Sélection de caractéristiques pour les champs aléatoires conditionnels par pénalisation L_1

Nataliya Sokolovska* **— Olivier Cappé*** **— François Yvon****

(*) Telecom ParisTech et LTCI/CNRS
{sokolovska,cappe}@telecom-paristech.fr
(**) Université Paris Sud 11 et LIMSI/CNRS
yvon@limsi.fr

RÉSUMÉ. *Les modèles probabilistes discriminants permettent de manipuler des représentations linguistiques riches, sous la forme de vecteurs de caractéristiques de très grande taille. Travailler en grande dimension pose des problèmes, en particulier computationnels, qui sont exacerbés dans le cadre de modèles de séquences tels que les champs aléatoires conditionnels (CRF). Sélectionner automatiquement les caractéristiques pertinentes s'avère alors intéressant et donne lieu à des modèles plus compacts et plus faciles à utiliser. Dans cette étude, nous proposons un algorithme d'estimation pour les CRF qui réalise une telle sélection, par le truchement d'une pénalisation L_1. Nous présentons également les résultats d'expériences menées sur des tâches de traitement des langues (le* chunking *et la détection des entités nommées). Nous proposons enfin des pistes pour améliorer l'efficacité computationelle de cette technique.*

ABSTRACT. *Discriminative probabilistic models are able to cope with enriched linguistic representations, typically in the form of extremely large feature vectors. Working in high dimensional spaces is however problematic, and these problems are compounded in the case of structured output models, such as conditional random fields (CRF). In this context, feature selection techniques help building more compact and efficient models. In this work, we propose a novel estimation algorithm for CRF with L_1 penalization, which yields sparse representations, thus implicitly selecting relevant features. We also report experiments conducted on two standard language engineering tasks (chunking and named Entity recognition), for which we analyze the generalization performance and the patterns of selected features. We finally suggest various implementation speed-ups that should allow to efficiently tackle even larger feature vectors.*

MOTS-CLÉS : *modèles de séquences, choix de caractéristiques, champs aléatoires conditionnels.*

KEYWORDS: *conditional random fields, probabilistic sequence models, feature selection.*

1. Introduction

1.1. *Apprentissage discriminant et structuré pour le TAL*

Les méthodes à base d'apprentissage automatique ont profondément bouleversé la méthodologie de développement d'applications de TAL. Les approches fondées sur l'accumulation de règles symboliques produites par des experts ont progressivement été supplantées par des méthodes numériques qui s'appuient principalement sur l'analyse statistique de corpus annotés. Le cadre d'utilisation le plus commun est celui de l'apprentissage supervisé de règles de classification, qui permettent d'assigner une ou des étiquettes catégorielles à la représentation d'une entité linguistique. Une analyse rétrospective des avancées dans ce domaine réalisées durant la dernière décennie permet de dégager deux axes d'innovations majeurs : d'une part la diffusion des modèles d'apprentissage discriminants (ou conditionnels), qui permettent d'intégrer des traits linguistiques riches et variés, d'autre part le développement de techniques à même de traiter les dépendances statistiques qui existent entre les diverses sous-parties de représentations linguistiques structurées telles que les séquences, les arbres ou les graphes acycliques.

Sur le premier axe, citons en particulier l'introduction pour le TAL de modèles de régression logistique multinomiale (aussi appelés modèles d'entropie maximale) (Rathnaparkhi, 1998), des classificateurs à vaste marge (SVM) (Cortes et Vapnik, 1995) ou encore du *boosting* (Freund et Schapire, 1996). Du point de vue statistique, ces modèles présentent l'intérêt de résoudre directement, plutôt qu'indirectement par la règle de Bayes, le problème de classification visé, en modélisant (dans le cas des modèles d'entropie maximale) la distribution conditionnelle de la classe sachant l'observation $p_\theta(y|x)$ sous la forme d'une distribution exponentielle selon :

$$p_\theta(y|x) = \frac{\exp(\theta^T F(x,y))}{Z_\theta(x)} \quad [1]$$

Dans cette équation, $F(x,y)$ est un vecteur de caractéristiques arbitraires de l'entrée x et de la classe y, chacune des composantes correspondant à un test atomique réalisé conjointement sur x et y ; θ est le vecteur de paramètres correspondant, contenant une composante par caractéristique ; $Z_\theta(x)$ est un terme de normalisation qui garantit que cette formulation définit une distribution de probabilité : $Z_\theta(x) = \sum_y p_\theta(y|x)$. D'un point de vue computationnel, l'estimation de ces modèles discriminants conduit à résoudre des problèmes d'optimisation convexe (ce qui assure l'unicité de la solution) : la maximisation de la marge pour les SVM ou la maximisation de la log-vraisemblance conditionnelle pour les modèles d'entropie maximale. Dans ce dernier cas, l'estimation des paramètres donne lieu au programme suivant (les sommes portent sur les instances d'apprentissage, indicées de $n = 1$ à N) :

$$\theta^* = \underset{\theta}{\operatorname{argmax}} \sum_{n=1}^{N} \log(p_\theta(y^{(n)}|x^{(n)})) = \underset{\theta}{\operatorname{argmin}} \sum_{n=1}^{N} \log(Z_\theta(x^{(n)})) - \theta^T F(x^{(n)}, y^{(n)}) \quad [2]$$

Il existe, pour ces problèmes, des techniques d'optimisation bien rodées, qui permettent de trouver efficacement les valeurs optimales des paramètres y compris dans des espaces de très grande dimension. L'efficacité de ces méthodes est conditionnée par l'ajout d'un terme de *régularisation* (ou de *pénalisation*) à la fonction objectif, qui permet d'obtenir une stabilité numérique de la solution même en très grande dimension. Ce terme de régularisation prend le plus souvent la forme d'une fonction linéaire du carré de la norme L_2 du vecteur de paramètres, ce qui préserve la différentiabilité et la convexité de la fonction objectif et se prête à une interprétation bayésienne (Chen et Rosenfeld, 2000) en termes de distribution *a priori* sur les paramètres. Concrètement, cela revient à ajouter un terme $\rho\|\theta\|_2^2$ à la fonction objectif du programme défini en [2]. Du point de vue de l'analyse linguistique enfin, ces modèles ont l'avantage de pouvoir intégrer des traits linguistiques riches et variés, permettant l'incorporation au sein du modèle de multiples sources de connaissances. En témoigne l'étude exemplaire de (Toutanova et Manning, 2000), qui, à partir d'une analyse serrée des erreurs commises par un étiqueteur morphosyntaxique, spécifie un ensemble de caractéristiques linguistiquement fondées qui conduisent à une amélioration très sensible des performances en généralisation. Cette flexibilité de modélisation a conduit au développement de modèles de complexité croissante, comportant un très grand nombre de paramètres : ainsi, l'étude précitée propose de prendre en compte non seulement des traits lexicaux (l'identité du mot à étiqueter, ou encore celle de ces voisins proches), mais également des traits typographiques (le mot est-il en majuscule ? capitalisé ? comporte-t-il des chiffres ? des tirets ? etc.), des traits visant à modéliser grossièrement la morphologie (présence de préfixes ou de suffixes de longueur bornée dans le mot), ou encore à mieux caractériser le contexte syntaxique (le mot est-il précédé d'un modal ? d'un auxiliaire ? etc.).

Les développements menés sur le second axe visent à prendre en compte le caractère *structuré* de nombreuses représentations linguistiques, en intégrant de manière plus explicite les dépendances qui existent entre les divers sous-constituants de ces représentations (on se reportera à (Bakir *et al.*, 2007) pour un état de l'art actuel de ces techniques). Par exemple, le modèle des champs aléatoires conditionnels (CRF), introduit dans (Lafferty *et al.*, 2001), permet d'étendre les modèles d'entropie maximale à des séquences d'étiquettes. Si la forme générale du modèle reste celle donnée dans l'équation [1], les observations x et les sorties y correspondent, dans ce nouveau modèle, à des séquences complètes et les caractéristiques peuvent simultanément intégrer des tests portant sur plusieurs étiquettes au sein de la séquence à prédire. Même si, pour des raisons computationnelles, ces tests portent sur des étiquettes voisines, cette extension permet d'obtenir de nouveaux gains très significatifs en généralisation : ainsi les auteurs de (Toutanova *et al.*, 2003), toujours sur une tâche d'étiquetage morphosyntaxique, rapportent que la modélisation explicite de ces dépendances conduit à une réduction de près de 2 points du taux d'erreur. Les modèles probabilistes discriminants pour les données structurées ont été généralisés pour traiter des tâches de réordonnancement (*ranking*) (Collins et Duffy, 2002 ; Charniak et Johnson, 2005), d'étiquetage d'arbres (Jousse *et al.*, 2006), d'analyse syntaxique en constituants (Rozenknop, 2002 ; Finkel *et al.*, 2008), d'analyse en dépendances (Koo

et al., 2007) ou encore d'alignement de mots en traduction automatique (Blunsom et Cohn, 2006). La prise en compte de dépendances dans des représentations structurées a toutefois pour conséquence directe l'augmentation massive du nombre de paramètres impliqués dans la modélisation, puisque, pour s'en tenir au simple cas des séquences, la prise en compte des dépendances entre étiquettes adjacentes requiert un nombre de paramètres qui croît comme le carré du nombre d'étiquettes possibles.

1.2. *Sélection de caractéristiques*

Au final, l'effet de ce double mouvement a été l'utilisation de modèles de complexité toujours croissante, intégrant typiquement des centaines de milliers, voire des millions de paramètres. Si cette augmentation de la complexité s'accompagne le plus souvent d'une amélioration des performances, elle n'en pose pas moins problème. La première difficulté est computationnelle : ces millions de caractéristiques doivent être évaluées pour chaque exemple d'apprentissage et de test ; les paramètres correspondants doivent être stockés en mémoire et conduisent à des problèmes d'optimisation en très grande dimension. Estimer ces modèles conduit finalement à se placer dans une situation où le nombre de paramètres surpasse de plusieurs ordres de grandeur le nombre d'instances d'apprentissage au risque d'instabilité numérique des solutions obtenues, même en présence de régularisation (voir, par exemple, les difficultés décrites dans (Sha et Pereira, 2003), dont les auteurs construisent un modèle intégrant près de 4 millions de caractéristiques pour une tâche d'analyse syntaxique de surface). La deuxième difficulté est d'ordre statistique : la présence de caractéristiques inutiles ou redondantes dans le modèle peut conduire à des phénomènes de surapprentissage et amener une dégradation des performances en généralisation (Kazama et Tsujii, 2003). En fait, de nombreux auteurs s'étonnent d'observer des dégradations des performances lorsque certaines caractéristiques sont injectées dans le modèle (voir *infra*). La troisième difficulté porte sur la modélisation linguistique. Il n'existe en pratique pas de limite aux traits que l'on voudrait pouvoir inclure dans un modèle : faute de critères (autres que les performances globales) pour décider de l'utilité de tel ou tel trait, la pratique la plus répandue consiste à ajouter tous les traits possibles et imaginables (dans la limite du raisonnable) et à évaluer empiriquement l'intérêt de telle ou telle combinaison de caractéristiques. Cette situation n'est pas satisfaisante et suscite des interrogations. Ainsi, dans (Toutanova et Manning, 2000), déjà mentionné, les auteurs constatent qu'ajouter des caractéristiques qui testent à la fois les mots suivants et précédents conduit à une petite dégradation des performances en comparaison à n'utiliser que des tests sur les mots suivants. De même, l'utilisation de caractéristiques portant sur les préfixes semble avoir un effet négatif :

> *Conversely, empirically it was found that the prefix features for rare words were having a net negative effect on accuracy. We do not at present have a good explanation for this phenomenon.*

Ces constatations plaident pour le développement de techniques de sélection automatique des caractéristiques les plus utiles. Les méthodes proposées dans la littérature

pour ce faire sont toutefois très heuristiques. Une pratique commune consiste à ne conserver que les caractéristiques qui sont suffisamment fréquentes dans les données d'apprentissage. Ainsi, Toutanova *et al.* (2000) imposent un seuil de fréquence minimal pour considérer des caractéristiques, heuristique qui est reprise sans autre forme de discussion dans de nombreux travaux sur les modèles exponentiels : les auteurs de (Lafferty *et al.*, 2001) se limitent ainsi à l'examen de quelques préfixes, ceux de (Bender *et al.*, 2003) utilisent la même stratégie pour sélectionner les caractéristiques incluses dans leur détecteur d'entités nommées, etc. Une approche plus fondée est proposée dans (McCallum et Li, 2003 ; McCallum, 2003) qui s'inspire de (Della Pietra *et al.*, 1997) pour développer un algorithme glouton de sélection des caractéristiques sur la base d'une approximation de leur contribution à la log-vraisemblance globale.

