Modèle hybride champs Markovien conditionnel et réseau de neurones profond

Trinh Minh Tri Do, Thierry Artières

Modèle hybride champs Markovien conditionnel et réseau de neurones profond

Trinh-Minh-Tri Do* et Thierry Artières**

** IDIAP, Martigny, Suisse*
*** LIP6, UPMC, Paris, France*
tri.do@idiap.ch, thierry.artieres@lip6.fr

RÉSUMÉ. *Nous proposons un modèle qui est un hybride de champs Markovien conditionnel et d'un réseau de neurones profond. Il s'agit d'un modèle graphique non linéaire exploitable pour toute tâche de prédiction de sorties structurées. L'intérêt est de combiner la capacité des réseaux de neurones profonds à extraire des caractéristiques de haut niveau et la capacité discriminante des champs conditionnels pour des données complexes.*

ABSTRACT. *We propose a non-linear graphical model for structured prediction. It combines the power of deep networks to extract high level features with the discriminant power of Markov networks for complex data, yielding a powerful and scalable model that we apply to signal labeling tasks.*

MOTS-CLÉS : *Champs aléatoires conditionnels, Réseaux de neurones profonds, Reconnaissance de caractères.*

KEYWORDS: *Conditional Random Fields, Deep Neural Networks, Character Recognition.*

1. Introduction

Cette étude porte sur la proposition de modèles pour ce que l'on appelle aujourd'hui la prédiction de sorties structurées dans le domaine de l'apprentissage automatique. Il s'agit d'un problème important aujourd'hui dans des domaines aussi divers que la bioinformatique, l'étiquetage syntaxique, l'extraction d'information, la reconnaissance de la parole ou de l'écriture. Nous nous intéressons ici à des tâches d'étiquetage de séquences telles que celles attaquées en reconnaissance de la parole ou de l'écrit.

Depuis des dizaines d'années les modèles Markoviens cachés sont la technologie de référence pour le traitement de données séquentielles, que ce soit la classification ou la segmentation. Ils sont cependant limités car d'une part ils reposent sur des hypothèses fortes sur les données et d'autre part ils sont classiquement appris par maximisation de la vraisemblance, c'est à dire à l'aide d'un critère non discriminant. Ce dernier point vient du fait que ces modèles sont des modèles génératifs de la loi jointe sur la séquence d'observations X et sur la séquence de variables cachées correspondante Y.

Bien entendu les modèles discriminants sont généralement plus performants que les modèles non discriminants puisqu'ils sont appris par optimisation d'un critère plus directement lié à la probabilité d'erreur. De nombreux travaux ont donc porté sur les moyens d'accroître la capacité discriminante des modèles Markoviens. Une première possibilité est d'utiliser un critère d'apprentissage discriminant pour les MMC et d'optimiser leurs paramètres par rapport à ce critère. Divers critères ont ainsi été proposés parmi lesquels le Minimum d'Erreur de Classification (ou MCE) (Juang *et al.,* 1992), le critère du Perceptron (Collins, 2002), le Maximum d'Information Mutuelle (ou MMI) (Woodland *et al.,* January 2002) et plus récemment des critères de maximisation de la marge (Sha *et al.,* 2007, Do *et al.,* 2009a).

Une approche plus directe consiste à utiliser un modèle discriminant qui modélise directement la loi de probabilité a posteriori $P(Y|X)$ plutôt que la loi jointe comme dans les modèles génératifs (Mccallum *et al.,* 2000, Lafferty, 2001). Les champs Markoviens conditionnels (ou Conditional Random Fields ou CRF) sont un exemple typique de ce type de modèles. Les réseaux de Markov à Maximum de Marge (ou Maximum Margin Markov networks ou M3N) de (Taskar *et al.,* 2004) sont un autre type de modèle de ce type qui vont "'plus loin"' en focalisant sur l'apprentissage d'une fonction discriminante ils sont une extension naturelle des machines à vecteurs de support aux données structurées. Bien qu'appris avec des critères très différents (vraisemblance conditionnelle et maximum de marge) les M3N et les CRFs sont très similaires et sont basés sur un même modèle graphique. On peut voir les CRFs comme une instance de M3Ns.

