Optimisation de la Topologie pour les Réseaux de Neurones Profonds

Ludovic Arnold, Hélène Paugam-Moisy, Michèle Sebag

Optimisation de la Topologie pour les Réseaux de Neurones Profonds

Ludovic ARNOLD[1] Hélène PAUGAM-MOISY[2] Michèle SEBAG[1]

[1] TAO – INRIA, CNRS, LRI
[2] Université de Lyon – TAO, INRIA

TAO-INRIA, LRI-CNRS, bât. 490, Université Paris-Sud, F-91405 Orsay, France
Ludovic.Arnold@lri.fr, Helene.Paugam-Moisy@univ-lyon2.fr, Michele.Sebag@lri.fr

Résumé

Récemment, les réseaux de neurones profonds ont démontré leur capacité à obtenir d'excellentes performances pour des tâches de classification et de réduction de dimension. Le problème du choix des hyper-paramètres est très critique pour ces réseaux car la taille de l'espace de recherche augmente exponentiellement avec le nombre de couches. En conséquence, l'approche grid-search *ne convient pas et c'est à l'expérimentateur de "deviner" des valeurs judicieuses pour les hyper-paramètres.*

Nous proposons ici de sélectionner les hyper-paramètres couche après couche, sur la base d'un critère non-supervisé, ce qui réduit à une complexité linéaire l'optimisation des hyper-paramètres. Deux critères non-supervisés sont envisagés et nous focalisons notre étude sur la détermination d'un nombre optimal de neurones par couche. Au cours de nos expériences, nous montrons que l'erreur de reconstruction constitue un critère satisfaisant pour l'optimisation, couche par couche, du nombre de neurones. Nous observons par ailleurs que la taille optimale des couches tend à diminuer lorsque le nombre d'exemples augmente et nous discutons ce comportement contre-intuitif.

Mots Clef

Apprentissage, Réseaux Profonds, Sélection de Modèle.

Abstract

Recently, deep neural networks have proven their ability to achieve excellent results on tasks such as classification and dimensionality reduction. The issue of hyper-parameter selection is decisive for these networks since the size of the search space increases exponentially with the number of layers. As a result, the grid-search approach is inappropriate and it is often left to the experimenter to "guess" sensible values for the hyper-parameters.

In this study, we propose to select hyper-parameters layer after layer, on the basis of an unsupervised criterion, thus reducing to linear the complexity of the hyper-parameter selection procedure. Two unsupervised criteria are considered and the study focuses on determining an optimal number of neurons per layer. Experimentally, we show that the reconstruction error constitutes an adequate criterion for the layer-wise optimization of the number of neurons. In addition, we observe that the optimal size of layers tends to decrease when the number of training samples increases and we discuss this counter-intuitive result.

Keywords

Learning, Deep Neural Networks, Model Selection.

1 Introduction

Les réseaux de neurones profonds, ou *Deep Neural Networks*, ne sont autres que des réseaux multicouches, d'architecture classique, mais ils comportent plusieurs couches cachées et c'est la manière de gérer leur apprentissage qui a donné, depuis 2006, un regain d'intérêt à leur étude. En effet, même si les théorèmes d'approximation de la fin des années 80 [7, 11] affirment que, en théorie, une seule couche cachée suffit pour l'approximation de n'importe quelle fonction suffisamment régulière, rien n'empêche a priori de mettre en œuvre un apprentissage par rétro-propagation dans des réseaux comportant plusieurs couches cachées. D'autre part, certains résultats [5, 12, 18] mettent en évidence l'intérêt de considérer deux ou plusieurs couches cachées pour obtenir des réseaux plus parcimonieux et plus efficaces, en composant plusieurs niveaux de non-linéarités. Cependant, l'apprentissage par rétro-propagation s'est souvent révélé inefficace dans les réseaux à plusieurs couches, en raison des minima locaux, souvent assez mauvais, dans lesquels la descente en gradient piégeait la méthode.

Pour toutes ces raisons, certains chercheurs [9, 3] ont proposé de nouvelles méthodes d'apprentissage, le plus souvent couche par couche, pour dépasser les limitations pratiques de la rétro-propagation et pour mieux exploiter les potentialités de représentation interne des réseaux dits profonds. L'inconvénient des architectures superficielles (*shallow architectures*), auxquelles n'échappent pas les SVM [1], a été dénoncé et argumenté par Bengio et LeCun [4]. Bengio et al [3] ont proposé un algorithme d'apprentis-

[1] SVM = *Support Vector Machine*, ou Séparteur à Vaste Marge, selon une certaine terminologie française.

sage glouton, basé sur un empilement d'auto-associateurs, qui permet de construire les couches cachées l'une après l'autre. Hinton et al [9] sont repartis du modèle de la machine de Boltzmann [1] pour définir un empilement de machines de Boltzmann restreintes, ou *Restricted Boltzmann Machines, RBMs*, pour construire des *Deep Belief Networks*. La recherche internationale est actuellement très active autour de l'étude de méthodes d'apprentissage permettant de mieux contrôler la construction, dans les couches cachées successives, de représentations internes par extraction de caractéristiques présentant des niveaux d'abstraction de plus en plus élevés [2, 16, 14, 15]. Les réseaux de neurones profonds se sont déjà montrés très performants dans différent domaines dont la reconnaissance d'image, le traitement du langage et la robotique.