La question de la sélection automatique de variables explicatives a pourtant donné lieu à une vaste littérature dans le domaine des statistiques, et au développement de méthodes efficaces (Guyon et Elisseeff, 2003). Parmi celles-ci, une approche initialement introduite dans un cadre de régression linéaire (Tibshirani, 1996) consiste à employer une pénalisation de la norme L_1 du vecteur de paramètres. Concrètement, cela revient à ajouter dans la fonction objectif un terme de la forme $\rho\|\theta\|_1$ en place de $\rho\|\theta\|_2^2$. Ce changement a pour effet d'annuler tous les paramètres dont la contribution à la log-vraisemblance est insuffisante pour contrebalancer le « coût » de leur inclusion dans le modèle, alors qu'avec une pénalisation L_2, ces paramètres prennent des valeurs arbitrairement faibles, mais non nulles. Seules les caractéristiques associées à des paramètres non nuls sont alors sélectionnées. Le comportement particulier de l'optimisation avec cette forme de régularisation est, en dernière analyse, dû à la non-différentiabilité du terme de régularisation en tout point où l'une des coordonnées de θ est nulle (voir également sur ce point la discussion de (Hastie *et al.*, 2001, p. 68 et suivantes)) . Cette propriété, malheureusement, interdit l'utilisation de techniques usuelles d'optimisation, qui présupposent l'existence du gradient de la fonction objectif. Pour les modèles d'entropie maximale, des techniques alternatives d'optimisation des paramètres qui restent valides dans ce cas ont été récemment développées (Kazama et Tsujii, 2003 ; Dudík *et al.*, 2004 ; Riezler et Vasserman, 2004 ; Friedman *et al.*, 2007) : nous étendons ici la dernière de ces propositions au cas des CRF.

1.3. *Contributions*

Le travail présenté dans cet article propose donc une nouvelle réponse au problème de la sélection de caractéristiques pour les champs conditionnels aléatoires, qui repose sur la mise en œuvre d'une stratégie particulière d'optimisation de la fonction objectif en présence d'une pénalisation L_1. En nous appuyant principalement sur les travaux de (Friedman *et al.*, 2008), notre approche consiste à optimiser de manière cyclique successivement sur chacune des coordonnées de la fonction objectif. Pour accélérer cette procédure, nous en développons une variante, qui considère simultanément des blocs de coordonnées. Sur la base d'une série d'expérimentations, conduites sur deux tâches standard d'étiquetage de séquences, nous analysons les résultats que ces tech-

niques permettent d'obtenir, à la fois en termes de performances, mais également en étudiant en détail les effets de cette pénalisation sur les caractéristiques qui sont sélectionnées. Notre contribution est donc ici double et porte à la fois sur les aspects computationnels de la méthode et sur l'analyse des résultats qu'elle permet d'obtenir. Des résultats complémentaires, portant notamment sur les aspects algorithmiques et présentant des comparaisons avec d'autres algorithmes d'optimisation des CRF avec pénalité L_1, sont présentés dans (Sokolovska *et al.*, 2009).

Le reste de cet article est organisé comme suit : dans la section 2, nous introduisons nos notations en présentant le modèle des champs aléatoires conditionnels, en nous limitant au cas de dépendances linéaires d'ordre 1. La section 3 constitue l'essentiel de notre contribution méthodologique en détaillant les différentes stratégies d'optimisation qui sont proposées en présence d'une régularisation L_1. Les données, tâches, et résultats expérimentaux qui illustrent les avantages de cette méthode sont discutés à la section 4. La leçon principale est que pour les problèmes considérés, l'utilisation d'une régularisation L_1 permet d'obtenir des performances équivalentes à celles obtenues avec pénalisation L_2 tout en utilisant un nombre bien plus restreint de caractéristiques. Dans la section 5, cette méthode est mise en relation avec d'autres approches de l'état de l'art ; nous proposons également diverses pistes pour améliorer l'efficacité de la méthode et concluons sur un rapide bilan du travail réalisé.

2. Champs aléatoires conditionnels pour l'étiquetage de séquences

2.1. *Champs aléatoires conditionnels*

Les champs aléatoires conditionnels (Lafferty *et al.*, 2001 ; Sutton et McCallum, 2006) correspondent à un modèle discriminant de prédiction supervisée de séquences appartenant à la famille logistique généralisée ou d'entropie maximale. Supposons donnée une séquence d'entrée $\mathbf{x} = (x_1, \ldots, x_T)$ ainsi qu'une séquence d'étiquettes à prédire $\mathbf{y} = (y_1, \ldots, y_T)$. Le modèle dit d'ordre un ou *linear chain* instancie l'équation [1] en postulant une distribution de probabilité conditionnelle de la séquence d'étiquettes donnée par :

$$p_\theta(\mathbf{y}|\mathbf{x}) = \frac{1}{Z_\theta(\mathbf{x})} \exp\left\{ \sum_{t=1}^{T} \sum_{k=1}^{K} \theta_k f_k(y_{t-1}, y_t, x_t) \right\} \quad [3]$$

Dans l'équation ci-dessus, $Z_\theta(\mathbf{x})$ désigne le facteur de normalisation défini par :

$$Z_\theta(\mathbf{x}) = \sum_{\mathbf{y} \in Y^T} \exp\left\{ \sum_{t=1}^{T} \sum_{k=1}^{K} \theta_k f_k(y_{t-1}, y_t, x_t) \right\} \quad [4]$$

où Y désigne l'ensemble des valeurs prises par y_t (de même X désignera l'ensemble des valeurs prises par x_t). Par convention, l'étiquette y_0 correspond à une valeur conventionnelle toujours observée indiquant le début de séquence et on note $\overline{Y}$ l'ensemble Y complété par cette étiquette de début de séquence. Dans l'équation [3],

$\theta = (\theta_1, \ldots, \theta_K)$ désigne le vecteur de paramètres du modèle tandis que les fonctions f_k correspondent aux caractéristiques sur lesquelles la prédiction des étiquettes va reposer. Par rapport à la structure très générale de l'équation [1], la contrainte principale qui apparaît dans [3], et justifie le terme de modèle d'ordre un, est due au fait que chaque caractéristique ne fait intervenir, au plus, que des bigrammes d'étiquettes successives (y_{t-1}, y_t). Ce choix implique que la loi conditionnelle $p_\theta(\mathbf{y}|\mathbf{x})$ possède une structure d'indépendance conditionnelle dans laquelle, sachant $\mathbf{x}$, y_{t-1} et y_{t+1}, y_t est indépendant de y_s pour $s < t-1$ ou $s > t+1$ et sa loi ne dépend que de x_t, y_{t-1} et y_{t+1}. Nous verrons ci-dessous dans la section 2.4 que la raison de ce choix est essentiellement de nature computationnelle. En ce qui concerne la dépendance des caractéristiques vis-à-vis de la séquence d'entrée $\mathbf{x}$, les possibilités sont en fait beaucoup plus larges et la forme retenue pour l'équation [3] ne constitue qu'un exemple, choisi pour sa simplicité, où chaque caractéristique est une fonction du triplet (y_{t-1}, y_t, x_t). En pratique, et selon les applications envisagées, il est fréquent que les caractéristiques ne portent pas uniquement sur la valeur x_t de la séquence d'entrée à la position t, mais plutôt sur le contenu de la séquence autour de la position t, par exemple sur le trigramme (x_{t-1}, x_t, x_{t+1}). Dans les exemples de la section 4, nous rencontrerons un autre cas de figure fréquent où la séquence d'entrée est en fait multimodale, $x_t = (x_t^1, \ldots, x_t^D)$ et où l'on utilise une superposition de caractéristiques portant sur chacune des modalités de la séquence d'entrée en remplaçant $\sum_{k=1}^K \theta_k f_k(y_{t-1}, y_t, x_t)$ dans [3] par $\sum_{k=1}^K \sum_{d=1}^D \theta_k^d f_k^d(y_{t-1}, y_t, x_t^d)$. Nous verrons cependant ci-dessous qu'en ce qui concerne l'estimation des paramètres θ du modèle, la dépendance des caractéristiques vis-à-vis de la séquence d'entrée ne pose pas de problème particulier dans la mesure où l'on suppose toujours cette séquence observée. Pour des raisons de simplicité d'écriture, nous conservons donc la forme présentée dans l'équation [3] qui permet d'illustrer l'ensemble des enjeux liés à l'utilisation des champs aléatoires conditionnels d'ordre un.

2.2. *Choix des caractéristiques*

Dans le cadre du traitement automatique des langues, où les séquences tant d'entrée que de sortie sont assimilables à des variables catégorielles, le choix le plus naturel pour les caractéristiques f_k consiste à utiliser des fonctions booléennes qui valent 1 ou 0 selon que le triplet (y_{t-1}, y_t, x_t) est ou n'est pas dans une configuration particulière. Plus précisément, nous considérerons deux types de caractéristiques, dites respectivement de type *unigramme* et *bigramme*, ce qui permet de décomposer le terme $\sum_{k=1}^K \theta_k f_k(y_{t-1}, y_t, x_t)$ selon :

$$\sum_{k=1}^{K} \theta_k f_k(y_{t-1}, y_t, x_t) = \sum_{y \in Y, x \in X} \mu_{y,x} \mathbb{1}(y_t = y, x_t = x) + \sum_{(y',y) \in \overline{Y} \times Y, x \in X} \lambda_{y',y,x} \mathbb{1}(y_{t-1} = y', y_t = y, x_t = x) \quad [5]$$

où $\mathbb{1}(\text{test}) = 1$ si le test est positif et vaut 0 sinon. Dans l'expression ci-dessus, le vecteur de paramètres $\mu = (\mu_{y,x})_{y \in Y, x \in X}$ correspond aux caractéristiques de type unigramme qui testent la cooccurrence d'une étiquette particulière y et d'une entrée x à la même position. Le vecteur de paramètres $\lambda = (\lambda_{y',y,x})_{(y',y) \in \overline{Y} \times Y, x \in X}$ correspond aux caractéristiques de type bigramme qui testent la succession de deux étiquettes conjointement avec l'occurrence d'une valeur particulière de l'entrée. Il est clair également que dans [5] la sommation sur les caractéristiques est dorénavant relativement fictive puisque l'on pourrait réécrire [5] sous la forme $\mu_{y_t,x_t} + \lambda_{y_{t-1},y_t,x_t}$.

Quelques commentaires s'imposent. Tout d'abord il n'est pas évident à ce stade que l'utilisation simultanée de caractéristiques des deux types soit nécessaire dans la mesure où pour toute valeur de μ et λ, il existe une valeur λ' des paramètres pour laquelle la partie bigramme seule réalise de façon équivalente [5] (en prenant $\lambda'_{y',y,x} = \lambda_{y',y,x} + \mu_{y,x}$). Nous verrons cependant que l'utilisation simultanée des deux types de caractéristiques est nécessaire pour obtenir des jeux de caractéristiques réduits, conduisant à de bonnes performances de classification. Notons également que l'utilisation des caractéristiques unigrammes seules correspondrait à un modèle plus simple de régression logistique position par position dans lequel :

$$p_\mu(\mathbf{y}|\mathbf{x}) = \prod_{t=1}^{T} \frac{\exp(\mu_{y_t,x_t})}{\sum_{y \in Y} \exp(\mu_{y,x_t})} \qquad [6]$$

L'aspect séquentiel du modèle, qui tient compte des corrélations entre étiquettes successives est donc uniquement le fait des caractéristiques de type bigramme. Par ailleurs, les caractéristiques unigrammes et bigrammes sont en nombres très différents puisque le total des caractéristiques unigrammes disponibles est de $|Y||X|$ (où $|Y|, |X|$ désignent le cardinal de Y et X) tandis qu'il y a $|Y|(|Y|+1)|X|$ caractéristiques de type bigramme, le terme $(|Y|+1)$ venant de l'utilisation de l'étiquette supplémentaire qui correspond au début de séquence. Les caractéristiques de type bigramme sont donc très majoritaires numériquement dès que le nombre d'étiquettes distinctes est important. On peut se demander si ces effectifs théoriques de caractéristiques ne peuvent pas être fortement élagués en pratique au vu des fréquences d'occurrences de ces caractéristiques dans les jeux de données disponibles pour l'entraînement des modèles. Quelle que soit la méthode d'inférence utilisée, il est aisé de vérifier que les caractéristiques unigramme ou bigramme portant sur une éventuelle modalité d'entrée x qui n'est jamais observée dans le corpus d'apprentissage peuvent être éliminées sans aucun risque, puisque les paramètres correspondant $\mu_{y,x}$ et $\lambda_{y',y,x}$ seront de toute façon estimés à zéro. Toute autre forme d'élagage *a priori*, fondé sur les statistiques d'occurrence dans le corpus d'apprentissage, modifie en revanche le résultat d'estimation. En particulier, le fait de forcer $\mu_{y,x}$ à zéro même si l'occurrence $(y_t = y, x_t = x)$ n'est jamais apparue dans le corpus d'apprentissage peut avoir des conséquences importantes si le motif $(y_t = y', x_t = x)$ est fréquent pour d'autres valeurs y' de l'étiquette. Nous verrons dans la section 4 qu'un des intérêts de l'approche discutée dans cet article est précisément de sélectionner des caractéristiques pertinentes de façon beaucoup plus efficace qu'en se contentant d'examiner, *a priori*, les fréquences d'occurrences des motifs.