Les CRFs "'linéaires"', exploitant des fonctions potentielles log-linéaires ont été intensivement utilisées en traitement du texte ou de séquences biologiques (Altun *et al.,* 2003, Sato *et al.,* 2005). Pourtant ces CRFs peuvent obtenir des performances assez modestes par rapport à des modèles exploitant des transformations non linéaires

via des noyaux (Taskar *et al.,* 2004) sur des données plus "'signal"' telles que les images ou la parole. Bien qu'il soit théoriquement possible d'utiliser des noyaux dans des CRFs (Lafferty *et al.,* 2004) cela s'avère peu exploitable en pratique, et les méthodes kernelisées posent de toutes façons souvent des problèmes de passage à l'échelle. Nous envisageons ici une autre approche pour dépasser les limitations des modèles de type CRF linéaires sur des données manuscrites.

Ces dernières années on assiste à une renouveau des réseaux de neurones profonds (à grand nombre de couches cachées) qui s'avèrent capables d'extraire automatiquement dans leurs couches cachées successives des caractéristiques pertinentes et de plus en plus haut niveau (Hinton *et al.,* 2006, Bengio *et al.,* 2007). Des modèles profonds ont été utilisés avec succès sur des images ((Hinton *et al.,* 2006)) des données de capture de mouvement (Taylor *et al.,* 2007), du texte etc. Ils se sont montrés efficaces pour l'extraction de caractéristiques exploitées par des modèles linéaires.

Nous proposons dans ce travail un modèle hybride alliant la capacité discriminante des modèles conditionnels Markoviens (CRFs) et la capacité des réseaux profonds à extraire des caractéristiques non linéaires de haut niveau. L'hybridation consiste à entrainer le réseau profond à produire directement les valeurs des fonctions potentielles d'un CRF linéaire. Nous proposons en fait une architecture de ce type que nous optimisons globalement.

Nous introduisons tout d'abord le formalisme des réseaux de Markov et des CRFs pour la prédiction de sorties structurées en Section 2. Puis nous présentons les NeuroCRF en Section 3 et Section 4 et décrivons des résultats expérimentaux obtenus sur une tâche de reconnaissance de mots écrits off-line pré segmentés.

2. Prédiction Structurée avec des CRFs

Nous commençons avec une brève introduction sur les réseaux de Markov et sur la prédiction structurée avec les CRFs. Les réseaux Markoviens constituent une importante famille de modèles graphiques permettant de modéliser une loi de probabilité sur des variables structurées Y composées d'un ensemble de variables Y_i. Ils sont définis par un graphe non dirigé $G = (V, E)$. Chaque composante Y_i de Y est associée à un noeud $v_i \in V$. L'hypothèse de Markov consiste à considérer que pour tout $U \subset Y \setminus \{Y_i, Y_j\}$, Y_i est indépendant de Y_j conditionnellement à U si et seulement si tout chemin de v_i à v_j passe par au moins un noeud de U. Aussi, s'il n'existe pas de chemin entre v_i et v_j alors Y_i et Y_j sont indépendants.

Un modèle graphique de cette famille peut être paramétrisé en utilisant le théorème d'Hammersley-Clifford (Hammersley *et al.,* 1971) qui stipule que toute distribution sur Y dont les dépendances conditionnelles sont codées par le graphe G peut être factorisée sur les cliques[1] dans G suivant $P(Y) \propto \prod_{c \in C} \psi_c(Y_c)$ où C est l'ensemble des cliques dans G, Y_c représente l'ensemble des noeuds (i.e. variables) dans la clique c, et

1. Une clique est un ensemble de noeuds $c \subset V$ qui forme un sous graphe totalement connecté

$\psi_c(Y_c)$ sont des fonctions potentielles (positives). La prédiction de sorties structurées vise à construire un modèle qui prédit avec précision une sortie structurée $y \in Yset$ pour toute entrée $x \in Xset$.

Considérons maintenant la sortie Y que l'on cherche à prédire comme un ensemble de variables aléatoires inter-dépendantes dont les composantes sont liées par les dépendances conditionnelles encodées par le graphe G de cliques $c \in C$. Notons X la variable aléatoire correspondant à l'entrée (i.e. l'observation). Alors, pour x donné, l'inférence consiste à trouver la sortie qui maximise la probabilité conditionnelle [2] $p(y|x)$. En exploitant (Hammersley *et al.,* 1971) un CRF définit une loi de probabilité conditionnelle qu'on obtient par normalisation :

$$p(y|x) = \prod_{c \in C} \psi_c(x, y_c)/Z(x) \quad [1]$$

où $Z(x) = \sum_{y \in Yset} \prod_{c \in C} \psi_c(x, y_c)$ est un facteur de normalisation global. Un choix habituel pour les fonctions potentielles est d'utiliser l'exponentielle d'une énergie, comme dans des machines de Boltzmann :

