basée sur les modèles d'îlots afin de palier à ce problème. Nos expériences sur plusieurs benchmarks : *Reuters-21578*, *7Sectors*, *Webkb* et *UCMerced LandUse* ont montré qu'on a réduit significativement le temps d'exécution ainsi que le nombre d'attributs avec une nette amélioration des performances en classification.

Mots clés : Algorithme Mimétiques Parallèles, Classification du Texte, Images Aériennes, Sélection et Pondération d'Attributs, Machines à Vecteurs de Support, Apprentissage profond.

Abstract

In this thesis, we propose a mimetic approach for learning and generating optimized complex data representations through simultaneous weighting and feature selection based on a combination between an evolutionary algorithm and Support Vector Machines to improve web documents and aerial images categorization. However, usual representations generate high dimensionality matrix, which processed with a mimetic approach, may increase drastically the processing time in training step. In this work, we present a parallel implementation of the proposed algorithm based on an Island model topology. Experiments on well-known datasets: *Reuters-21578, 7Sectors, Webkb* and *UCMerced LandUse* show that we have significantly reduced the set of features and computing time while improving categorization performance.

Keywords: Parallel Mimetic Algorithm, Text Categorization, Aerial Images, Feature Selection & Weighting, Support Vector Machine, Deep Learning.