Dans la suite, nous utiliserons à la fois la représentation en termes des caractéristiques unigrammes et bigrammes paramétrée par μ et λ et, lorsque c'est plus simple, la représentation totalement vectorisée dans laquelle les deux types de caractéristiques ne sont pas distinguées et où θ désigne l'ensemble des paramètres du modèle.

2.3. *Estimation des paramètres*

Étant donné un corpus d'apprentissage constitué de N séquences étiquetées $\{\mathbf{x}^{(n)}, \mathbf{y}^{(n)}\}_{n=1}^{N}$ supposées statistiquement indépendantes, le critère de base pour estimer θ consiste à minimiser l'opposée de la log-vraisemblance conditionnelle des observations définie par (cf. [2]) :

$$\begin{aligned} l(\theta) &= -\sum_{n=1}^{N} \log p_\theta(\mathbf{y}^{(n)}|\mathbf{x}^{(n)}) \\ &= \sum_{n=1}^{N} \left\{ \log Z_\theta(\mathbf{x}^{(n)}) - \sum_{t=1}^{T_n} \sum_{k=1}^{K} \theta_k f_k(y_{t-1}^{(n)}, y_t^{(n)}, x_t^{(n)}) \right\} \end{aligned} \quad [7]$$

Dans les utilisations usuelles en traitement automatique des langues où le nombre total d'occurrences dans le corpus d'apprentissage n'est pas significativement plus grand que le nombre de paramètres, il est nécessaire d'inclure dans cette fonction objectif un terme de pénalisation (dit également « de régularisation ») visant à éviter le surapprentissage. Dans un premier temps, ce terme de régularisation est ignoré et nous nous concentrons sur les aspects computationnels de la minimisation de $l(\theta)$. Il est aisé de vérifier que $l(\theta)$ est convexe en θ et que le vecteur gradient du critère est donné par :

$$\frac{\partial l(\theta)}{\partial \theta_k} = \sum_{n=1}^{N} \sum_{t=1}^{T_n} \mathrm{E}_{p_\theta(\mathbf{y}|\mathbf{x}^{(n)})} f_k(y_{t-1}, y_t, x_t^{(n)}) - \sum_{i=1}^{N} \sum_{t=1}^{T_n} f_k(y_{t-1}^{(n)}, y_t^{(n)}, x_t^{(n)}) \quad [8]$$

où $\mathrm{E}_{p_\theta(\mathbf{y}|\mathbf{x}^{(n)})}$ désigne l'espérance sous la loi conditionnelle correspondant à la n-ième séquence, c'est-à-dire telle que :

$$\mathrm{E}_{p_\theta(\mathbf{y}|\mathbf{x}^{(n)})} f_k(y_{t-1}, y_t, x_t^{(n)}) = \sum_{(y',y)\in \overline{Y}\times Y} f_k(y', y, x_t^{(n)}) \, \mathrm{P}_\theta(y_{t-1} = y', y_t = y|\mathbf{x}^{(n)}) \quad [9]$$

Ainsi, tous les algorithmes d'optimisation numériques utilisés pour estimer les paramètres des champs aléatoires conditionnels, sont confrontés à la nécessité d'évaluer la constante de normalisation $Z_\theta(\mathbf{x}^{(n)})$ pour le calcul de $l(\theta)$ ainsi que les probabilités conditionnelles $\mathrm{P}_\theta(y_{t-1} = y', y_t = y|\mathbf{x}^{(n)})$, pour $(y', y) \in \overline{Y} \times Y$ et $t = 1, \ldots, T_n$, afin d'évaluer le gradient de $l(\theta)$. Le calcul de ces quantités est traditionnellement effectué par un algorithme dit *forward-backward* qui s'inspire de l'algorithme du même nom utilisé dans le cas des modèles de Markov cachés.

2.4. *Algorithme* forward-backward

En présence de dépendances d'ordre un dans [3], en particulier si on se limite à l'utilisation de caractéristiques de type unigramme et bigramme, le calcul des quantités nécessaires à l'évaluation de $l(\theta)$ et de son gradient peut être effectué par un algorithme dont le coût de calcul n'augmente que proportionnellement à la longueur T de la séquence mais est proportionnel au carré du nombre d'étiquettes. L'algorithme prend la forme suivante dans le cas d'une superposition de caractéristiques unigrammes et bigrammes.

$$\begin{cases} \alpha_1(y) = \exp(\mu_{y,x_1} + \lambda_{y_0,y,x_1}) \\ \alpha_{t+1}(y) = \sum_{y'} \alpha_t(y') \exp(\mu_{y,x_{t+1}} + \lambda_{y',y,x_{t+1}}) \end{cases} \quad \text{(Récursion avant)}$$

$$\begin{cases} \beta_{T_i}(y) = 1 \\ \beta_t(y') = \sum_{y} \beta_{t+1}(y) \exp(\mu_{y,x_{t+1}} + \lambda_{y',y,x_{t+1}}) \end{cases} \quad \text{(Récursion arrière)}$$

À la différence du cas des modèles de Markov cachés, il est important de noter que les variables α_t et β_t considérées séparément n'ont pas directement une interprétation probabiliste. Pour α_t par exemple, on a :

$$\alpha_t(y) = \sum_{(y_0,\ldots,y_{t-1}) \in Y^t, y_t = y} \exp\left\{ \sum_{s=1}^{t} \left(\mu_{y_s,x_s} + \lambda_{y_{s-1},y_s,x_s} \right) \right\}$$

Si bien que seul $\alpha_T(y) / \sum_{y \in Y} \alpha_T(y)$ peut légitimement s'interpréter comme $\mathrm{P}_\theta(y_T = y|\mathbf{x})$.

Par la suite, les probabilités jointes conditionnelles $\mathrm{P}_\theta(y_{t-1} = y', y_t = y|\mathbf{x})$ et la constante de normalisation $Z_\theta(\mathbf{x})$ s'obtiennent en normalisant, respectivement, $\alpha_t(y') \exp(\mu_{y,x_{t+1}} + \lambda_{y',y,x_{t+1}}) \beta_{t+1}(y)$ et $\alpha_T(y)$. Cet algorithme peut éventuellement être utilisé également en phase d'inférence (pour déterminer une séquence d'étiquettes) afin de calculer la probabilité marginale conditionnelle $\mathrm{P}_\theta(y_t = y|\mathbf{x})$ pour effectuer ce que l'on appelle du décodage position par position. Le plus souvent, cependant, cette tâche d'inférence est réalisée en utilisant l'algorithme dit de Viterbi qui constitue une variante de la récursion avant destinée à déterminer directement la séquence d'étiquettes la plus probable, celle qui maximise $p_\theta(\mathbf{y}|\mathbf{x})$, par programmation dynamique.

3. Optimisation coordonnée par coordonnée pour le critère avec pénalité L_1

3.1. *Choix de la pénalisation*

L'approche la plus commune pour estimer le paramètre θ consiste à ajouter un terme de pénalisation L_2 au critère de perte logarithmique défini en [7], auquel cas la fonction objectif à minimiser devient $l(\theta) + \rho_2 \|\theta\|_2^2$, où ρ_2 est un paramètre de

régularisation. Outre ses bonnes performances empiriques, l'intérêt pratique de cette approche est que les évaluations de la fonction objectif et de son gradient nécessitent les mêmes calculs que dans le cas de $l(\theta)$ (*cf.* section 2.3) et n'importe quelle approche numérique de minimisation d'une fonction différentiable et convexe, de surcroît sans contrainte de domaine, peut être utilisée. Les limitations principales de cette approche standard sont, d'une part, liées au temps d'exécution avec la nécessité de réaliser la récursion *forward-backward* pour toutes les séquences d'apprentissage lors de chaque évaluation de la fonction ou du gradient et, d'autre part, liées à l'empreinte mémoire du code du fait de la taille habituellement très grande du vecteur de paramètres. En pratique, ce deuxième aspect interdit l'utilisation d'algorithmes cherchant à estimer directement le hessien ou son inverse et se traduit par l'utilisation prépondérante d'algorithmes de type gradient conjugué ou quasi-Newton à mémoire limitée (de type L-BFGS, *Limited Memory BFGS*, en particulier).

La pénalisation L_2, si elle est efficace pour éviter le surapprentissage au moment de l'entraînement du modèle, ne réalise pas, à proprement parler de sélection de caractéristiques : les paramètres θ_k estimés sont tous non nuls et la sélection de caractéristiques fondée sur leur amplitude conduit à des performances relativement dégradées (voir section 4). Pour effectivement incorporer l'objectif de sélection de caractéristiques dans le terme de pénalisation, nous considérons l'inclusion d'un terme de pénalité additionnel de type L_1. La fonction objectif obtenue :

$$l(\theta) + \rho_1 \|\theta\|_1 + \frac{\rho_2}{2} \|\theta\|_2^2 \qquad [10]$$

est dite *elastic net* par (Zhou et Hastie, 2005) et comporte maintenant deux paramètres de régularisation ρ_1 et ρ_2[1]. Cette forme de pénalisation fournit un degré de liberté supplémentaire pour ajuster le compromis entre le caractère creux des solutions et la qualité des performances en généralisation. En particulier, elle permet d'atteindre, pour $\rho_1 = 0$ ou $\rho_2 = 0$ les solutions obtenues avec chacune des pénalités utilisée séparément, mais rend également accessible d'autres solutions qui sont peut-être plus intéressantes. Dans la mesure, toutefois, où le terme de pénalisation L_1 joue un rôle déterminant pour la sélection de caractéristiques, nous continuerons, par la suite, à parler de *modèle avec régularisation* L_1 pour désigner notre modèle.

3.2. *Algorithme d'optimisation coordonnée par coordonnée*

Le principe de l'approche proposée par (Friedman *et al.*, 2008) consiste à remarquer que si la minimisation directe du critère [10] est un problème délicat du fait de la non-différentiabilité de la fonction objectif en les points où l'un au moins des θ_k vaut zéro, la minimisation coordonnée par coordonnée d'une approximation quadratique locale de [10] est un problème très simple qui admet une résolution explicite. L'idée

1. (Zhou et Hastie, 2005) utilisent une paramétrisation différente de la régularisation qui, dans le type de problèmes considérés ici, s'est avérée plus difficile à régler car chaque paramètre joue simultanément sur les deux types de régularisation.

est que l'inefficacité intrinsèque de la minimisation coordonnée par coordonnée peut être compensée par l'extrême simplicité de la mise à jour à effectuer pour chaque coordonnée associée à la possibilité d'utiliser des schémas plus efficaces de balayage des coordonnées, possibilité dont on verra qu'elle est particulièrement attractive dans les cas des champs aléatoires conditionnels.

Étant donnée une valeur courante $\bar{\theta}$ du vecteur de paramètres, l'approximation quadratique locale vis-à-vis de la k-ième coordonnée prend la forme suivante :

$$l_{k,\bar{\theta}}(\theta_k) = C^{st} + \frac{\partial l(\bar{\theta})}{\partial \theta_k}(\theta_k - \bar{\theta}_k) + \frac{1}{2}\frac{\partial^2 l(\bar{\theta})}{\partial \theta_k^2}(\theta_k - \bar{\theta}_k)^2 + \rho_1|\theta_k| + \frac{\rho_2}{2}\theta_k^2 \quad [11]$$

En écrivant les conditions d'optimalité au premier ordre (dite Karush-Kuhn-Tucker), il est aisé de vérifier que le minimum de l'approximation quadratique locale [11] est atteint en :

$$\theta_k = \frac{s\left\{\frac{\partial^2 l(\bar{\theta})}{\partial \theta_k^2}\bar{\theta}_k - \frac{\partial l(\bar{\theta})}{\partial \theta_k}, \rho_1\right\}}{\frac{\partial^2 l(\bar{\theta})}{\partial \theta_k^2} + \rho_2} \quad [12]$$

où s désigne la fonction de seuillage progressif ou seuillage doux définie par :

$$s(z,\rho) = \begin{cases} z - \rho & \text{if } z > \rho \\ z + \rho & \text{if } z < -\rho \\ 0 & \text{sinon} \end{cases} \quad [13]$$

Il est intéressant de noter qu'une version alternative de la même idée est présentée dans (Dudík *et al.*, 2004), dans laquelle le comportement local de l est approché sous une forme équivalente au premier ordre, mais non quadratique, qui conduit elle aussi à une minimisation coordonnée par coordonnée explicite. Cette forme d'approximation repose cependant crucialement sur le fait que chaque coordonnée θ_k du vecteur de paramètres est multipliée par une caractéristique à valeur dans $\{0, 1\}$. Cette propriété n'est malheureusement pas vérifiée dans les cas des champs aléatoires conditionnels, puisque la k-ième coordonnée du vecteur de paramètres est multipliée par $\sum_{t=1}^{T} f_k(y_{t-1}, y_t, x_t)$. Ce terme peut être strictement supérieur à 1, même si f_k est à valeur dans $\{0, 1\}$, dès que la caractéristique correspondante est présente à plusieurs positions distinctes dans la séquence d'apprentissage.