$$\psi_c(x, y_c) = e^{-E_c(x, y_c, w)} \quad [2]$$

pour faciliter l'apprentissage qui se résume alors à un problème d'optimisation convexe l'usage est d'utiliser des fonctions d'énergie linéaires dans le vecteur de paramètres w_c et un vecteur de caractéristiques joint $\Phi_c(x, y_c)$ suivant $E_c(x, y_c, w) = -\langle w_c, \Phi_c(x, y_c)\rangle$, ces choix conduisent à un modèle log-linéaire.

3. Modèle hybride réseau profond et champ Markovien conditionnel

3.1. *Principe*

Pour dépasser les limitations intrinsèques des fonctions d'énergie linéaires pour modéliser des entrées complexes, nous proposons d'apprendre un réseau de neurones (NN) profond à extraire des caractéristiques pertinentes et à produire des scores correspondant aux fonctions d'énergie E_c (Cf. Eq. (2)) exploitées par le champ Markovien conditionnel (CMC).

Ainsi le NN prends une observation en entrée et produit un ensemble de *sorties énergie*[3] paramétrisées par w. Ce NN a un nombre de cellules de sortie égal au nombre de cliques multiplié par le nombre de réalisations pour Y_c. *Les sorties énergie* sont calculées pour chaque clique c et pour chaque réalisation Y_c. Par exemple si une clique c concerne deux variables aléatoires $Y_c \in L \times L$, alors il y a $|L|^2$ cellules de sortie du NN dédiées à la clique c.

---

2. Nous utilisons la notation $p(y|x) = p(Y = y|X = x)$
3. Nous utilisons ici le terme *sortie énergie* pour marquer la différence entre les sorties du NN, les sorties énergie, et les sorties du modèle y

La loi de probabilité conditionnelle $P(Y|X)$ est donc complètement définie par les paramètres du réseau w. La figure 1 montre un exemple de NeuroCRF avec une structure en arbre. Il y a 3 cliques de taille 2 et 4 cliques de taille 1 (où nous nous intéressons aux cliques sur les noeuds de Y uniquement puisque nous nous intéressons à une distribution conditionnelle des Y's conditionnellement à X). Le nombre de sorties énergie (i.e. de cellules de sortie dans le réseau de neurones) est alors $(3|L|^2 + 4|L|)$. Comme on le voit sur la figure, les sorties du réseau de neurones (i.e. sorties énergie) sont regroupées par clique (i.e. il y a un groupe par clique).

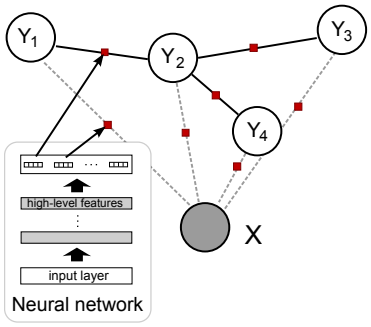

Figure 1. *Exemple de NeuroCRF de structure en arbre*

L'inférence dans un NeuroCRF consiste à trouver l'étiquetage $\widehat{y}$ qui correspond le mieux à l'observation x, i.e. la sortie de plus basse énergie :

$$\widehat{y} = \underset{y}{\operatorname{argmax}}\, p(y|x, w) = \underset{y}{\operatorname{argmin}} \sum_{c \in C} E_c(x, y_c, w) \qquad [3]$$

Pour cela, on met en entrée du réseau l'observation x et l'on propage de couche en couche jusqu'à calculer les sorties énergie $E_c(x, y_c, w)$. Puis on utilise une passe de programmation dynamique pour trouver $\widehat{y}$ d'énergie minimale.

3.1.1. *Architecture du réseau neuronal.*

Nous avons utilisé des réseaux de neurones à propagation avant pour construire nos NeuroCRFs, en reprenant des idées de (Hinton *et al.,* 2006) où des réseaux de neurones profonds de ce type se sont avérés capables d'extraire des caractéristiques de plus en plus abstraites dans les couches cachées successives.

Diverses architectures peuvent être utilisées. On peut utiliser un réseau de neurones différent pour chaque clique, au risque de surapprendre, ou bien partager les couches cachées du réseau de neurones (Figure 2-gauche) pour calculer l'ensemble des sorties énergie (Figure 2-droite). Afin d'aviter le surapprentissage et de limiter le nombre de paramètres à apprendte nous n'avons utilisé dans nos expérimentations que l'architecture la plus légère avec partage de couches cachées.