Notre objectif est de définir une méthode automatique pour optimiser les hyper-paramètres d'un réseau de neurones profond, et en particulier ceux qui contrôlent sa topologie : le nombre de couches cachées et le nombre de neurones dans chacune de ces couches cachées. Le problème du choix des hyper-paramètres, déjà crucial pour les MLP [2], est encore plus critique pour les réseaux de neurones profonds. En effet, chaque couche possède ses propres hyper-paramètres, donc la taille de l'espace de recherche augmente exponentiellement avec le nombre de couches. En conséquence, une approche *grid-search* nécessite un temps de calcul si important que l'expérimentateur se contente en général de "deviner" des valeurs judicieuses pour les hyper-paramètres. Formellement, si l'on veut tester k tailles potentielles pour chacune des n couches cachées, on est amené à tester k^n topologies différentes. En nous appuyant sur le fait que, dans les réseaux profonds, les apprentissages sont dissociés, couche par couche, nous proposons de rechercher un nombre de neurones optimal pour chacune des couches cachées avant de procéder à l'apprentissage de la couche suivante, ce qui réduit la cardinalité de l'espace de recherche à kn. Pour cette optimisation, nous proposons deux critères non-supervisés, et nous focalisons notre étude sur la détermination d'un nombre optimal de neurones par couche. Le modèle que nous avons mis en œuvre se base sur l'approche RBM, mais pourrait sans difficulté se généraliser aux Auto-Associateurs empilés.

L'article s'articule de la manière suivante : nous décrivons succinctement le modèle des machines de Boltzmann restreintes empilées (*stacked RBMs*) ; nous discutons du bienfondé d'une optimisation couche par couche et nous définissons deux critères : une mesure d'énergie et une mesure d'erreur ; nous présentons les résultats des expériences réalisées avec un algorithme d'apprentissage rapide et enfin nous discutons un comportement surprenant que nous avons observé : la taille optimale des couches tend à diminuer lorsque le nombre d'exemples appris augmente.

[2] MLP = *Multi-Layer Perceptron*, réseau multicouche à apprentissage par rétro-propagation.

2 Réseaux de neurones profonds

Nous présentons ici le modèle de réseau profond proposé par Hinton et al. [9] en 2006, modèle basé sur un empilement de machines de Boltzmann restreintes. Nous précisons ci-après la topologie et la méthode d'apprentissage utilisées dans ce modèle. Pour mémoire, la machine de Boltzmann est un modèle de réseau de neurones qui a été défini en 1985 par Ackley, Hinton et Sejnowski [1]. Ce réseau comporte des unités visibles et des unités cachées, complétement interconnectées, et son apprentissage est basé sur des évaluations statistiques, par méthode de recuit simulé [13], lors de deux phases distinctes : éveil (sous l'influence de stimuli d'entrée) et repos (relaxation en système fermé).

2.1 Machine de Boltzmann Restreinte

Une machine de Boltzmann est dite restreinte lorsque ses connexions sont limitées à un sous-ensemble strict de toutes les connexions possibles, comme défini ci-dessous :

Définition 1 *Une* Machine de Boltzmann Restreinte *(RBM) est un graphe biparti, non-orienté et pondéré. Ses noeuds sont appelés unités et ses arêtes connexions. Les deux parties de ce graphe sont respectivement appelées couche visible (*$\mathbf{v}$*) et couche cachée (*$\mathbf{h}$*).*

Une représentation de machine de Boltzmann restreinte est donnée sur la droite de la figure 1.

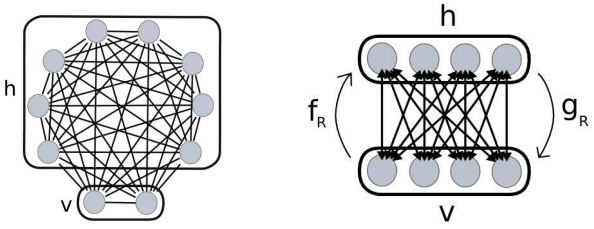

FIG. 1 – À gauche : Topologie d'une machine de Boltzmann classique. À droite : Topologie d'une machine de Boltzmann restreinte (RBM)

Définition 2 *La* configuration d'une RBM *est un étiquetage binaire d'une Machine de Boltzmann Restreinte. Dans la suite, on note* $\mathbf{v} = v_1, \ldots, v_q$ *et* $\mathbf{h} = h_1, \ldots, h_r$ *les restrictions de cet étiquetage aux unités de la couche visible et de la couche cachée respectivement. Chaque unité peut être étiquetée par la valeur* 1 *ou la valeur* 0*. Dans le premier cas, elle est dite* active*, sinon elle est* inactive*.*

Pour simplifier les notations, on suppose qu'une unité de biais, toujours active, est présente dans la couche visible ainsi que dans la couche cachée. En notant w_{ij} le poids de la connexion entre les unités v_j et h_i et $\mathbf{W} = (w_{ij})$ la matrice de ces poids, on peut alors définir l'énergie d'une machine de Boltzmann restreinte par :

$$Energy(\mathbf{v}, \mathbf{h}) \; = \; -\mathbf{h}^\top \mathbf{W} \mathbf{v} \qquad (1)$$

Dotée de cette fonction d'énergie, la machine de Boltzmann restreinte peut être vue comme un réseau à satisfaction de contraintes. Chaque unité correspond alors à une hypothèse du domaine à laquelle on affecte une valeur de vérité : vrai (1) ou faux (0). Les connexions représentent des contraintes sur ces hypothèses : si le poids w_{ij} est positif, alors l'activation simultanée des deux unités fait baisser l'énergie. À l'inverse, si w_{ij} est négatif, il faut que i et j ne soient pas activées en même temps pour que l'énergie diminue. En définitive, l'énergie liée à une configuration $(\mathbf{v}, \mathbf{h})$ est d'autant plus basse que cette configuration respecte mieux les contraintes imposées par les poids.
De plus, la fonction d'énergie permet d'associer à une machine de Boltzmann restreinte, de poids $\mathbf{W}$, une mesure de probabilité sur l'espace des configurations :

$$P_W(\mathbf{v}, \mathbf{h}) = \frac{e^{-Energy(\mathbf{v},\mathbf{h})}}{Z} \tag{2}$$