3.3. *Applications aux champs aléatoires conditionnels*

Pour appliquer l'approche précédente dans le cas des champs aléatoires conditionnels, il est nécessaire de disposer des dérivées d'ordre un et deux de $l(\theta)$. La dérivée de premier ordre est définie par [8], les probabilités jointes conditionnelles étant calcu-

lables par l'algorithme *forward-backward* décrit dans la section 2.4. Nous considérons maintenant le cas de la dérivée seconde de $l(\theta)$. Un calcul direct donne :

$$\frac{\partial^2 l(\theta)}{\partial \theta_k^2} = \sum_{n=1}^{N} \left\{ E_{p_\theta(\mathbf{y}|\mathbf{x}^{(n)})} \left(\sum_{t=1}^{T_n} f_k(y_{t-1}, y_t, x_t^{(n)}) \right)^2 - \left(E_{p_\theta(\mathbf{y}|\mathbf{x}^{(n)})} \sum_{t=1}^{T_n} f_k(y_{t-1}, y_t, x_t^{(n)}) \right)^2 \right\} \quad [14]$$

Le premier terme est problématique, car il implique des termes qui ne dépendent pas uniquement des probabilités jointes conditionnelles $\mathrm{P}_\theta(y_{t-1} = y', y_t = y|\mathbf{x}^{(n)})$ et ne sont donc pas calculables à partir de la récursion *forward-backward*. Même si des solutions de calcul exactes existent (*cf.* chapitre 4 de (Cappé *et al.*, 2005)), celles-ci ne semblent pas praticables étant donné l'échelle des modèles auxquels nous nous intéressons ici. Nous proposons d'utiliser l'approximation suivante :

$$\frac{\partial^2 l(\theta)}{\partial \theta_k^2} \approx \sum_{n=1}^{N} \sum_{t=1}^{T_n} \left\{ \mathrm{E}_{p_\theta(\mathbf{y}|\mathbf{x}^{(n)})} f_k^2(y_{t-1}, y_t, x_t^{(n)}) - \left(\mathrm{E}_{p_\theta(\mathbf{y}|\mathbf{x}^{(n)})} f_k(y_{t-1}, y_t, x_t^{(n)}) \right)^2 \right\} \quad [15]$$

Cette approximation correspond à l'hypothèse que conditionnellement à $\mathbf{x}^{(n)}$, les caractéristiques $f_k(y_{t-1}, y_t, x_t^{(n)})$ et $f_k(y_{s-1}, y_s, x_s^{(n)})$ sont décorrélées dès que $s \neq t$. Lorsque l'on utilise des caractéristiques unigrammes ou bigrammes, cette approximation est exacte pour la n-ième séquence d'apprentissage dès lors que θ_k correspond soit à un paramètre unigramme $\mu_{y,x}$, soit à un paramètre bigramme $\lambda_{y',y,x}$ pour lequel le symbole x n'est présent qu'à une unique position dans la séquence d'apprentissage $x^{(n)}$. On remarque d'ailleurs également que si le symbole x n'est pas présent dans la séquence d'apprentissage $\mathbf{x}^{(n)}$, celle-ci ne contribue en aucune façon à la minimisation de l'approximation quadratique locale. Cette remarque essentielle nous permet, lorsque l'on met à jour un paramètre $\mu_{y,x}$ ou $\lambda_{y',y,x}$, de limiter la sommation dans [8] et [15] aux séquences dans lesquelles le symbole x apparaît, c'est-à-dire de n'effectuer la récursion *forward-backward* que pour les séquences correspondantes. En terme de temps de calcul, le gain est donc éventuellement très significatif, même s'il ne compense pas la nécessité de remettre à jour successivement chacune des coordonnées du vecteur de paramètres. L'algorithme correspondant est décrit ci-dessous (voir l'algorithme 1).

Algorithme 1 : Optimisation coordonnée par coordonnée

Initialiser θ;
tant que *Le critère de convergence n'est pas atteint* **faire**
  pour $k = 1 : K$ **faire**
    pour *Les séquences pour lesquelles la caractéristique* f_k *est active* **faire**
      Effectuer la récursion *forward-backward*;
    fin
    Évaluer $\partial l(\theta)/\partial\theta_k$ et $\partial^2 l(\theta)/\partial\theta_k^2$ en utilisant [8] et [15];
    Mettre à jour θ_k utilisant [12]–[13];
  fin
fin

3.4. *Mise à jour simultanée de blocs de coordonnées*

L'utilisation de l'algorithme décrit ci-dessus bute sur la très grande dimension du vecteur de paramètres utilisé dans les applications. Dans le cas d'une utilisation conjointe de caractéristiques unigrammes et bigrammes selon [5], le nombre total de paramètres est de $|Y||X|$ pour les caractéristiques unigrammes, plus $|Y|(|Y|+1)|X|$ pour les caractéristiques bigrammes. Même si la mise à jour de chaque coordonnée n'implique qu'un nombre réduit de séquences parmi l'ensemble des séquences d'apprentissage, la dimension excessive du vecteur de paramètres rend difficilement envisageable la mise à jour coordonnée par coordonnée. Pour imaginer des schémas plus efficaces de mise à jour des coordonnées bloc par bloc, il est important de noter que pour mettre à jour un paramètre de type unigramme $\mu_{y,x}$ ou de type bigramme $\lambda_{y',y,x}$, il est nécessaire d'effectuer la récursion *forward-backward* pour le sous-ensemble des séquences d'apprentissage qui comportent le symbole x. Or, on constate à l'examen de [8] et [15] que le coût de calcul de la dérivée première et de l'approximation de la dérivée seconde est marginal une fois que les probabilités jointes conditionnelles $\mathrm{P}_\theta(y_{t-1} = y', y_t = y|\mathbf{x}^{(n)})$ ont été obtenues pour l'ensemble des indices n pour lesquels la séquence $\mathbf{x}^{(n)}$ contient le symbole x. En d'autres termes, pour un surcoût de calcul marginal, il est possible d'évaluer simultanément [8] et [15] pour l'ensemble des paramètres $(\mu_{y,x})_{y\in Y}$ et $(\lambda_{y',y,x})_{(y',y)\in\overline{Y}\times Y}$. Cette observation conduit à regrouper les caractéristiques par blocs correspondant à l'ensemble des caractéristiques unigrammes ou bigrammes qui partagent un même symbole d'entrée x. Il est intéressant de constater que cette contrainte computationnelle conduit à choisir des blocs de caractéristiques qui sont orthogonaux à ceux utilisés dans le cas de la régression logistique par (Friedman *et al.*, 2008) dans lequel les caractéristiques sont regroupées par valeur commune de l'étiquette y. L'algorithme correspondant est donné ci-dessous (algorithme 2).

Algorithme 2 : Optimisation bloc par bloc

Initialiser θ;
tant que *Le critère de convergence n'est pas atteint* **faire**
  pour $x \in X$ **faire**
    pour *Les séquences contenant le symbole* x **faire**
      Effectuer la récursion *forward-backward*;
    fin
    Calculer;

$$\{\partial l(\mu,\lambda)/\partial \mu_{y,x}, \partial^2 l(\mu,\lambda)/\partial \mu^2_{y,x}\}_{y\in Y}$$

$$\{\partial l(\mu,\lambda)/\partial \lambda_{y',y,x}, \partial^2 l(\mu,\lambda)/\partial \lambda^2_{y',y,x}\}_{(y',y)\in Y^2}$$

    en utilisant ([8]) et [15];
    Mettre à jour $\{\mu_{y,x}\}_{y\in Y}$ et $\{\lambda_{y',y,x}\}_{(y',y)\in Y^2}$ simultanément en utilisant [12]–[13];
  fin
fin

Le coût de calcul associé à une itération de cet algorithme, l'itération correspondant à la mise à jour de toutes les coordonnées du vecteur de paramètres, est donc de l'ordre de $|X|$ (le nombre de symboles d'entrée) multiplié par le nombre moyen de séquences d'apprentissage contenant chaque symbole. Dans les expériences présentées dans la section 4, ce coût est en moyenne du même ordre de grandeur que celui associé à chaque itération d'un algorithme d'optimisation globale du vecteur θ qui requiert le traitement de l'ensemble des N séquences d'apprentissage pour l'évaluation du vecteur gradient de $l(\theta)$.

Un problème qui peut survenir lors de l'utilisation de l'algorithme 2 est celui de l'instabilité numérique qui se manifeste par une convergence peu régulière vers la solution pour certaines valeurs de l'initialisation. L'interprétation à donner de ce phénomène est liée à la mise à jour simultanée d'un bloc de coordonnées, ce qui revient à approcher le hessien de chaque bloc par une matrice diagonale dont les éléments diagonaux sont donnés par [14]. Il est connu que dans ce type d'algorithme de mise à jour par blocs avec calcul approché du hessien, il est en général nécessaire d'ajuster le pas de l'algorithme pour garantir la stabilité globale de l'optimum (Nocedal et Wright, 2006). Dans le cas qui nous préoccupe, la stabilité peut être garantie en recherchant, pour chaque coordonnée, la valeur $0 < \alpha_k \leq 1$ la plus grande possible du pas telle que la mise à jour :

$$\frac{s\left(\alpha_k^{-1}\frac{\partial^2 l(\bar{\theta})}{\partial \theta_k^2}\bar{\theta}_k - \frac{\partial l(\bar{\theta})}{\partial \theta_k}, \rho_1\right)}{\alpha_k^{-1}\frac{\partial^2 l(\bar{\theta})}{\partial \theta_k^2} + \rho_2}$$

conduise effectivement à une diminution de la fonction objectif[2]. Cette façon de procéder conduirait toutefois à une mise à jour au coût prohibitif, impliquant en particulier la nécessité de recalculer $l(\theta)$ de façon répétée lors de chaque mise à jour. Face à ce problème, nous avons utilisé une solution heuristique consistant à fixer le pas α globalement pour le bloc complet des paramètres remis à jour simultanément en s'ajustant sur la taille du plus grand pas potentiellement effectué par l'algorithme de Newton ignorant les termes de pénalités ; c'est-à-dire en utilisant :

$$\alpha^{-1} = \kappa \times \max\left\{1, \max\left(\left|\frac{\partial l(\bar{\theta})}{\partial \theta_k} \Big/ \frac{\partial^2 l(\bar{\theta})}{\partial \theta_k^2}\right|\right)\right\} \quad [16]$$

Cette heuristique utilisée avec $\kappa = 1,5$ conduit à un algorithme très stable, avec des pas de taille suffisamment grande pour ne pas trop ralentir la convergence : typiquement α^{-1} peut être de l'ordre de plusieurs centaines initialement lorsque le paramètre est très mal estimé mais se fixe, lorsque l'on approche de la convergence, à des valeurs de l'ordre de 5.

4. Expériences

4.1. *Données et tâches*

Pour cette étude, nous nous sommes intéressés à deux tâches standard, à savoir l'analyse syntaxique de surface (ou *chunking*) et l'extraction d'entités nommées, qui toutes deux peuvent se formaliser comme des tâches d'étiquetage séquentielles.

4.1.1. *L'analyse syntaxique de surface*

L'analyse syntaxique de surface (Abney, 1991) consiste à segmenter (non récursivement) une phrase en groupes syntaxiques et à typer les groupes identifiés. L'exemple ci-dessous, reproduit de (Tjong Kim Sang et Buchholz, 2000), illustre le type de résultat attendu.