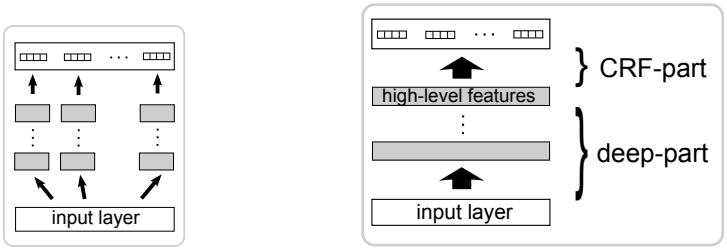

Figure 2. *Architecture du réseau de neurones profond sans poids partagés (gauche) ou avec poids partagés (droite).*

On obtient un point de vue intéressant si on considère un réseau de neurones avec des sorties linéaires (avec une fonction d'activation linéaire pour les unités de sortie). Dans ce cas un NeuroCRF peut être vu comme un CRF linéaire exploitant les caractéristiques de haut niveau calculées sur la dernière couche cachée du réseau profond. Dans la suite nous appellerons la partie "'haute"' (la dernière couche) d'un NeuroCRF la *partie CRF* et le reste la *partie profonde* (voir Figure 2-right).

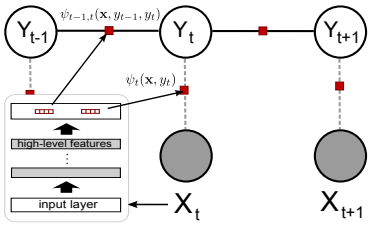

Figure 3. *Exemple de NeuroCRF de structure en chaîne*

3.2. *NeuroCRFs de structure chaîne pour l'étiquetage de séquences*

Bien que le formalisme des NeuroCRFs que nous proposons soit assez général, nous avons jusqu'ici mené des expérimentations sur des NeuroCRFs de structure chaîne basés sur une hypothèse de Markov du premier ordre (Figure 3). Cette instance nous permet d'explorer le potentiel des NeuroCRFs sur le problème de l'étiquetage de séquences. Dans ces modèles il y a deux types de cliques :

– *cliques locales* (x, y_t) à chaque position t, dont les fonctions potentielles sont notées $\psi_t(x, y_t)$, et les fonctions d'énergie correspondantes sont notées E_{loc}.

– *cliques de transitions* (x, y_{t-1}, y_t) entre deux positions successives à $t-1$ et t, dont les fonctions potentielles sont notées $\psi_{t-1,t}(x, y_{t-1}, y_t)$, et les fonctions d'énergie correspondantes sont notées E_{trans}.

Dans de tels modèles il est commun de considérer des fonctions d'énergie partagées entre les cliques similaires à différents instants (Lafferty, 2001)[4]. Pour prendre en compte cette modélisation les fonctions d'énergie prennent un paramètre additionnel, le temps t.

$$\psi_t(x, y_t) = e^{-E_{loc}(x,t,y_t,w)} \text{ et } \psi_{t-1,t}(x, y_{t-1}, y_t) = e^{-E_{trans}(x,t,y_{t-1},y_t,w)} \quad [4]$$

Au final la loi de probabilité conditionnelle sur les sorties y connaissant l'entrée x est définie par $p(y|x, w) = \exp(-[\sum_{t\geq 1} E_{loc}(x, t, y_t, w) + \sum_{t>1} E_{trans}(x, t, y_{t-1}, y_t, w)])/Z(x)$ où $Z(x)$ est un facteur de normalisation. Ainsi on peut utiliser un NN avec seulement $|L|+|L|^2$ sorties pour calculer toutes les sorties énergie. Ces sorties correspondent aux énergies locales (au nombre de $|L|$ qui est le nombre d'états), et aux énergies de transition au nombre de $|L|^2$.