avec $Z = \sum_{v,h} e^{-Energy(\mathbf{v},\mathbf{h})}$ le facteur de normalisation approprié pour que P_W somme à 1 sur l'ensemble des configurations.
Par construction de la fonction d'énergie, les probabilités d'activation des unités v_i sont indépendantes sachant $\mathbf{h}$ et réciproquement, les h_i sont indépendants sachant $\mathbf{v}$. Soit $sgm(t) = \frac{1}{1+e^{-t}}$ une fonction sigmoïde, les probabilités conditionnelles pour une machine de Boltzmann restreinte sont données par :

$$\forall i \leq r, \;\; P_W(h_i|\mathbf{v}) = sgm(\sum_j w_{ij} v_j) \tag{3}$$

$$\forall j \leq q, \;\; P_W(v_j|\mathbf{h}) = sgm(\sum_i w_{ij} h_i) \tag{4}$$

Les machines de Boltzmann restreintes constituent donc un modèle probabiliste à part entière : une RBM modélise la probabilité d'une entrée $\mathbf{v}$ avec la probabilité jointe $P_W(\mathbf{v}, \mathbf{h})$. Il est possible de réaliser un tirage aléatoire selon cette distribution, c'est-à-dire de générer des configurations $(\mathbf{v}, \mathbf{h})$ et donc des exemples crédibles d'entrées $\mathbf{v}$ à partir du modèle. Pour ceci, on utilise la technique du *Gibbs sampling*. Cette technique est une méthode de Monte Carlo par chaîne de Markov ; il s'agit de faire itérativement des tirages aléatoires dans $P_W(\mathbf{h}|\mathbf{v})$ et $P_W(\mathbf{v}|\mathbf{h})$ et l'on a ainsi la garantie que la chaîne de Markov converge vers la distribution P_W.
Soit P_D la distribution de probabilités correspondant au tirage aléatoire d'un exemple $\mathbf{v}$ dans une base d'entrainement D, puis au tirage aléatoire d'une représentation cachée $\mathbf{h}$ associée à $\mathbf{v}$. P_D est la distribution objectif de l'apprentissage, elle est fixée par la base d'exemples. La distribution P_W quant à elle est la distribution modélisée par la RBM, indépendamment des entrées, et selon laquelle on peut faire des tirages aléatoires avec la méthode du *Gibbs sampling*.
L'objectif de l'apprentissage est de minimiser la divergence de Kullback Leibler entre ces deux distributions $KL(P_D||P_W)$. Si cette divergence est nulle, la probabilité de tirer un exemple $\mathbf{v}$ dans la base d'exemple sera identique à celle de le générer à partir du RBM entrainé.

La minimisation exacte de la divergence de Kullback Leibler étant très coûteuse, on minimise un critère approché : *Contrastive Divergence* [8] avec une approximation en champ moyen [21].

2.2 Réseau profond par RBM empilées

Les machines de Boltzmann restreintes empilées (*stacked RBMs*) sont un type particulier de réseau de neurones profond basé sur les Machines de Boltzmann Restreintes. La topologie d'un tel réseau est présentée sur la figure 2.

L'entrainement des RBM empilées se fait couche par couche, de manière gloutonne. Une première RBM est entrainée sur la base d'exemples de manière à minimiser la divergence de Kullback Leibler $KL(P_W|P_D)$. Chacune des RBM suivantes effectue son apprentissage sur les représentations cachées de la RBM précédente.

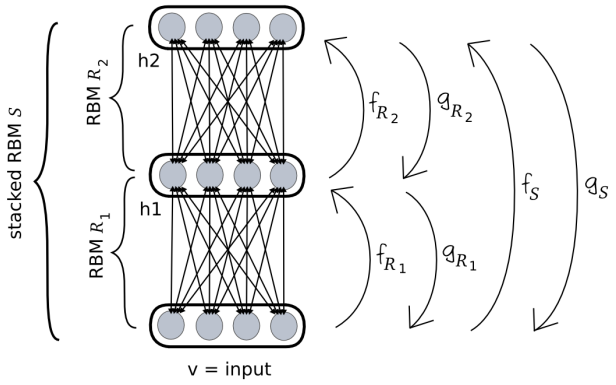

FIG. 2 – Topologie d'un réseau profond : RBM empilées.

Si l'on considère une RBM simple R, ses unités cachées sont entraînées pour refléter les corrélations des entrées, mais les unités cachées ne sont pas marginalement indépendantes. Elles le sont seulement conditionnellement à la configuration des entrées $\mathbf{v}$. Lorsqu'on utilise des RBM empilées R_i, l'avantage est d'apprendre les corrélations entre les unités cachées, sans faire l'hypothèse que toutes les configurations d'unités cachées sont équiprobables.

Pour modéliser une probabilité $P_D(\mathbf{v})$, on commence par introduire une RBM R_1 de probabilité jointe associée $P_{W_1}(\mathbf{v}, \mathbf{h^1})$. On modélise alors $P_D(\mathbf{v})$ avec la probabilité marginale $P(\mathbf{v}) = \sum_{\mathbf{h^1}} P_{W_1}(\mathbf{v}, \mathbf{h^1})$. Cette RBM définit également les probabilités conditionnelles $P_{W_1}(\mathbf{v}|\mathbf{h^1})$ et $P_{W_1}(\mathbf{h^1}|\mathbf{v})$. Les unités de $\mathbf{h^1}$ n'étant pas marginalement indépendantes, on peut modéliser la probabilité $P_{W_1}(\mathbf{h^1})$ avec une seconde RBM R_2 de probabilité associée $P_{W_2}(\mathbf{h^1}, \mathbf{h^2})$. $P(\mathbf{v})$ se décompose alors selon l'équation suivante :

$$P(\mathbf{v}) = \sum_{\mathbf{h^1}} \left[P_{W_1}(\mathbf{v}|\mathbf{h^1})P_{W_2}(\mathbf{h^1})\right]$$
$$= \sum_{\mathbf{h^1}} \left[P_{W_1}(\mathbf{v}|\mathbf{h_1}) \sum_{\mathbf{h^2}} P_{W_2}(\mathbf{h^1}, \mathbf{h^2})\right]$$