> [NP He] [VP reckons] [NP the current account deficit] [VP will narrow]
> [PP to] [NP only £ 1.8 billion] [PP in][NP September]

La sortie d'un système de *chunking* est donc un parenthésage de la phrase d'entrée, dans laquelle chaque groupe est associé à une étiquette qui spécifie le type du groupe. Une telle analyse se réexprime sous la forme d'une tâche d'étiquetage de séquences, en associant aux mots de la phrase à analyser une étiquette qui indique si le mot débute un groupe de type X (étiquette B-X), est à l'intérieur d'un groupe de type X (étiquette I-X) ou bien encore à l'extérieur de tout groupe (étiquette O). Pour l'exemple précédent, on aurait alors :

2. On montre relativement aisément qu'avec une valeur constante de α_k, le seul point stable de l'algorithme coordonnée par coordonnée est bien le minimum de la fonction objectif, et ce, même en tenant compte du caractère approché du calcul de la dérivée seconde (*cf.* section 3.3)

He	reckons	the	current	account	deficit	will	narrow	...
B-NP	B-VP	B-NP	I-NP	I-NP	I-NP	B-VP	I-VP	...

Pour cette tâche, nous avons utilisé les données publiques de la campagne d'évaluation CoNLL 2000 (Tjong Kim Sang et Buchholz, 2000), qui contiennent une analyse de surface pour toutes les phrases apparaissant dans les sections 15 à 18 (pour l'apprentissage) et dans la section 20 (pour le test) du PennTreeBank. Ces analyses ont été dérivées automatiquement de l'analyse syntaxique profonde, en utilisant 11 types de groupes, soit un total de 23 ($2 \times 11 + 1$) étiquettes possibles. Les types de groupe sont très inégalement distribués, puisque les trois types principaux (NP, VP, et PP) représentent plus de 90 % des occurrences. Les données contiennent, en plus d'une tokenisation prédéfinie des phrases d'entrée, les étiquettes morphosyntaxiques associées, tirées de nouveau du PennTreeBank : ces deux types d'information sont disponibles pour prédire les frontières de syntagmes[3].

Dans cette étude, à visée principalement méthodologique, nous nous limitons à utiliser des caractéristiques simples construites à partir des informations directement disponibles dans les jeux de données. Nous nous intéresserons en particulier à évaluer l'apport de tests conjoints sur des paires d'étiquettes et sur les observations, qui donnent lieu à un très grand nombre de caractéristiques et qui sont donc rarement utilisés en pratique. Pour atteindre des performances optimales ((Sha et Pereira, 2003) obtient une F-mesure de 94,4, en utilisant des CRF d'ordre 2), il semble indispensable d'incorporer d'autres caractéristiques, en particulier typographiques et morphologiques. De telles caractéristiques sont utilisées dans de nombreux systèmes ayant pris part à la campagne d'évaluation de 2000 ; pour mémoire, les performances (en F-mesure) de ces systèmes s'échelonnaient entre 94,1 et 85,7 ; la majorité des systèmes se situe au-dessus de 91,5.

4.1.2. *Le repérage des entités nommées*

Le repérage des entités nommées est une autre tâche intermédiaire standard en traitement automatique, qui consiste à repérer des groupes ou syntagmes qui correspondent à des entités nommées, c'est-à-dire essentiellement ceux qui correspondent à des noms de personnes, d'entreprises et de lieux, ainsi qu'à divers autres types de groupes ayant un statut linguistique particulier, tels que les dates, les montants, etc.

Nous avons, pour ces expériences, utilisé des données publiques issues de la campagne d'évaluation CoNLL 2003 (Tjong Kim Sang et de Meulder, 2003), qui s'intéresse au repérage de 4 types d'entités (personnes [PER], lieux [LOC], organisations [ORG] et autres [MISC]) dans des textes journalistiques en anglais et allemand. Seules les données en anglais, qui proviennent du corpus Reuters, ont été utilisées. Comme précédemment, cette tâche se formalise comme une tâche d'étiquetage de séquences, en assignant à chaque mot une étiquette B-X (respectivement I-X) s'il se trouve au début (respectivement à l'intérieur) d'une entité de type X ; tous les autres mots re-

3. On se reportera aux tables 5 et 6 en annexe pour une description des étiquettes présentes dans ces données.

	Chunking		Entités nommées		
	Phrases	Tokens	Articles	Phrases	Tokens
Entraînement	9 836	211 727	946	14 987	203 621
Developpement	-	-	216	3 466	51 362
Test	2 012	47 377	231	3 684	46 435

Tableau 1. *Détail des corpus utilisés*

çoivent l'étiquette O. Un exemple de phrase étiquetée selon ce schéma est reproduit ci-dessous :

U.N.	official	Ekeus	heads	for	Baghdad
B-ORG	O	B-PER	O	O	B-LOC

Pour cette tâche, trois ensembles de données sont disponibles, correspondant respectivement au corpus d'apprentissage, de développement [test-A] et de test [test-B][4]. On dispose de surcroît d'un large corpus de données non étiquetées, destiné à évaluer l'apport de techniques d'apprentissage non supervisé. Seuls les trois ensembles de données annotées, qui contiennent, en sus d'une tokenisation de la phrase d'entrée, des étiquettes morphosyntaxiques et des frontières de syntagmes, ont été utilisés. Comme précédemment, nous n'utilisons que des jeux de caractéristiques directement dérivées des données, à l'exclusion de toute autre source d'information (morphologie, dictionnaires de noms propres, de pays, etc). Pour cette tâche, les systèmes d'annotation automatique évalués dans (Tjong Kim Sang et de Meulder, 2003) présentent des performances qui s'échelonnent entre 88,8 à 60,1 de F-mesure, une majorité de systèmes obtenant des scores supérieurs à 84.

4.1.3. *Protocole expérimental*

Pour toutes nos expériences, nous avons utilisé deux implémentations différentes des CRF : d'une part l'implémentation publique réalisée par Taku Kudo[5], avec une régularisation L_2 ; d'autre part notre propre implantation des CRF en Matlab, qui implémente la régularisation L_1 par descente coordonnée par coordonnée. Pour des raisons d'efficacité, seule la méthode de mise à jour par bloc (voir section 3.4) est utilisée dans ces expériences. Pour les deux tâches, les performances sont usuellement mesurées en termes de taux d'erreur, de précision, de rappel et de F-mesure. La première de ces mesures sera majoritairement utilisée dans la suite. Le tableau 1 détaille les différents corpus utilisés.

4. Le jeu de test désigné dans la suite [test-B] est notoirement difficile, car il correspond à des articles collectés très postérieurement aux autres articles du corpus d'apprentissage et de développement [test-A]. Suivant une pratique courante, nous avons traité le corpus de développement comme un second corpus de test et nous donnerons les performances sur les deux corpus.

5. `http://crfpp.sourceforge.net/`

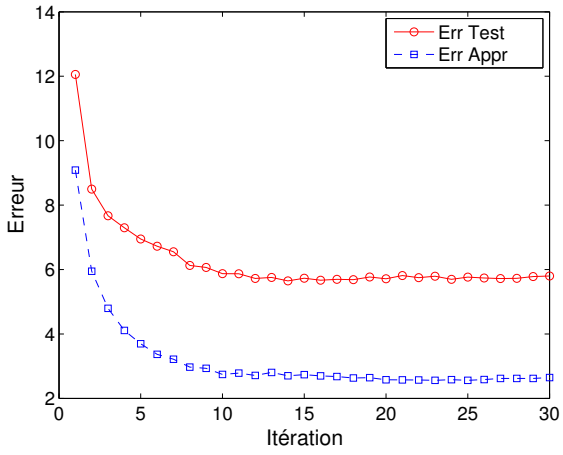


Figure 1. *Convergence de l'optimisation coordonnée par coordonnée (CoNNL 2000) Ces courbes représentent le taux d'erreur obtenu sur les données d'apprentissage et de test en fonction du nombre d'itérations. Les caractéristiques utilisées sont de la forme* $(y_t, x_t^d) + (y_{t-1}, y_t, x_t^d)$*, pour* $d \in \{1, 2\}$.

Rappelons enfin que pour les deux tâches, l'observation $\mathbf{x}$ est une séquence multidimensionnelle : dans le cas du *chunking*, nous disposons de deux composantes (le token courant et son étiquette morphosyntaxique), désignés respectivement par x_t^1 et x_t^2 ; dans le cas des entités nommées, nous utilisons trois composantes, puisqu'en plus du mot et de son étiquette morphosyntaxique, un marquage syntaxique de surface est également disponible, noté x_t^3.

4.1.4. *Quelques remarques générales*

Tous les résultats ci-après sont donnés pour des valeurs optimales[6]des paramètres de régularisation ρ_1 et ρ_2 ; pour les résultats obtenus en utilisant la méthode de descente coordonnée par coordonnée, nous avons réalisé 30 itérations de l'algorithme, les performances en généralisation semblant se stabiliser autour de cette valeur (voir figure 1).

6. D'un point de vue méthodologique, il aurait certainement été préférable d'opérer ce réglage sur un corpus de développement indépendant du test, et de répéter ces mesures sur plusieurs répartitions différentes des données entre apprentissage et test . Comme nous ne prétendons pas présenter une méthode dont les performances en généralisation seraient meilleures que celles d'autres méthodes, nous nous en sommes tenus ici à des mesures de performances plus simples à mettre en œuvre, même si elles conduisent à une surestimation des performances réelles.

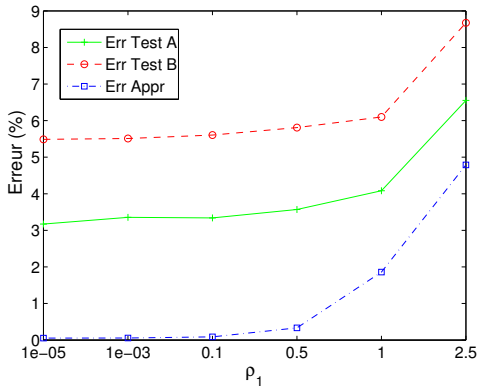


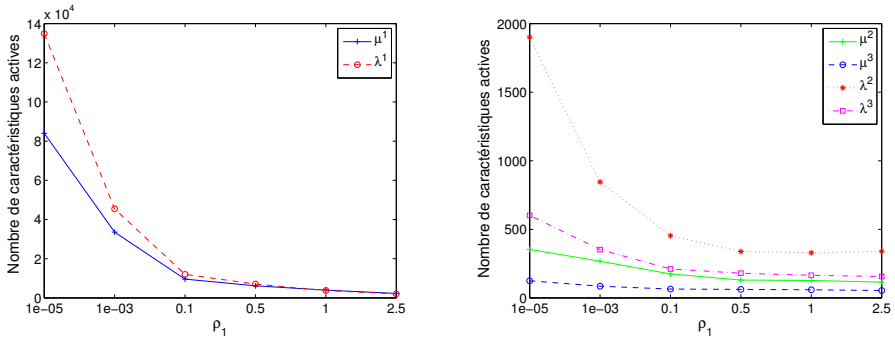
Figure 2. *Réglage du paramètre de régularisation L_1 (CoNLL 2003).*
La figure représente l'évolution des performances en généralisation en fonction de ρ_1.

Si le choix du paramètre de régularisation ρ_1 est crucial pour réguler le nombre de caractéristiques extraites, il existe une large plage de valeurs dans laquelle les performances en généralisation restent stables, alors que le nombre de caractéristiques actives varie beaucoup. Du point de vue des seules performances, on peut se contenter de fixer grossièrement ρ_1 (voir la figure 2) ; du point de vue de la complexité du modèle appris, il importe de régler ce paramètre avec plus de finesse (voir la figure 3). En revanche, *pour la plage de valeurs de ρ_1 considérée*, les performances sont peu sensibles au choix particulier de ρ_2, qui a été fixé à une valeur faible (10^{-5}). Pour des valeurs plus faibles de ρ_1, il aurait été indispensable de régler plus finement cette valeur.