4. Estimation des paramètres

On utilise un ensemble d'apprentissage de n paires d'entrée-sortie $(x^1, y^1), ..., (x^n, y^n) \in Xset \times Yset$. Et on cherche un jeu de paramètres w tel que $\forall i : y^i = \text{argmax}_{y\in Yset}\, p(y|x^i, w)$. L'apprentissage peut être mis sous la forme d'un problème d'optimisation du type :

$$\min_w \lambda\Omega(w) + R(w) \quad [5]$$

où $R(w) = \frac{1}{n}\sum_i R_i(w)$ est un terme d'adéquation aux données d'apprentissage (e.g. risque empirique), et $\Omega(w)$ est un terme de régularisation (nous avons utilisé la norme L_2), où λ est un facteur permettant de régler le compromis entre une bonne modélisation des données d'apprentissage et une bonne généralisation. Nous avons utilisé deux critères pour apprendre les NeuroCRFs, que nous détaillons ici.

4.1. *Critères*

On peut utiliser de multiples critères discriminants pour les CRFs (et plus généralement pour les modèles log-linéaires), ils peuvent être utilisés de la même façon pour apprendre les NeuroCRFs.

---

4. Ces auteurs considèrent deux ensembles de paramètres, un pour les cliques locales l'autre pour les cliques de transition

4.1.1. *Vraisemblance Conditionnelle.*

Le critère de vraisemblance conditionnelle (Conditional Maximum Likelihood) est celui proposé dans l'article originel (Lafferty, 2001), pour l'estimation des paramètres des CRFs **w** :

$$\begin{aligned} R_i^{CML}(w) &= -\log p(y^i|x^i,w) \\ &= \sum_c E_c(x^i,y_c^i,w) - \sum_{y\in Yset} \exp[\sum_c E_c(x^i,y_c,w)] \end{aligned} \quad [6]$$

4.1.2. *Large marge.*

Les méthodes de grande marge visent plus directement à donner des scores discriminants importants aux sorties correctes. Dans les NeuroCRFs, la fonction discriminante est une somme des fonctions d'énergie sur les cliques (voir Eq. (3)) : $F(x,y,w) = -\sum_{c\in C} E_c(x,y_c,w)$. Le critère de large marge pour des données structurées consiste à trouver w tel que (Taskar *et al.,* 2004) :

$$F(x^i,y^i,w) \geq F(x^i,y,w) + \Delta(y^i,y)\ \forall y \in Yset \quad [7]$$

où $\Delta(y^i,y)$ est un terme de pénalité permettant de prendre en compte les différences entre étiquetages (e.g. par exemple la distance de Hamming entre y et y^i). Nous supposons ici une pénalité décomposable (comme la distance de Hamming) telle que $\Delta(y^i,y) = \sum_c \delta(y_c^i,y_c)$ de façon à ce qu'elle puisse être factorisée sur le graphe et donc intégrable dans la procédure de programmation dynamique nécessaire pour calculer $\mathrm{argmax}_{y\in Yset}\, p(y|x)$. La fonction de coût élémentaire dans les NeuroCRFs est donc (en notant $\Delta E_c(x^i,y_c,y_c^i,w) = -E_c(x^i,y_c,w) + E_c(x^i,y_c^i,w)$) :

$$\begin{aligned} R_i^{LM}(w) &= \max_{y\in Yset} F(x^i,y,w) - F(x^i,y^i,w) + \Delta(y^i,y) \\ &= \max_{y\in Yset} \sum_c \Delta E_c(x^i,y_c,y_c^i,w) + \delta(y_c^i,y_c) \end{aligned} \quad [8]$$

4.2. *Apprentissage*

Du fait de la non convexité du critère d'optimisation des NeuroCRFs, une bonne initialisation est cruciale. Heureusement diverses procédures d'apprentissage non supervisé ont été proposées récemment avec succès (Hinton *et al.,* 2006). Nous détaillons l'initialisation du réseau de neurones puis nous discutons de la réestimation globale du NeuroCRF.

4.2.1. *Initialisation*

L'initialisation des différentes couches cachées d'un NeuroCRF est réalisée incrémentalement, couche par couche, comme cela a été popularisé ces dernières années pour l'apprentissage d'architectures profondes. Dans notre implémentation la partie profonde du NeuroCRF est initialisée de façon non supervisée à l'aide de machines de Boltzmann restreintes (RBMs) comme cela a été proposé notamment dans (Hinton *et al.,* 2006). Une fois qu'une cascade de RBMs a été apprise, une par une, on transforme ces RBMs cascadées en un réseau de neurones à propagation avant, il s'agit

de la partie profonde du NeuroCRF. Plus de détails sur cette procédure peuvent être trouvés dans (Hinton *et al.,* 2006).