Plus généralement, en empilant une nouvelle RBM R_i de matrice de poids W_i, on décompose la probabilité d'un état interne $P_{W_i}(\mathbf{h^{i-1}})$ selon l'équation

$$P_{W_i}(\mathbf{h^{i-1}}) = \sum_{\mathbf{h^i}} \left[P_{W_i}(\mathbf{h^{i-1}}|\mathbf{h^i})P_{W_{i+1}}(\mathbf{h^i})\right] \quad (5)$$

où l'on étend la notation en posant $\mathbf{h_0} = \mathbf{v}$ pour la première couche, en entrée de la pile de RBM.
Soit une RBM simple R, nous définissons f_R qui correspond à la propagation d'une entrée vers la couche cachée, selon $P_W(\mathbf{h}|\mathbf{v})$ et g_R, qui correspond à la rétro-propagation d'une représentation cachée pour générer une entrée (voir figure 1). Dans la suite on note $S = R_0, \ldots, R_n$ une pile de RBM et on lui associe la fonction $f_S = f_{R_n} \circ \cdots \circ f_{R_1}$ qui correspond à la propagation d'une entrée jusqu'à la couche cachée la plus haute, et la fonction $g_S = g_{R_0} \circ \cdots \circ g_{R_n}$, qui correspond à la rétro-propagation permettant d'obtenir une entrée à partir de la couche cachée la plus haute. Ces notations sont indiquées sur la figure 2.

3 Optimisation de la topologie par critères non-supervisés

3.1 Validité de l'approche

Notre objectif est de découvrir un nombre de neurones optimal, couche après couche, la taille de la couche précédente étant fixée. En définitive, nous voulons évaluer, de manière non-supervisée, la performance de la couche en cours de construction. Ceci nécessite de pouvoir comparer différentes topologies avec un critère non-supervisé. Néanmoins une telle méthodologie, assez inhabituelle, peut conduire à plusieurs interrogations.
Premièrement, on peut se poser la question de l'existence d'un seul optimum global pour la taille d'une couche. Selon le critère utilisé, il est possible qu'il y ait plusieurs ou même une infinité d'optimums globaux. Le but usuel d'une procédure de sélection de modèle étant de déterminer le modèle le plus efficace en performance, mais aussi le plus parcimonieux en temps de calcul, nous chercherons donc à établir une borne inférieure sur le nombre de neurones nécessaire à chaque couche cachée.
Deuxièmement, concernant la méthode itérative, couche par couche, que nous proposons, il n'y a aucune garantie du fait que le problème d'optimisation des hyper-paramètres soit séparable[3] par rapport aux couches.

[3]Un problème d'optimisation est dit séparable s'il peut être optimisé indépendamment selon chacune de ses dimensions. Par extension, on dit que la sélection des hyper-paramètres est séparable par rapport aux couches si l'on peut optimiser les hyper-paramètres de chaque couche indépendamment.

Étant donné qu'il serait trop coûteux de faire une optimisation simultanée de tous les hyper-paramètres dans un réseau très profond, nous étudierons la séparabilité pour les deux premières couches cachées. Si le résultat d'une optimisation globale coïncide avec les hyper-paramètres obtenus par une optimisation couche par couche, cela suggérera que la séparabilité peut constituer une hypothèse raisonnable (voir partie 4.3).

3.2 Erreur de reconstruction

Les RBM empilées pouvant être considérées comme des modèles génératifs, il est naturel de s'intéresser à l'erreur de reconstruction comme critère non-supervisé pour la sélection de modèle. Pour effectuer une reconstruction, nous propageons d'abord l'entrée vers la couche la plus haute (voir figure 2) en utilisant les probabilités conditionnelles $P(\mathbf{h}|\mathbf{v})$ de chaque RBM. Dans un second temps, la configuration de la couche la plus haute est rétro-propagée avec les probabilités conditionnelles $P(\mathbf{v}|\mathbf{h})$. L'erreur de reconstruction correspond alors à la distance entre l'entrée initiale et l'entrée reconstruite.
Étant donné la méthode d'apprentissage (voir partie 2.1), mieux une RBM est entrainée, plus l'erreur de reconstruction moyenne sur un ensemble de test doit être faible.
Formellement, l'erreur de reconstruction est donnée par

$$Rec\_Error(\mathbf{v}) = d_E(g_S \circ f_S(\mathbf{v}), \mathbf{v}) \quad (6)$$

où $\mathbf{v}$ est un exemple aléatoire tiré dans l'ensemble des données, d_E est la distance euclidienne, et enfin f_S et g_S sont les fonctions associées respectivement à la propagation avant et arrière dans une RBM empilée, comme décrit en partie 2.2.