4.2. *Sélection de caractéristiques : l'impact sur les performances*

4.2.1. *Sélection heuristique et exacte des caractéristiques*

Il existe plusieurs manières de sélectionner les caractéristiques de façon heuristique. L'une, déjà mentionnée, consiste à fixer des seuils de fréquence d'apparition et à éliminer *a priori* toutes les caractéristiques trop rares. Une deuxième approche consiste à ordonner les caractéristiques en fonction de leur association statistique avec la variable à prédire et à ne conserver que les caractéristiques qui présentent le plus fort degré d'association. Pour mettre en œuvre ici cette approche, nous avons utilisé comme mesure d'association *l'information mutuelle*, suivant les propositions de



(a) (b)

Figure 3. *Réglage du paramètre de régularisation L_1 (CoNLL 2003) Les figures donnent l'évolution du nombre de caractéristiques unigrammes et bigrammes extraites, en distinguant celles qui portent sur les mots (a) et celles qui portent sur les étiquettes morphosyntaxiques et syntaxiques (b).*

(Yang et Pedersen, 1997)[7], en l'adaptant pour sélectionner des caractéristiques, plutôt que des variables. Concrètement, la sélection des caractéristiques de type unigramme repose sur le calcul, pour chaque valeur possible de x et y, de l'information mutuelle entre les variables aléatoires booléennes $V_x = \mathbb{1}(X_t = x)$ et $V_y = \mathbb{1}(Y_t = y)$ selon :

$$I(V_x, V_y) = \sum_{v_x=0,1} \sum_{v_y=0,1} p(v_x, v_y) \log(\frac{p(v_x, v_y)}{p(v_x)p(v_y)}) \quad [17]$$

Ces valeurs sont calculées en approximant les distributions par les distributions empiriques, ce qui implique ici d'incorporer un terme de lissage minimal pour éviter les problèmes numériques. Il en va de même pour les caractéristiques bigrammes, qui demandent de calculer l'information mutuelle entre les variables aléatoires $\mathbb{1}(X_t = x, Y_{t-1} = y')$ et $\mathbb{1}(Y_t = y)$ pour toutes les valeurs possibles de x, y' et y. Ces deux stratégies de sélection *a priori* ont été mises en œuvre en utilisant la pénalité L_2 pour régulariser le modèle. Il existe une troisième manière de procéder, qui demande d'entraîner un CRF de manière conventionnelle (toujours avec pénalité L_2), puis à supprimer *a posteriori* les caractéristiques qui donnent lieu à des paramètres trop petits. Ces trois approches sont comparées avec l'utilisation d'une régularisation L_1 sur la figure 4, qui représente les taux d'erreur obtenus pour différents niveaux de filtrage, exprimés en log du nombre de caractéristiques actives dans le modèle.

7. Dans (Yang et Pedersen, 1997), l'information mutuelle est appelée *gain d'information* ; le terme d'*information mutuelle* étant utilisé, de façon très inhabituelle, pour désigner une autre quantité généralement appelée (en anglais) *Pointwise Mutual Information*.

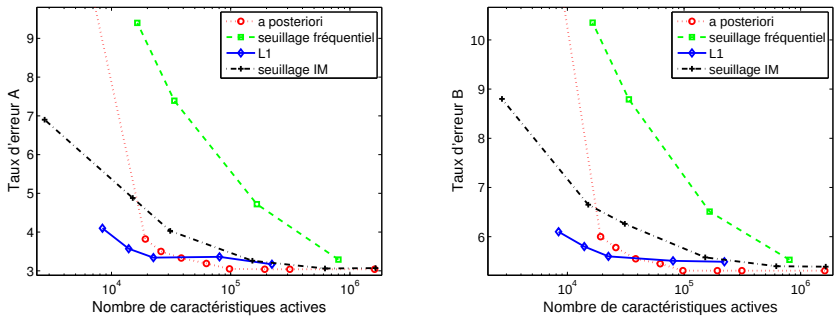


Figure 4. *Taux d'erreur en généralisation en fonction du nombre de caractéristiques retenues pour différentes techniques de sélection de caractéristiques. (CoNLL 2003, tests A et B)*

Cette figure appelle plusieurs commentaires : on note, d'une part, que l'utilisation d'une bonne méthode de sélection permet de filtrer l'immense majorité (en fait plus de 99 %) des caractéristiques sans incidence sur les performances. De ce point de vue, le filtrage *a posteriori* sur l'intensité des paramètres donne des performances équivalentes à l'utilisation de la pénalité L_1 pour une large plage de valeurs, l'avantage de notre méthode apparaissant clairement pour des forts taux de sélection. Le filtrage heuristique fondé sur l'information mutuelle donne des résultats systématiquement moins bons que ces deux méthodes, sauf pour des taux de sélection très forts. La stratégie la moins bonne consiste à utiliser un seuillage sur les fréquences.

4.2.2. *L'importance des caractéristiques bigrammes*

Comme expliqué *supra* (section 2.2), dans le modèle des CRF, l'ajout d'un nouveau test de type unigramme portant sur les observations se traduit par l'adjonction de l'ordre de $|Y|$ paramètres. L'utilisation de tests bigrammes, qui évaluent simultanément un test sur les observations et sur un couple d'étiquettes est beaucoup plus coûteux, puisque chaque nouveau test de ce type ajoute de $O(|Y|^2)$ nouveaux paramètres dans le modèle. Il n'est dès lors pas étonnant que ces derniers types de tests soient utilisés avec parcimonie, au point que l'implantation des CRF réalisée dans `CRFsuite` (Okazaki, 2007) ne prévoit pas l'utilisation de telles caractéristiques, en dehors de tests « génériques » sur les couples d'étiquettes. Leur inclusion permet pourtant d'améliorer les performances en généralisation, comme en attestent les résultats présentés dans le tableau 2, qui contraste les performances (mesurées par les taux d'erreur et le F-score) obtenues avec et sans caractéristiques bigrammes.

Pour les deux jeux de données, on observe un gain, particulièrement spectaculaire dans la tâche de *chunking*, lié à l'inclusion de paramètres $\lambda_{y',y}$ par rapport aux simples caractéristiques unigrammes. L'ajout de paramètres $\lambda_{y',y,x}$ apporte un gain supplé-

Caractéristiques	CoNLL 2000 Err. [F]	CoNLL 2003 Err. [F] (A)	CoNLL 2003 Err. [F] (B)
(y_t, x_t^d)	17,66 [77,3]	3,97 [73,1]	5,82 [64.0]
$(y_{t-1}, y_t) + (y_t, x_t^d)$	5,97 [90,4]	3,55 [81,6]	5,4 [74.9]
(y_{t-1}, y_t, x_t^d)	6,7 [89,5]	4,84 [73,9]	7,51 [63.1]
$(y_t, x_t^d) + (y_{t-1}, y_t, x_t^d)$	5,57 [91,1]	3,04 [83.0]	5,15 [73.1]

Tableau 2. *L'importance des caractéristiques bigrammes.*
Pour chaque jeu de test et chaque type de caractéristique, ce tableau donne le taux d'erreur obtenu et la F-mesure correspondance (entre crochets]). Pour CoNLL 2000, $d \in \{1, 2\}$*, pour CoNLL 2003,* $d \in \{1, 2, 3\}$.

mentaire en généralisation, *à condition d'inclure également les caractéristiques unigrammes correspondantes*, dont le rôle de lissage est ici clairement mis en évidence.

4.3. *Analyse des caractéristiques extraites*

Le tableau 3 permet de se faire une première idée des caractéristiques qui sont retenues par notre algorithme. Comme prévu, ce sont les caractéristiques lexicales, qui, fournissant le gros des tests sur les observations, donnent lieu à un élagage plus sévère. Ainsi, pour CoNLL 2000, seules 5 % des caractéristiques unigrammes portant sur un mot sont conservées, et seules 0,001 % des caractéristiques bigrammes sont jugées utiles. À l'inverse, les caractéristiques portant sur les étiquettes morphosyntaxiques et (pour CoNLL 2003), celles qui portent sur la localisation des frontières de syntagmes, sont plus fréquemment retenues. Cette expérience illustre un des principaux intérêts des techniques de sélection de caractéristiques : nombreuses sont les tâches pour lesquelles l'ajout d'information lexicale est décisif pour les performances, au prix d'une augmentation importante du nombre de paramètres. Ici, nous sommes capables de sélectionner les seuls tests lexicaux qui sont réellement utiles pour la tâche, en des proportions qui décourageraient l'identification « manuelle » de ces mots utiles.

Caractéristiques	CoNLL 2000 # Total	CoNLL 2000 # Extrait	CoNLL 2003 # Total	CoNLL 2003 # Extrait
(y_t, x_t^1)	496 547	5 057	242 320	10 550
(y_t, x_t^2)	1 012	485	386	186
(y_t, x_t^3)	-	-	144	67
(y_{t-1}, y_t, x_t^1)	11 917 128	9 439	2 180 880	13 585
(y_{t-1}, y_t, x_t^2)	24 288	1 591	3 312	488
(y_{t-1}, y_t, x_t^3)	-	-	1 296	214

Tableau 3. *Détail des caractéristiques extraites*
Pour CoNLL 2000, $\rho_1 = 0,5$*,* $\rho_2 = 1e^{-5}$ *; pour CoNLL 2003,* $\rho_1 = 0,1$*,* $\rho_2 = 1e^{-5}$.

Type	# occ (rg)	Type	# occ (rg)	Type	# occ (rg)	Type	# occ (rg)
,	10 770 (1)	"	1 503 (13)	about	439 (45)	*after*	224 (78)
and	3 578 (7)	*as*	899 (25)	*but*	371 (53)	*depressed*	7 (245)
down	154 (116)	*due*	87 (165)	*for*	1 857 (10)	*following*	32 (220)
half	56 (196)	*if*	193 (91)	*in*	3 398 (8)	*including*	80 (172)
is	1 415 (16)	*like*	122 (133)	*n't*	751 (31)	*pending*	14 (238)
not	266 (68)	*now*	156 (114)	*of*	5 201 (4)	*rather*	22 (230)
off	112 (142)	*on*	1 126 (19)	*or*	664 (36)	*today*	57 (195)
out	268 (66)	*right*	51 (201)	*tax*	115 (140)	*that*	1 790 (11)
the	9 219 (2)	*times*	52 (200)	*to*	5 045 (5)	*while*	123 (132)

Tableau 4. *Les mots les plus importants (CoNLL 2000) :* $\sum_y |\mu_{y,x}| > 10$
Pour chaque mot, on indique son nombre d'occurrences et son rang.

L'analyse des mots les plus importants, sélectionnés sur la base de leur contribution totale à l'ensemble des caractéristiques unigrammes qui les concernent (voir le tableau 4), permet de mettre en évidence cette décorrélation partielle entre fréquence et importance pour la tâche. De manière attendue pour cette tâche d'analyse syntaxique, beaucoup de mots grammaticaux figurent dans cette liste, notamment les principales prépositions, mais y figurent également des mots plus rares dont l'intérêt pour l'analyse syntaxique de surface était moins attendu (*depressed*, *tax*, *right*, etc.).

Une analyse similaire conduite sur la tâche CoNLL 2003 fait ressortir une liste de mots très différente, dans laquelle les mots rares sont très majoritaires. On retrouve en particulier des adjectifs de nationalité (*French*, *Dutch*, *Nigerian*, etc.) qui reçoivent le plus souvent l'étiquette B-MISC ou I-MISC ; cette étiquette ne peut pas être prédite sur la base des seules informations morphosyntaxiques, puisque la très grande majorité des adjectifs est étiquetée O. Ceci illustre de nouveau l'intérêt d'automatiser la sélection de caractéristiques, puisque, selon les tâches, ce sont des traits très différents qui s'avèrent utiles.

Il est possible d'analyser de manière plus globale les caractéristiques extraites. La figure 5 représente les associations positives (à gauche) et les plus négatives (à droite)[8] entre étiquettes de chunks et informations morphosyntaxiques, pour un modèle comportant les seules caractéristiques (x_t, y_t^d) et (y_{t-1}, y_t).

Côté gauche, se dégagent essentiellement quelques motifs horizontaux, qui se prêtent à une interprétation simple : pour les étiquettes B-NP[9] (ligne 6) et I-NP (ligne 17), sont particulièrement fortes les associations avec les étiquettes nominales (colonnes 19 à 22), les pronoms (25-26), les pronoms interrogatifs (41-43) ; les étiquettes B-VP et I-VP sont associées aux labels morphosyntaxiques associés aux

8. Le réglage des niveaux de gris, réalisé indépendamment pour chaque figure, masque le fait que l'intensité des caractéristiques positives est en général bien supérieure à celle des caractéristiques négatives.
9. Les tableaux 5 et 6 en annexe explicitent les correspondances entre indices et étiquettes.

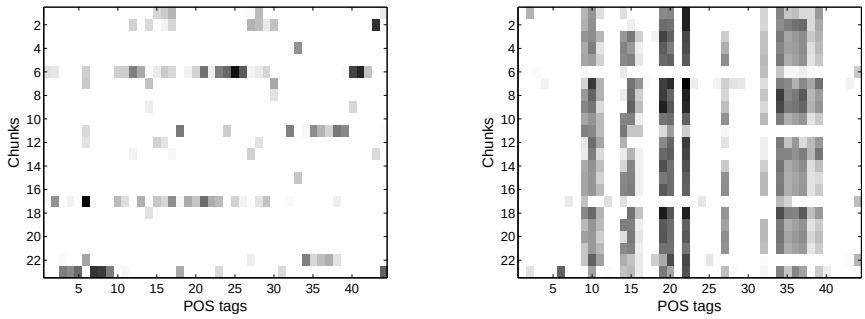


Figure 5. *Les caractéristiques unigrammes les plus significatives - CoNLL 2000*
Chaque case correspond à un paramètre $\mu_{x,y}$, le niveau de gris croît avec l'intensité.