Une fois cette partie profonde initialisée, on utilise le réseau de neurones pour calculer des représentations de haut niveau des données d'entrée (i.e. le vecteur des activations de la dernière couche cachée). La partie CRF est alors initialisée en apprenant, en mode supervisé un CRF linéaire classique sur ces représentations de haut niveau (avec un algorithme standard pour cela, LBFGS). Comme nous l'avons déjà dit, un CRF linéaire de ce type correspond en fait à une couche de sortie du réseau de neurones qui est mise en cascade en sortie de la partie profonde.

L'ensemble des poids de la partie profonde et de la partie CRF constitue une solution initiale pour w_0, qui est ensuite raffinée globalement en mode supervisé.

4.2.2. *Réapprentissage fin*

Le réapprentissage fin (ou fine tuning) consiste à réestimer les paramètres du NeuroCRF de façon globale, en partant d'une solution initiale raisonnable. Aucun des critères considérés plus haut n'est convexe ce qui rend l'optimisation éventuellement difficile.

Aucun des critères d'apprentissage que nous avons évoqués ne conduit à un problème d'optimisation convexe (Subsection 4.1) puisque nous utilisons des réseaux de neurones non linéaires (les fonctions d'activation des cellules cachées sont des sigmoïdes). Néanmoins, n'importe quelle méthode à base de gradient peut être utilisée afin de trouver un optimum local à condition que l'on puisse exprimer et calculer le gradient du coût par rapport aux poids du réseau de neurones. Nous avons utilisé une méthode propre basée sur la technique des plans sécants (Do *et al.,* 2009a). Nous montrons comment obtenir ce gradient maintenant.

En fait, tant que $R_i(w)$ est continu et que l'on peut calculer $\frac{\partial R_i(w)}{\partial E_c(x,y_c,w)}$ (ce qui est le cas pour les critères évoqués plus haut) on peut calculer le (sous)-gradient de $R(w)$ par rapport à w à l'aide d'une étape de rétropropagation. En notant E_i l'ensemble des énergies pour une entrée x^i, et en utilisant la règle de composition des dérivées pour chaque $\frac{\partial R_i(w)}{\partial w}$:

$$\frac{\partial R(w)}{\partial w} = \frac{1}{n}\sum_i \frac{\partial R_i(w)}{\partial w} = \frac{1}{n}\sum_i \frac{\partial R_i(w)}{\partial E_i}\frac{\partial E_i}{\partial w} \qquad [9]$$

où $\partial E_i \overline{\partial w}$ est la matrice Jacobienne des sorties du réseau de neurones (pour l'entrée x^i) par rapport aux poids w. En posant $\frac{\partial R_i(w)}{\partial E_i}$ comme les erreurs à la sortie du réseau de neurones et en les rétropropageant on peut obtenir $\frac{\partial R_i(w)}{\partial w}$.

5. Résultats expérimentaux

Nous avons réalisé des expériences sur des caractères manuscrits sur les données de (Kassel, 1995) (pour d'autres résultats en reconnaissance de la parole voir (Do *et*

al., 2009b)) avec des NeuroCRF à structure de chaîne et en comparant à des méthodes de référence sur ces jeux de données.

Le jeu de données OCR consiste en 6876 mots contenant environ à 50 000 caractères (Kassel, 1995, Taskar *et al.,* 2004). Ces données sont des séquences de caractères isolés, chacun représenté par un vecteur binaire de dimension 128, appartenant à 26 classes (caractères minuscules). Le jeu de données est divisé en 10 parts. Nous fournissons des résultats obtenus en validation croisée dans deux situations. Une première situation correspond à une grande base d'apprentissage, nous utilisons 9 parts en apprentissage et une en test. Une seconde situation correspond à une petite base d'apprentissage, nous utilisons 1 part en apprentissage et 9 en test.

Nous avons appris des NeuroCRFs à deux couches cachées, les sortie énergie correspondant aux transitions n'ont de connexion qu'avec une unité de biais ce qui revient à ne pas considérer l'entrée et donc à utiliser des scores assimilables à des probabilités de transition. Lors de l'initialisation de la partie profonde du NeuroCRF est initialisée par 50 itérations sur la base d'apprentissage en utilisant l'algorithme de Contrastive Divergence proposé par Hinton.