3.3 Énergie induite

Comme énoncé en partie 2.1, le modèle des RBM est basé sur une fonction d'énergie. Ainsi, à chaque couche d'une RBM empilée (figure 2) correspond une probabilité donnée par l'équation (2). Dans ce contexte, il est très intéressant de considérer la probabilité qu'une RBM associe à un exemple comme critère non-supervisé pour la sélection de modèle.
Une RBM bien entrainée doit en toute logique associer une probabilité élevée à des exemples d'une base de test[4]. Cependant, le facteur de normalisation Z rend le calcul de la probabilité exacte impossible en des temps raisonnables. Pour surmonter cette difficulté, nous nous intéressons directement à l'énergie, qui à la normalisation près, determine complètement la mesure de probabilité. Suivant l'équation (2) et pour Z constant, une énergie plus basse correspond à une probabilité plus haute (dépendance en exponentielle inverse).
Ainsi, pour une RBM simple R, nous considérons la quantité appellée "énergie induite" et qui ne dépend que de la configuration de ses unités visibles $\mathbf{v}$, les états des unités

[4]La base de test est un ensemble d'exemples disjoint de la base utilisée pour l'apprentissage.

cachées $\mathbf{h}$ ayant été induits par ceux des unités visibles, comme on le voit dans la relation :

$$Ind\_Energy(\mathbf{v}) = -\mathbf{h}^\top \mathbf{W}\mathbf{v} = -\underbrace{f_R(\mathbf{v})^\top}_{\mathbf{h}^\top}\mathbf{W}\mathbf{v} \quad (7)$$

Pour évaluer un modèle, cette énergie induite est moyennée sur un ensemble de test, de la même manière que pour l'erreur de reconstruction.

4 Expériences sur la topologie

4.1 Protocole expérimental

Pour l'étude expérimentale, la bibliothèque PLearn [20] utilisée par Larochelle et Bengio a été écartée en raison de sa complexité (plus de 500.000 lignes de code), qui rendait difficile les ajustements nécessaires à la réalisation de nos expériences. Nous avons donc programmé, en langage SCALA, une nouvelle implémentation des RBM et de l'algorithme de Mean Field Contrastive Divergence. Cette implémentation présente l'avantage d'être facilement adaptable et extensible[5].

Pour les expériences présentées ici, des RBMs empilées sont entrainées sur tout ou partie de l'ensemble de données MNIST, qui est composé de 60.000 images de taille 28×28, chacune représentant un chiffre entre 0 et 9, codé en niveaux de gris. Un ensemble de test disjoint composé de 1.000 exemples est utilisé pour évaluer les critères non supervisés, c'est-à-dire pour comparer les modèles sur la base de leur performance en généralisation.

4.2 Optimisation de la taille d'une couche

La figure 3 donne l'erreur de reconstruction en fonction du nombre de neurones dans la première couche cachée $\mathbf{h_1}$ d'une pile de RBM. La taille de la couche, en abscisse, est présentée selon une échelle logarithmique. On constate que la meilleure performance est obtenue pour les configurations ayant un minimum de 300 neurones dans cette couche cachée. De plus, une fois ce minimum atteint, ajouter davantage de neurones ne permet pas d'augmenter significativement la performance. Cette observation valide notre choix (cf. partie 3.1) de déterminer une borne inférieure pour la taille optimale d'une couche cachée.

La figure 4 donne l'énergie induite en fonction du nombre de neurones dans la première couche cachée $\mathbf{h_1}$ d'une pile de RBM.

On voit ici que le nombre optimum de neurones obtenu par le critère d'énergie induite est plus bas que celui donné par l'erreur de reconstruction. Ceci peut s'expliquer par le fait que nous avons négligé l'influence du facteur de normalization Z (cf. explications en partie 3.3). Plus précisément, les RBM avec davantage de neurones associent bien une probabilité plus forte à l'ensemble de test, mais ceci avec une énergie en moyenne plus haute et donc un numérateur

[5]Cette implémentation est accessible à l'adresse suivante : `http://www.ludovicarnold.com/machinelearning`

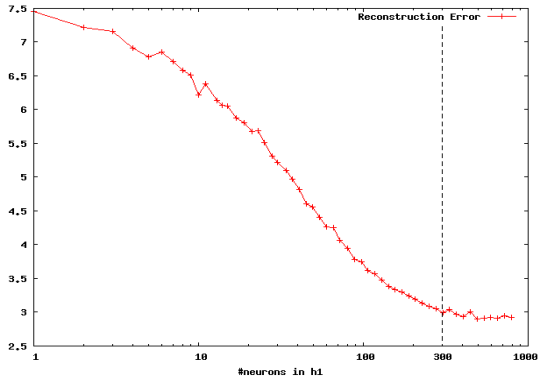

FIG. 3 – Erreur de reconstruction en fonction de la taille de la couche cachée $\mathbf{h_1}$ dans une RBM empilée à une couche cachée.

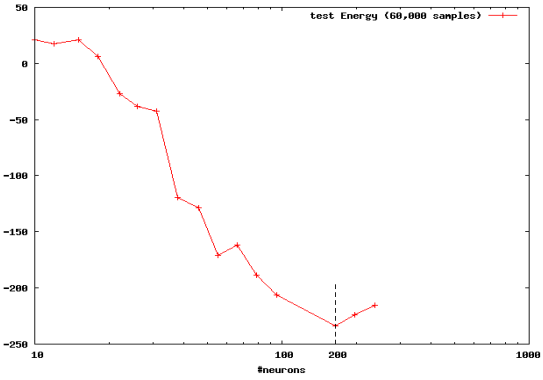

FIG. 4 – Énergie induite en fonction de la taille de la couche cachée $\mathbf{h_1}$ dans une RBM empilée à une couche cachée.

plus petit dans l'equation (2). La probabilité devant sommer à 1, ceci implique que le facteur de normalisation Z diminue quand le nombre de neurones augmente.

On peut également remarquer que, contrairement à l'erreur de reconstruction qui diminue très régulièrement, ce critère de l'énergie induite est sujet à une variation assez chaotique, pénalisant une procédure d'optimisation basée sur ce critère et nous conduisant à préférer le critère Rec_Error à Ind_Energy pour la sélection de modèle.

4.3 Validation de l'approche par couche

Pour valider les résultats précédents, nous optimisons la taille de la deuxième couche $\mathbf{h_2}$ après avoir fixé la taille de la couche $\mathbf{h_1}$ selon l'optimum précédent (300 neurones). La figure 5 montre le résultat de cette optimisation au moyen du critère de l'erreur de reconstruction et suggère que le nombre optimal de neurones pour la deuxième couche cachée $\mathbf{h_2}$ est borné inférieurement par 200 neurones. De la même manière que pour la première couche, ajouter plus de neurones ne permet pas d'améliorer significativement la performance de reconstruction.