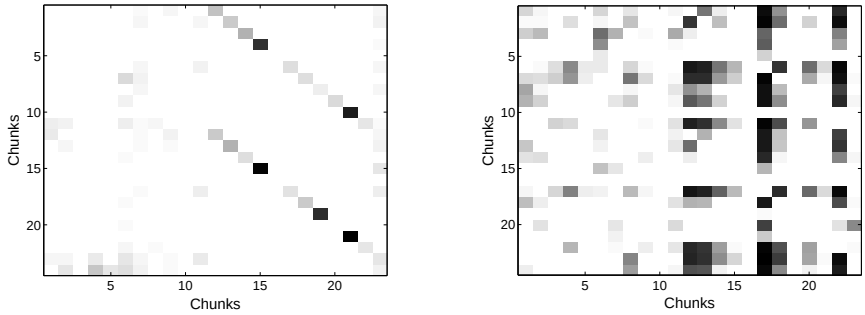


Figure 6. *Visualisation des caractéristiques bigrammes (CoNLL 2000)*
Chaque case correspond à un paramètre $\lambda_{y,y'}$, le niveau de gris croît avec l'intensité.

verbes (34 à 39). Côté droit, la prédominance des motifs verticaux résulte directement de la « normalisation » implicitement réalisée par l'utilisation d'une pénalisation : l'existence d'une association fortement positive pour un paramètre $\mu_{y,x}$ implique que les associations $\mu_{y',x}, y \neq y'$ seront négatives, et ce, *même si le couple* (y', x) *n'a jamais été observé dans les données d'apprentissage*.

Il est également intéressant de visualiser les dépendances bigrammes qui sont réellement utiles pour le modèle. La figure 6 représente (à gauche) les caractéristiques correspondant à des paramètres $\lambda_{y',y}$ positifs, et à droite celles pour lesquelles l'association est négative.

Le graphique de gauche révèle des dépendances très fortes le long de deux diagonales, qui correspondent aux contraintes déterministes qui pèsent sur les étiquettes

(I-X n'apparaît qu'après B-X ou I-X) : ainsi la cellule (1,12) correspond au couple d'étiquettes (B-ADJ,I-ADJ), la cellule (2,13) au couple (B-ADV,I-ADV), etc.

4.3.1. *La structure des caractéristiques*

Une dernière observation importante, qui ne sera pas, faute de place, illustrée ici, est l'existence d'une structure très remarquable dans l'ensemble des paramètres qui sont retenus. En particulier, on observe quasi systématiquement qu'une caractéristique bigramme $\lambda_{y',y,x}$ n'est retenue que si la caractéristique unigramme qui la généralise $\mu_{y,x}$ est significative. Ceci suggère qu'il est possible de développer des stratégies heuristiques incrémentales de sélection de caractéristiques, en sélectionnant tout d'abord les caractéristiques unigrammes, puis bigrammes, etc.

5. Discussion

Les techniques développées dans cet article visent à atteindre deux objectifs : le premier est de disposer de méthodes efficaces pour sélectionner des variables dans les CRF, donnant ainsi plus de latitude à l'utilisateur pour introduire dans les modèles toutes les variables qui lui semblent importantes, en laissant l'algorithme sélectionner celles qui sont réellement utiles pour les performances. L'analyse des caractéristiques sélectionnées permet également de confirmer ou d'infirmer certaines intuitions concernant l'intérêt de tel ou tel aspect du modèle. Le second objectif est d'exploiter le caractère très creux des vecteurs de paramètres appris pour réduire la complexité computationnelle de l'estimation et de l'inférence. Il nous reste beaucoup à faire pour réaliser pleinement ce second objectif, qui implique des changements profonds dans l'implémentation que nous utilisons actuellement.

5.1. *Sélection de caractéristiques : quelques propositions alternatives*

Dans cette section, nous positionnons notre approche par rapport à l'état de l'art, en nous concentrant délibérément sur le cas des modèles exponentiels et en particulier sur celui des CRF[10]. Si l'on omet les approches très empiriques évoquées ci-dessous, le premier travail sur la sélection de caractéristiques pour les CRF est probablement celui de (McCallum, 2003), qui, à la suite de (Della Pietra *et al.*, 1997), propose une heuristique gloutonne pour introduire des caractéristiques, sur la base de leur contribution à la log-vraisemblance. Partant d'un ensemble initialement vide, les caractéristiques les plus utiles au sens de ce critère sont progressivement ajoutées au modèle, par bloc de 1 000. Diverses approximations sont réalisées qui permettent de rendre cette stratégie computationnellement efficace. Les performances sont spectaculaires, puisqu'en ne conservant que 6 400 caractéristiques, les performances excèdent de 10 points celles

10. Il existe une très riche littérature sur la question de la sélection de caractéristiques en apprentissage artificiel dont nous ne rendrons pas compte ici. Nous renvoyons le lecteur intéressé par ces questions à (Guyon et Elisseeff, 2003) et aux références citées dans ce travail.

qui sont obtenues en conservant toutes les caractéristiques (environ 1 million dans son cas). Notre approche, comme celles présentées ci-dessous, permet de remplacer une approche gloutonne par un algorithme exact.

Parmi les méthodes heuristiques, signalons également le travail décrit dans (Dietterich *et al.*, 2004), qui est toutefois plus tourné vers la question de l'induction de caractéristiques que la sélection à proprement parler.

Les travaux les plus récents concernant la sélection de caractéristiques considèrent principalement l'utilisation de pénalisation L_1. La proposition de (Kazama et Tsujii, 2003), originellement formulée pour les modèles d'entropie maximale, s'étend directement aux CRF : pour l'essentiel, elle consiste à exprimer chaque paramètre θ comme une différence de deux termes positifs $\theta = \theta^+ - \theta^-$ dans l'équation [2], et à ajouter une pénalité de la forme $\rho(\theta^+ + \theta^-)$. Le programme d'optimisation qui en résulte se résout sans problème et aboutit à des solutions effectivement creuses (au sens où, à l'optimum, $\theta^+ \times \theta^- = 0$). Cette reformulation est simple à mettre en œuvre, mais elle conduit à un doublement du nombre de paramètres, et semble longue à converger (Andrew et Gao, 2007). Dans (Kudo *et al.*, 2004), une autre stratégie d'optimisation est adoptée pour utiliser une pénalisation L_1 : le résultat principal de cette étude est que, pour les CRF linéaires, les deux types de pénalisation (L_1 et L_2) donnent des résultats équivalents, avec environ 6 à 8 fois moins de caractéristiques quand on utilise une pénalité L_1. Dans (Andrew et Gao, 2007), un autre algorithme efficace (dénommé OWL-QN) est présenté pour réaliser l'optimisation de la fonction objectif avec pénalité L_1, qui se fonde sur une légère modification de l'algorithme L-BFGS (Liu et Nocedal, 1989). Cette modification s'appuie sur le fait que la fonction objectif pénalisée (i) reste différentiable en tout point θ qui ne contient pas de coordonnées nulles et (ii) qu'il est possible d'utiliser, même quand certaines coordonnées sont nulles, un pseudo-gradient dans lequel on annule toutes les composantes du gradient qui correspondent à des coordonnées nulles. Les auteurs montrent alors que cette implantation surpasse grandement en vitesse la proposition de (Kazama et Tsujii, 2003) et permet de sélectionner efficacement des caractéristiques pour des problèmes comportant plusieurs millions de paramètres, avec des performances identiques à celles que l'on obtient en utilisant la pénalisation L_2 (Gao *et al.*, 2007). Notons, pour finir, les travaux récents de (Tsuruoka *et al.*, 2009), qui proposent une version d'un algorithme de descente de gradient stochastique avec pénalisation L_1 : si cette méthode s'avère sensiblement plus rapide que OWL-QN, elle souffre des mêmes déficiences que les autres algorithmes de cette famille : en particulier la convergence n'est pas garantie et les paramètres obtenus ne sont pas toujours optimaux.

5.2. *Perspectives : optimiser l'apprentissage et l'inférence*

Travailler avec des vecteurs de caractéristiques très creux présente un intérêt computationnel évident en inférence : seules les caractéristiques utiles doivent être évaluées et manipulées durant l'inférence. De manière plus subtile, il est également possible de profiter du caractère clairsemé des paramètres *pendant l'apprentissage* pour

optimiser cette étape, qui reste extrêmement coûteuse en temps. La complexité de l'estimation est un problème bien documenté pour les CRF et a donc donné lieu à de multiples propositions : ainsi, dans (Pal *et al.*, 2006), les auteurs proposent d'utiliser des approximations durant le déroulement de l'algorithme *forward-backward* pour accélérer le calcul du gradient ; Cohn (2006) propose lui d'utiliser des schémas de partage des paramètres, permettant de réduire considérablement la complexité des modèles, et incidemment, de l'apprentissage ; les améliorations présentées dans (Vishwanathan *et al.*, 2006) reposent, quant à elles, sur l'utilisation d'un algorithme de descente de gradient stochastique pour optimiser la fonction objectif.

La stratégie d'optimisation présentée dans ce travail est efficace, au sens où l'optimisation sur chaque coordonnée est résolue analytiquement, et ne demande que l'examen de quelques-unes des instances d'apprentissage. En l'état, elle se compare, dans notre implantation MatLab, à l'utilisation de méthodes d'optimisation classiques avec pénalisation L_2. Son implantation peut toutefois être grandement améliorée, en remarquant que :

– le vecteur de paramètre est très creux, ce qui permet d'accélérer les récursions de l'algorithme *forward-backward* ;

– lorsqu'on s'intéresse à un seul paramètre (ou à seul bloc), correspondant à une observation en position t de la séquence $\mathbf{x}^{(n)}$, les récursions de l'algorithme *forward-backward* n'ont pas besoin d'être effectuées jusqu'à leur terme.

D'autres optimisations sont également envisageables pour accélérer l'apprentissage, consistant en particulier à faire croître incrémentalement le modèle (en diminuant progressivement ρ_1) : nous avons observé expérimentalement que dans ce schéma, initialiser les paramètres à l'étape k à partir des valeurs obtenues à l'étape $k-1$ accélère grandement la convergence. Une autre heuristique potentiellement utile consiste à ne plus réévaluer les dimensions qui ont été annulées à partir d'un certain nombre d'itérations, ou encore à rechercher d'autres façons de construire les blocs de paramètres.

Notre prochain travail va consister à développer une version optimisée de l'optimisation avec régularisation L_1 par descente coordonnée par coordonnée, de manière à pouvoir réaliser des tests comparatifs en vraie grandeur qui permettront de comparer l'efficacité computationnelle des différentes implémentations de la pénalisation L_1. Des résultats préliminaires de ce travail sont donnés dans (Sokolovska *et al.*, 2009). L'autre axe de développement va consister à utiliser cette implantation pour construire des modèles de très grande dimension, dans lesquels il ne sera plus nécessaire de se limiter dans le choix des caractéristiques, en particulier des caractéristiques bigrammes, qui sont incluses dans le modèle. Dans le même ordre d'idées, nous souhaitons utiliser le caractère creux des dépendances entre étiquettes qui sont extraites pour inclure certaines dépendances à plus long terme.

5.3. *Bilan et conclusion*

Dans cette étude, nous avons proposé un nouvel algorithme pour réaliser l'étape d'estimation dans les modèles CRF en présence d'une régularisation L_1. Nos résultats sont conformes à l'état de l'art, en ce sens qu'ils démontrent qu'il est possible d'élaguer très fortement les paramètres du modèle sans dégrader de manière significative les performances. En revanche, pour les données de tests et les caractéristiques utilisées, nous n'avons pas observé de cas où l'utilisation d'une pénalité L_1 améliore les performances par rapport à l'utilisation d'une pénalité L_2. Nous avons également comparé cette méthode de sélection avec des méthodes heuristiques usuellement utilisées, et démontré sa supériorité empirique.

L'autre contribution de ce travail a été d'analyser les caractéristiques qui sont extraites, ce qui nous a permis de constater qu'elles avaient le plus souvent une interprétation linguistique claire (pour les associations positives) ; à l'inverse, les paramètres négatifs sont plus délicats à interpréter, dans la mesure où la valeur de ces caractéristique est le plus souvent fixée par compensation avec une ou plusieurs associations positives.