La figure 4 montre l'influence de l'architecture de la partie profonde des NeuroCRFs sur leurs performances. Nous y montrons des résultats obtenus avec une petite base d'apprentissage pour des modèles à une ou deux couches cachées et pour différentes tailles de ces couches. Comme on peut le voir, augmenter la taille de ces couches cachées permet d'accroître la performance de modèles à une ou deux couches cachées. Quelle que soit l'architecture (une ou deux couches cachées) la performance semble plafonner à partir d'une certaine taille de la couche cachée, et cette performance plancher est meilleure et est atteinte plus vite dans le cas de modèles à deux couches cachées. Enfin quelle que soit l'expérience, les modèles à deux couches cachées sont plus performants que les modèles à une couche cachée. Ces résultats suggèrent que la stratégie de multiplier les couches cachées et leurs dimensions peut encore permettre d'obtenir de meilleures performances. Nous ne fournissons que des résultats pour des architectures limitées car les apprentissage sont assez longs.

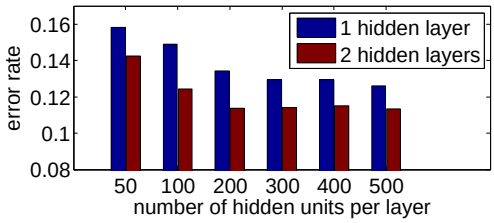

Figure 4. *Influence de l'architecture du NN sur les performances (small training set).*

Nous comparons ensuite les NeuroCRFs avec les méthodes à l'état de l'art sur cette base de données et pour la prédiction de sorties structurées : les M3N, les CRF linéaires (opérant sur les données brutes) et deux variantes de NeuroCRFs apprises

avec les deux critères d'apprentissage évoqués (maximum de vraisemblance conditionnelle et maximum de marge). Les NeuroCRFs ont ici deux couches cachées de 200 cellules chacune. La Table 1 rassemble les résultats obtenus par validation croisée pour les deux cas de petite et grande base de données d'apprentissage. Pour les NeuroCRFs la performance obtenue par le système après initialisation(avant réapprentissage global) est indiquée entre parenthèses. Ces résultats montrent que les NeuroCRFs surpassent significativement les autres méthodes, y compris les M3N avec noyau non linéaire (dont la performance sur les grandes bases de données d'apprentissage ne sont pas fournis pour des problèmes de passage à l'échelle). On peut voir également que si l'initialisation permet d'atteindre des performances déjà intéressantes, le réapprentissage global permet lui d'accroître nettement la performance. Enfin on voit que les deux critères d'apprentissage semblent assez proches avec un léger avantage à la vraisemblance conditionnelle.

Tableau 1. *Taux d'erreurs des NeuroCRF et de méthodes à l'état de l'art pour la prédiction de sorties structurées sur le jeux de données OCR dans les deux contextes : petite base d'apprentissage ou grande base d'apprentissage. La performance des NeuroCRFs après initialisation et avant réapprentissage global est indiquée entre parenthèses. Les résultats des SVM cubiques et des M3N cubiques viennent de (Taskar* et al., *2004).*

	petite base d'apprentissage	grande base d'apprentissage
CRF linéaire	0.2162	0.1420
M3N linéaire	0.2113	0.1346
SVM cubique	0.192	*non disponible*
M3N cubique	0.127	*non disponible*
NeuroCRF^{CML}	**0.1080** (*0.1224*)	**0.0444** (*0.0697*)
NeuroCRF^{LM}	**0.1102** (*0.1221*)	**0.0456** (*0.0736*)

6. Conclusion

Nous avons présenté un modèle combinant CRFs et réseaux de neurones profonds qui permet de combiner les avantages des deux approches, la capacité des réseaux profonds à extraire automatiquement des caractéristiques non linéaires de haut niveau, et la capacité discriminante des CRFs. Nous avons détaillé la procédure d'apprentissage et produit des résultats expérimentaux pour la reconnaissance de séquences de caractères. Ces résultats montrent clairement l'avantage de cette architecture et l'intérêt des réseaux profonds pour l'extraction de caractéristiques pertinentes.

7. Bibliographie

Altun Y., Johnson M., Hofmann T., « Investigating Loss Functions and Optimization Methods for Discriminative Learning of Label Sequences », 2003.

Bengio Y., Lamblin P., Popovici D., Larochelle H., Montréal U. D., Québec M., « Greedy layer-wise training of deep networks », *NIPS*, MIT Press, 2007.

Collins M., « Discriminative training methods for hidden Markov models : theory and experiments with perceptron algorithms », *EMNLP*, p. 1-8, 2002.