Ensuite, nous comparons le résultat de l'optimisation couche par couche à une optimisation simultanée des deux

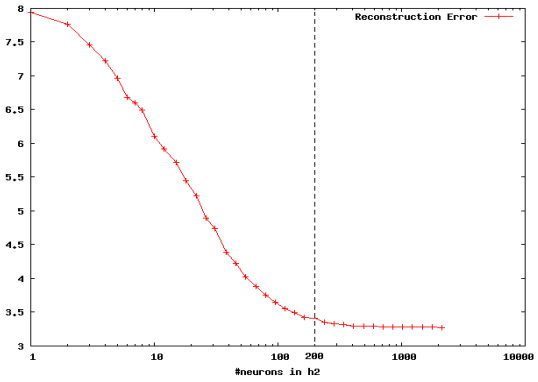

FIG. 5 – Erreur de reconstruction en fonction de la taille de la couche $\mathbf{h_2}$, dans une RBM empilée à deux couches cachées, la taille de $\mathbf{h_1}$ ayant été fixée à 300 neurones.

premières couche cachées $\mathbf{h_1}$ et $\mathbf{h_2}$ comme annoncé en partie 3.1. Les résultats sont donnés par la figure 6, pour le critère Rec_Error.

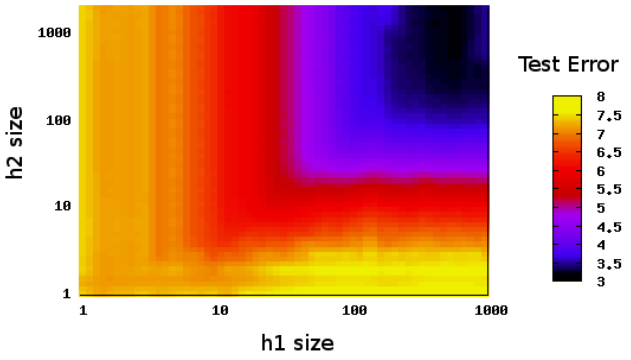

FIG. 6 – Erreur de reconstruction en fonction des tailles respectives des couches $\mathbf{h_1}$ et $\mathbf{h_2}$ dans une RBM empilée à deux couches cachées.

Les bornes inférieures de la topologie optimale, à savoir 300 neurones sur $\mathbf{h_1}$ et 200 sur $\mathbf{h_2}$, se retrouvent avec l'optimisation simultanée des deux couches cachées, comme on peut le constater sur la figure 6, en observant le carré sombre, en haut à droite (voir également partie 5).

4.4 Influence de la base d'exemples

Un phénomène intéressant apparaît lorsque l'on compare les nombres de neurones optimaux pour différentes tailles de l'ensemble d'entrainement. La figure 7 montre un recul du nombre de neurones minimum, pour l'erreur de reconstruction, à mesure que la taille de la base d'exemples augmente. On retrouve, sur la figure 8, le même phénomème pour l'énergie induite ; bien que l'optimum soit atteint pour des valeurs inférieures à celles obtenues avec l'erreur de reconstruction, on observe également que le nombre optimal de neurones diminue lorsque le nombre d'exemples augmente.Ce résultat surprenant sera discuté dans la partie 5.
Un deuxième aspect concerne la diversité des exemples de

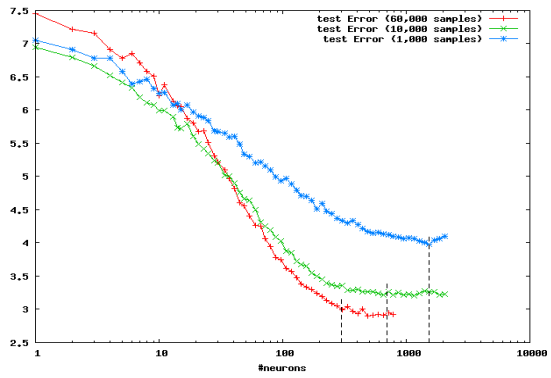

FIG. 7 – Erreur de reconstruction en fonction du nombre de neurones de la couche cachée dans une RBM empilée à une couche cachée, pour différentes tailles de la base d'exemples.

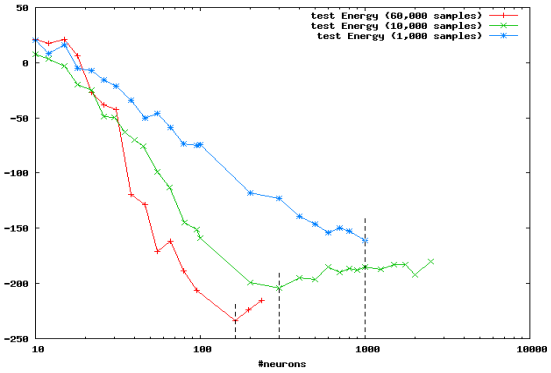

FIG. 8 – Energie Induite par l'entrée, en fonction du nombre de neurones de la couche cachée, dans une RBM empilée à une couche cachée et pour différentes tailles de la base d'exemples.

la base. Pour étudier cela, nous avons comparé (figure 9) l'évolution du critère Rec_Error lors de l'apprentissage de 60 000 exemples en tout, mais dans deux conditions différentes : une seule passe[6] d'une base de 60 000 exemples vs 60 passes d'une base de 1 000 exemples. On peut observer que le nombre optimal de neurones devient plus petit si l'on présente une plus grande variété d'exemples durant l'apprentissage. L'interprétation proposée est que la diversité des exemples permet une meilleure identification des caractéristiques à extraire et donc une meilleure selection des états cachés importants pour la représentation interne de l'entrée.