Pour ces deux raisons, il semble que l'utilisation d'une pénalité L_1 permette simultanément d'estimer de manière robuste des modèles discriminants, tout en effectuant une sélection parmi les caractéristiques les plus utiles. Ces deux facteurs plaident simultanément pour son utilisation systématique (à la place d'une pénalité L_2) dans tous les problèmes d'apprentissage dans lesquels le nombre de caractéristiques potentiellement utiles est très grand, comme c'est souvent le cas dans des applications de traitement des langues.

Remerciements

Ce travail a été en partie financé par l'ANR dans le cadre des projets CroTal (ANR-07-MDCO-003) et MGA (ANR-07-BLAN-0311-02). Les auteurs remercient les relecteurs anonymes, dont les remarques ont contribué à l'amélioration de ce texte.

6. Bibliographie

Abney S., « Parsing By Chunks », *Principle-Based Parsing*, Kluwer Academic Publishers, p. 257-278, 1991.

Andrew G., Gao J., « Scalable Training of L1-Regularized Log-Linear Models », *Proceedings of the 24th international conference on Machine learning (ICML)*, Corvalis, Oregon, p. 33-40, 2007.

Bakir G. H., Hofmann T., Schölkopf B., Smola A. J., Taskar B., Vishwanathan S. V. N. (eds), *Predicting Structured Data*, Neural Information Processing, The MIT Press, 2007.

Bender O., Och F. J., Ney H., « Maximum Entropy Models for Named Entity Recognition », *Proceedings of CoNLL-2003*, Edmonton, Canada, p. 148-151, 2003.

Blunsom P., Cohn T., « Discriminative word alignment with conditional random fields », *ACL-44 : Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics*, Morristown, NJ, USA, p. 65-72, 2006.

Cappé O., Moulines E., Rydén T., *Inference in Hidden Markov Models*, Springer, 2005.

Charniak E., Johnson M., « Coarse-to-Fine n-Best Parsing and MaxEnt Discriminative Reranking », *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL'05)*, Ann Arbor, Michigan, p. 173-180, 2005.

Chen S. F., Rosenfeld R., « A survey of smoothing techniques for maximum entropy models », *IEEE transactions on Speech and Audio Processing*, vol. 8, n° 2, p. 37-50, 2000.

Cohn T., « Efficient Inference in Large Conditional Random Fields », *Proceedings of the 17th European Conference on Machine Learning*, Berlin, p. 606-613, September, 2006.

Collins M., Duffy N., « New Ranking algorithms for parsing and tagging : kernels over discrete structures and the voted perceptron », *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*, Philadelphia, PA, p. 489-496, 2002.

Cortes C., Vapnik V., « Support-Vector Networks », *Machine Learning*, vol. 20, p. 273, 1995.

Della Pietra S., Della Pietra V. J., Lafferty J. D., « Inducing Features of Random Fields », *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 19, n° 4, p. 380-393, 1997.

Dietterich T. G., Ashenfelter A., Bulatov Y., « Training Conditional Random Fields via Gradient Tree Boosting », *ICML*, Banff, Canada, 2004.

Dudík M., Phillips S. J., Schapire R. E., « Performance Guarantees for Regularized Maximum Entropy Density Estimation », *in* J. Shawe-Taylor, Y. Singer (eds), *Proceedings of the 17th annual Conference on Learning Theory, (COLT 2004), Banff, Canada*, vol. 3120 of *Lecture Notes in Computer Science*, Springer, p. 472-486, 2004.

Finkel J. R., Kleeman A., Manning C. D., « Efficient, Feature-based, Conditional Random Field Parsing », *Proceedings of ACL-08 : HLT*, Columbus, Ohio, p. 959-967, 2008.

Freund Y., Schapire R. E., « Experiments with a New Boosting Algorithm », *Proceedings of the thirteenth International Conference on Machine Learning*, Morgan Kaufmann, p. 148-156, 1996.

Friedman J., Hastie T., Höfling H., Tibshirani R., « Pathwise Coordinate Optimization », *Annals of Applied Statistics*, vol. 1, n° 2, p. 302-332, 2007.

Friedman J., Hastie T., Tibshirani R., Regularization Paths for Generalized Linear Models via Coordinate Descent, Technical report, Department of Statistics, Stanford University, 2008.

Gao J., Andrew G., Johnson M., Toutanova K., « A Comparative Study of Parameter Estimation Methods for Statistical Natural Language Processing », *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, Prague, Czech republic, p. 824-831, 2007.

Guyon I., Elisseeff A., « An Introduction to Variable and Feature Selection », *Journal of Machine Learning Research*, vol. 3, p. 1157-1182, 2003.

Hastie T., Tibshirani R., Friedman J., *The Elements of Statistical Learning : Data Mining, Inference and Prediction*, Springer, 2001.

Jousse F., Gilleron R., Tellier I., Tommasi M., « Conditional Random Fields for XML Trees », *Proceedings of the ECML Workshop on Mining and Learning in Graphs*, 2006.

Kazama J., Tsujii J., « Evaluation and extension of maximum entropy models with inequality constraints », *Proceedings of the 2003 conference on Empirical methods in natural language processing*, Morristown, NJ, USA, p. 137-144, 2003.

Koo T., Globerson A., Carreras X., Collins M., « Structured Prediction Models via the Matrix-Tree Theorem », *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, Association for Computational Linguistics, Prague, Czech Republic, p. 141-150, June, 2007.

Kudo T., Yamamoto K., Matsumoto Y., « Applying Conditional Random Fields to Japanese Morphological Analysis », *in* D. Lin, D. Wu (eds), *Proceedings of EMNLP 2004*, Barcelona, Spain, p. 230-237, 2004.

Lafferty J., McCallum A., Pereira F., « Conditional random fields : probabilistic models for segmenting and labeling sequence data », *Proceedings of the International Conference on Machine Learning (ICML)*, Morgan Kaufmann, San Francisco, CA, p. 282-289, 2001.

Liu D. C., Nocedal J., « On the Limited Memory BFGS Method for Large Scale Optimization », *Mathematical Programming*, vol. 45, p. 503-528, 1989.

McCallum A., « Efficiently Inducing Features of Conditional Random Fields », *Proceedings of the conference Uncertainty in Artificial Intelligence (UAI)*, Acapulco, Mexico, 2003.

McCallum A., Li W., « Early Results for Named Entity Recognition with Conditional Random Fields, Feature Induction and Web-Enhanced Lexicons », *Proceedings of CoNLL-2003*, Edmonton, Canada, p. 188-191, 2003.

Nocedal J., Wright S., *Numerical Optimization*, Springer, 2006.

Okazaki N., « CRFsuite : A fast implementation of Conditional Random Fields (CRFs) », 2007, `http://www.chokkan.org/software/crfsuite/`.

Pal C., Sutton C., McCallum A., « Sparse Forward-Backward using Minimum Divergence Beams for Fast Training of Conditional Random Fields », *Proceedings of the International Conference on Acoustics, Speech, and Signal Processing (ICASSP), 2006*, Toulouse, France, 2006.

Rathnaparkhi A., Maximum Entropy Models for Natural Language Ambiguity Resolution, PhD thesis, University of Pennsylvania, 1998.

Riezler S., Vasserman A., « Incremental Feature Selection and l1 Regularization for Relaxed Maximum-Entropy Modeling », *in* D. Lin, D. Wu (eds), *Proceedings of EMNLP 2004*, Association for Computational Linguistics, Barcelona, Spain, p. 174-181, July, 2004.

Rozenknop A., Modèles syntaxiques probabilistes non-gènératifs, PhD thesis, Dpt. d'informatique, École Polytechnique Fédérale de Lausanne, 2002.

Sha F., Pereira F., « Shallow Parsing with Conditional Random Fields », *Proceedings of Human Language Technology-NAACL 2003*, Edmonton, Canada, p. 213-220, 2003.

Sokolovska N., Lavergne T., Cappé O., Yvon F., « Efficient Learning of Sparse Conditional Random Fields for Supervised Sequence Labelling », 2009, `http://hal.archives-ouvertes.fr/hal-00414107`.

Sutton C., McCallum A., « An Introduction to Conditional Random Fields for Relational Learning », *in* L. Getoor, B. Taskar (eds), *Introduction to Statistical Relational Learning*, The MIT Press, Cambridge, MA, 2006.

Tibshirani R., « Regression Shrinkage and Selection via the Lasso », *J.R.Statist.Soc.B*, vol. 58, n° 1, p. 267-288, 1996.

Tjong Kim Sang E. F., Buchholz S., « Introduction to the CoNLL-2000 Shared Task : Chunking », *Proceedings of CoNLL-2000 and LLL-2000*, Lisbon, Portugal, p. 127-132, 2000.

Tjong Kim Sang E. F., de Meulder F., « Introduction to the CoNLL-2003 Shared Task : Language-Independent Named Entity Recognition », *Proceedings of CoNLL-2003*, Edmonton, Canada, p. 155-158, 2003.

Toutanova K., Klein D., Manning C. D., Singer Y., « Feature-rich part-of-speech tagging with a cyclic dependency network », *NAACL '03 : Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology*, Morristown, NJ, USA, p. 173-180, 2003.

Toutanova K., Manning C. D., « Enriching the knowledge sources used in a maximum entropy part-of-speech tagger », *Proceedings of the 2000 Joint SIGDAT conference on Empirical methods in natural language processing and very large corpora*, Morristown, NJ, USA, p. 63-70, 2000.

Tsuruoka Y., Tsujii J., Ananiadou S., « Stochastic Gradient Descent Training for L1-regularized Log-linear Models with Cumulative Penalty », *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, Suntec, Singapore, p. 477-485, 2009.

Vishwanathan S. V. N., Schraudolph N. N., Schmidt M., Murphy K., « Accelerated Training of Conditional Random Fields with Stochastic Gradient Methods », *Proceedings of the 23th International Conference on Machine Learning*, ACM Press, New York, NY, USA, p. 969-976, 2006.

Yang Y., Pedersen J. O., « A comparative study on feature selection in text categorization », *in* D. H. Fisher (ed.), *Proceedings of ICML-97, 14th International Conference on Machine Learning*, Morgan Kaufmann Publishers, San Francisco, US, Nashville, US, p. 412-420, 1997.

Zhou H., Hastie T., « Regularization and variable selection via the elastic net », *J. Royal. Stat. Soc. B.*, vol. 67, n° 2, p. 301-320, 2005.

Annexes

Description	Id	Label	Description	Id	Label
Conjonction de coordination	9	CC	Nombre cardinal	10	CD
Déterminant	11	DT	*there* existentiel	12	EX
Emprunt	13	FW	Préposition ou conjonction de subordination	14	IN
Adjectif	15	JJ	Adjectif comparatif	16	JJR
Adjectif superlatif	17	JJS	Modal	18	MD
Nom singulier	19	NN	Nom propre singulier	20	NNP
Nom propre pluriel	21	NNPS	Nom pluriel	22	NNS
Prédéterminant	23	PDT	Suffixe possessif	24	POS
Pronom personnel	25	PRP	Pronom possessif	26	PRP$
Adverbe	27	RB	Averbe comparatif	28	RBR
Adverbe superlatif	29	RBS	Particule	30	RP
Symbole	31	SYM	*to*	32	TO
Interjection	33	UH	Verbe infinitif	34	VB
Verbe, passé	35	VBD	Verbe, gérondif	36	VBG
Verbe, participe passé	37	VBN	Verbe fini présent	38	VBP
Verbe, présent 3e pers. sing.	39	VBZ	déterminant Wh	40	WDT
Pronom *Wh*	41	WP	Pronom possessif *Wh*	42	WP$
Adverbe *Wh*	43	WRB		44	“

Tableau 5. *Étiquettes morphosyntaxiques du PennTreeBank.*
Les identifiants 1 à 8 sont affectés aux signes de ponctuation.

Description	Id	Label	Id	Label
Syntagme adjectival	1	B-ADJP	12	I-ADJP
Syntagme adverbial	2	B-ADVP	13	I-ADVP
Syntagme coordination	3	B-CONJP	14	I-CONJP
Syntagme interjectif	4	B-INTJ	15	I-INTJ
Marqueur de liste	5	B-LST	16	I-LST
Syntagme nominal	6	B-NP	17	I-NP
Syntagme prépositionnel (*)	7	B-PP	18	I-PP
Syntagme particule (*)	8	B-PRT	19	I-PRT
Syntagme subordonné	9	B-SBAR	20	I-SBAR
Coordination *unlike*	10	B-UCP	21	I-UCP
Syntagme verbal	11	B-VP	22	I-VP
Hors syntagme	23	O		

Tableau 6. *Chunks (CoNLL 2000) d'après (Tjong Kim Sang et Buchholz, 2000)*
*(*) les particules (*up, down, out*) sont systématiquement étiquettées B-PRT, et constituent des syntagmes à elles seules. Il en va de même pour les prépositions.*