Do T.-M.-T., Artières T., « Large Margin Training for Hidden Markov Models with Partially Observed States », *ICML*, Omnipress, p. 265-272, 2009a.

Do T.-M.-T., Artières T., « Neural conditional random fields », *Workshop on Deep Learning for Speech Recognition and Related Applications, NIPS 2009*, 2009b.

Hammersley J. M., Clifford P., « Markov field on finite graphs and lattices », 1971, Unpublished manuscript.

Hinton G. E., Osindero S., Teh Y.-W., « A Fast Learning Algorithm for Deep Belief Nets », *Neural Computation*, vol. 18, n° 7, p. 1527-1554, July, 2006.

Juang B., Katagiri S., « Discriminative learning for minimum error classification », *IEEE Trans. Signal Processing, Vol.40, No.12*, 1992.

Kassel R. H., A comparison of approaches to on-line handwritten character recognition, PhD thesis, Cambridge, MA, USA, 1995.

Lafferty J., « Conditional random fields : Probabilistic models for segmenting and labeling sequence data », *ICML*, Morgan Kaufmann, p. 282-289, 2001.

Lafferty J., Zhu X., Liu Y., « Kernel conditional random fields : representation and clique selection », *ICML*, 2004.

Mccallum A., Freitag D., Pereira F., « Maximum Entropy Markov Models for Information Extraction and Segmentation », *ICML*, p. 591-598, 2000.

Sato K., Sakakibara Y., « RNA secondary structural alignment with conditional random fields », *ECCB/JBI*, p. 242, 2005.

Sha F., Saul L. K., « Large Margin Hidden Markov Models for Automatic Speech Recognition », *in* , B. Schölkopf, , J. Platt, , T. Hoffman (eds), *NIPS 19*, MIT Press, Cambridge, MA, p. 1249-1256, 2007.

Taskar B., Guestrin C., Koller D., « Max-Margin Markov Networks », *in* , S. Thrun, , L. Saul, , B. Schölkopf (eds), *NIPS 16*, 2004.

Taylor G. W., Hinton G. E., Roweis S. T., « Modeling Human Motion Using Binary Latent Variables », *NIPS 19*, 2007.

Woodland P., Povey D., « Large scale discriminative training of hidden Markov models for speech recognition », *Computer Speech and Language*, vol. 16, p. 25-47(23), January 2002.

SERVICE ÉDITORIAL – HERMES-LAVOISIER
14 rue de Provigny, F-94236 Cachan cedex
Tél : 01-47-40-67-67
E-mail : revues@lavoisier.fr
Serveur web : http://www.revuesonline.com

ANNEXE POUR LE SERVICE FABRICATION
A FOURNIR PAR LES AUTEURS AVEC UN EXEMPLAIRE PAPIER
DE LEUR ARTICLE ET LE COPYRIGHT SIGNE PAR COURRIER
LE FICHIER PDF CORRESPONDANT SERA ENVOYE PAR E-MAIL

1. ARTICLE POUR LES ACTES :

   Trinh-Minh-Tri Do et al.

2. AUTEURS :

   Trinh-Minh-Tri Do et Thierry Artières***

3. TITRE DE L'ARTICLE :

   Modèle hybride champs Markovien conditionnel et réseau de neurones profond

4. TITRE ABRÉGÉ POUR LE HAUT DE PAGE MOINS DE 40 SIGNES :

   Modèle Hybride CRF et RN profond

5. DATE DE CETTE VERSION :

   17 février 2010

6. COORDONNÉES DES AUTEURS :

   – adresse postale :
   * IDIAP, Martigny, Suisse
   ** LIP6, UPMC, Paris, France
   tri.do@idiap.ch, thierry.artieres@lip6.fr
   – téléphone : 00 00 00 00 00
   – télécopie : 00 00 00 00 00
   – e-mail : Roger.Rousseau@unice.fr

7. LOGICIEL UTILISÉ POUR LA PRÉPARATION DE CET ARTICLE :

   LaTeX, avec le fichier de style `article-hermes.cls`,
   version 1.2 du 03/03/2005.

8. FORMULAIRE DE COPYRIGHT :

   Retourner le formulaire de copyright signé par les auteurs, téléchargé sur :
   `http://www.revuesonline.com`

SERVICE ÉDITORIAL – HERMES-LAVOISIER
14 rue de Provigny, F-94236 Cachan cedex
Tél : 01-47-40-67-67
E-mail : revues@lavoisier.fr
Serveur web : http://www.revuesonline.com