5 Discussion

L'étude que nous venons de présenter comporte quelques points qui méritent d'être confrontés à certains résultats de la littérature, comme nous allons en discuter.
Sur la figure 6, commentée en partie 4.3, on peut re-

[6]Une passe de la base d'exemples est une présentation unique de chaque exemple de la base.

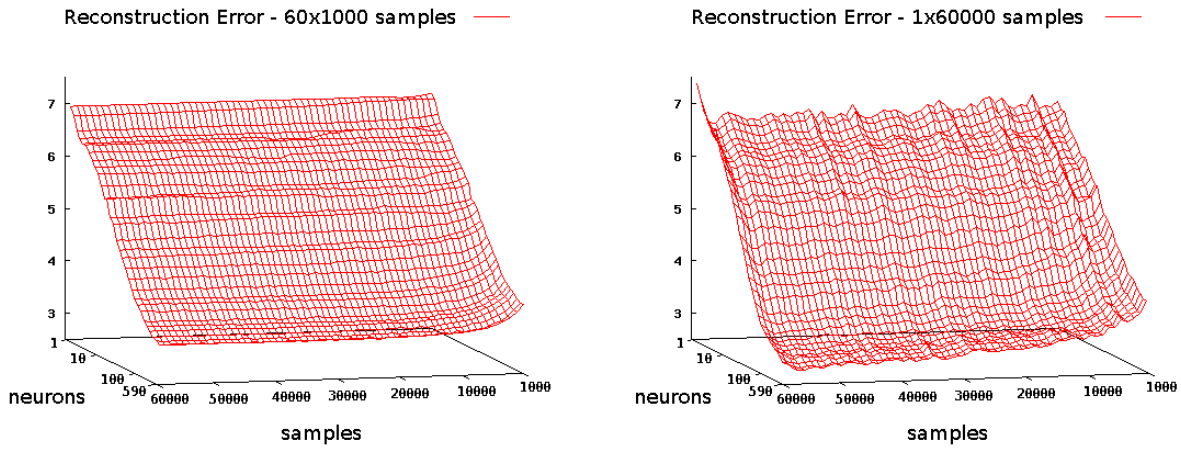

FIG. 9 – Erreur de reconstruction en fonction du nombre de neurones de la couche cachée dans une RBM empilée à une couche cachée, selon le nombre d'exemples présentés. À gauche, les mêmes 1.000 exemples sont présentés pendant l'apprentissage jusqu'à 60 passes. À droite, 60 000 exemples différents sont présentés par groupes de 1 000, en une seule passe.

marquer que le fait de mettre trop peu de neurones sur l'une ou l'autre des deux couches cachées amplifie l'erreur de reconstruction et il est impossible de faire rediminuer cette erreur en ajoutant des neurones sur l'autre couche. Notons que ce résultat met en évidence un comportement des RBM empilées, ou tout au moins du citère non-supervisé Rec_Error, très différent de celui des réseaux multicouches classiques. En effet, pour ceux-ci, il avait été démontré, de manière expérimentale et théorique, qu'une bonne performance en généralisation était obtenue à partir d'un nombre minimal de connexions dans l'ensemble du réseau, indépendamment de leur répartition sur l'une ou l'autre des couches cachées [6]. Dans la présente étude, les lignes de niveaux qui peuvent être définies par l'erreur de reconstruction ont plutôt une forme carrée alors que, dans l'étude citée, celles de la performance évaluée avaient une forme hyperbolique.

Un autre point concerne le résultat contre-intuitif présenté dans la partie 4.4. Il montre que, dans un réseau profond par RBM empilées, le nombre de neurones requis sur la première couche cachée diminue lorsque la taille de la base d'exemples augmente. Cependant, le phénomène contraire avait été observé dans des réseaux MLP, à la suite d'une approche *grid-search* exhaustive menée sur machine parallèle [17]. De plus, la théorie statistique de l'apprentissage [19] conduit à des conclusions contraires. En effet, la complexité en échantillon, ou le nombre d'exemples requis pour obtenir un niveau donné de performance en test, croît avec la VC-dimension[7] du modèle. Or, pour un réseau à une couche cachée de neurones sigmoïdes ayant N_w poids, les bornes inférieure et supérieure connues pour la VC-dimension sont de l'ordre de N_w^2 et N_w^4 respectivement. Les résultats de la théorie statistique de l'apprentissage tendent à prouver que le nombre de poids d'un "bon" réseau multicouche, et donc la taille de sa couche cachée, peut augmenter quand la taille de la base d'exemples augmente. Cependant, un des arguments avancés pour justifier l'intérêt des réseaux profonds [4, 10] est qu'un mode d'apprentissage couche par couche peut permettre d'extraire de la base d'exemples des caractéristiques de plus en plus abstraites. L'explication proposée pour le résultat contre-intuitif obtenu est qu'un plus grand nombre d'exemples appris donne la possibilité au modèle de mieux saisir les caractéristiques à extraire des données. L'expérience sur la diversité des exemples appris (figure 9 commentée en partie 4.4) apporte un argument supplémentaire à cette explication.

Enfin nous pouvons remarquer que la topologie optimale que nous obtenons ici (300, puis 200 neurones sur les deux premières couches cachées) est cohérente, au niveau de l'ordre de grandeur (quelques centaines de neurones), avec les topologies retenues expérimentalement pour la base de données MNIST (voir par exemple [10] et [15]). Notre méthode automatique d'optimisation de la topologie conduit à des modèles plus parcimonieux, puisque les études précédentes portaient sur des réseaux profonds ayant au moins 500 neurones sur chacune des deux premières couches cachées. Toutefois il convient de noter que d'une part les critères d'optimisation considérés ne sont pas supervisés et que d'autre part, nous avons utilisé une version champ moyen dans laquelle chaque neurone est potentiellement capable de transmettre plus d'information [21].

6 Conclusion et perspectives

En réponse au problème de la sélection d'hyper-paramètres dans les réseaux de neurones profonds, nous avons introduit une approche couche par couche dont la validité est confirmée empiriquement. Nous avons étudié deux critères non-supervisés pour optimiser la topologie d'un réseau de neurones profond et plus particulièrement le nombre de

[7] VC-dimension ou dimension de Vapnik-Chervonenkis ; c'est, une mesure de la capacité d'un système d'apprentissage, par exemple un réseau de neurones (cf [19]).

neurones dans les couches cachées. Nous concluons à la supériorité de l'erreur de reconstruction sur l'énergie induite pour la sélection de modèle. Sur la base MNIST, nous obtenons, pour la classification de chiffres manuscrits, des topologies optimales de réseaux comparables à celles utilisées jusqu'à présent dans la littérature sur les réseaux profonds. Nos expérimentations montrent également que dans les réseaux profonds, il est possible de considérer le nombre de neurones dans chaque couche de manière indépendante, contrairement aux réseaux MLP classiques dans lesquels le nombre total de neurones est le critère principal. D'autre part, nous avons observé que le nombre de neurones requis tend à diminuer avec le nombre d'exemples et avec leur diversité. L'explication proposée pour ce résultat pourra être approfondie dans de futurs travaux, portant par exemple sur une visualisation des caractéristiques apprises. Comme extension à ce travail, nous envisageons aussi d'appliquer la méthode gloutonne d'optimisation couche par couche à d'autres hyper-paramètres des réseaux profonds tels l'optimisation du nombre de couches cachées.

Références

[1] D.H. Ackley, G.E. Hinton, and T.J. Sejnowski. A learning algorithm for boltzmann machines. *Cognitive Science*, 9(1) :147–169, 1985.

[2] Y. Bengio. Learning deep architechtures for ai. Technical report, Université de Montréal, Dept. IRO, 2007.

[3] Y. Bengio, P. Lamblin, V. Popovici, and H. Larochelle. Greedy layer-wise training of deep networks. In B. Schölkopf, J. Platt, and T. Hoffman, editors, *Advances in Neural Information Processing Systems 19*, pages 153–160. MIT Press, Cambridge, MA, 2007.

[4] Y. Bengio and Y. LeCun. Scaling learning algorithms towards ai. In *Large-Scale Kernel Machines*. MIT Press, 2007.

[5] M. Cosnard, P. Koiran, and H. Paugam-Moisy. A step towards the frontier between one-hidden-layer and two-hidden layer neural networks. In *Proc. of Int. Joint Conf. Neural Networks, IJCNN'93*, volume 3, pages 2292–2295, 1993.

[6] J. Droniou, A. Elisseeff, H. Paugam-Moisy, and O. Teytaud. Contrôle de l'architecture et des représentations internes dans les réseaux de neurones multicouches. In *Actes de la Conférence sur l'Apprentissage, CAP'99*, pages 185–194, 1999.

[7] K. Funahashi. On the approximate realization of continuous mappings by neural networks. *Neural Networks*, 2(3) :183–192, 1989.

[8] G.E. Hinton. Training products of experts by minimizing contrastive divergence. *Neural Computation*, 14 :1771–1800, 2002.

[9] G.E. Hinton, S. Osindero, and Yee-Whye Teh. A fast learning algorithm for deep belief nets. *Neural Conputation*, 18 :1527–1554, 2006.

[10] G.E. Hinton and R. Salakhutdinov. Reducing the dimensionality of data with neural networks. *Science*, 313(5786) :504–507, July 2006.

[11] K. Hornik, M. Stinchcombe, and H. White. Universal approximation of an unknown mapping and its derivatives using multilayer feedforward networks. *Neural Networks*, 3(5) :551–560, 1990.

[12] C. Kenyon and H. Paugam-Moisy. Multilayer neural networks and polyhedral dichotomies. *Annals of Mathematics and Artificial Intelligence*, 24 :115–128, 1998.

[13] S. Kirkpatrick, C.D. Gelatt Jr., and M.P. Vecchi. Optimization by simulated annealing. *Science*, 220 :671–680, 1983.

[14] H. Larochelle and Y. Bengio. Classification using discriminative restricted boltzmann machines. In *ICML '08 : Proceedings of the 25th international conference on Machine learning*, pages 536–543, New York, NY, USA, 2008. ACM.

[15] H. Larochelle, Y. Bengio, J. Louradour, and P. Lamblin. Exploring strategies for training deep neural networks. *The Journal of Machine Learning Research*, 10 :1–40, 2009.

[16] H. Larochelle, D. Erhan, A. Courville, J. Bergstra, and Y. Bengio. An empirical evaluation of deep architectures on problems with many factors of variation. In *ICML '07 : Proceedings of the 24th international conference on Machine learning*, pages 473–480, New York, NY, USA, 2007. ACM.

[17] H. Paugam-Moisy. Parallel neural computing based on network duplicating. In I. Pitas, editor, *Parallel Algorithms for Digital Image Processing, Computer Vision and Neural Networks*, pages 305–340. John Wiley, 1993.

[18] P.E. Utgoff and D.J. Stracuzzi. Many-layered learning. *Neural Computation*, 14(10) :2497–2529, 2002.

[19] V.N. Vapnik. *The nature of statistical learning theory*. Springer, 1995.

[20] P. Vincent, Y. Bengio, J. Keable, R. Ducharme, M. Monperrus, and O. Delalleau. Plearn user's guide - how to use the plearn machine-learning library and tools. Website, 2005. `http://plearn.berlios.de/users_guide/index.html`.

[21] M. Welling and G.E. Hinton. A new learning algorithm for mean field boltzmann machines. In *Proceedings of the International Conference on Artificial Neural Networks (ICANN)*, 2002.