Annotation fonctionnelle de corpus arborés avec des Champs Aléatoires Conditionnels *

Erwan Moreau[1], Isabelle Tellier[2], Antonio Balvet[3], Grégoire Laurence[4], Antoine Rozenknop[1], Thierry Poibeau[1]
(1) LIPN, université de Paris 13, (2) LIFO, université d'Orléans, (3) UMR STL 8163, université de Lille (4) LIFL, Inria Lille-nord Europe
erwan.moreau@lipn.univ-paris13.fr, isabelle.tellier@univ-orleans.fr

Résumé. L'objectif de cet article est d'évaluer dans quelle mesure les "fonctions syntaxiques" qui figurent dans une partie du corpus arboré de Paris 7 sont apprenables à partir d'exemples. La technique d'apprentissage automatique employée pour cela fait appel aux "Champs Aléatoires Conditionnels" (Conditional Random Fields ou CRF), dans une variante adaptée à l'annotation d'arbres. Les expériences menées sont décrites en détail et analysées. Moyennant un bon paramétrage, elles atteignent une F1-mesure de plus de 80%.

Abstract. The purpose of this paper is to evaluate whether the "syntactic functions" present in a part of the Paris 7 Treebank are learnable from examples. The learning technic used is the one of "Conditional Random Fields" (CRF), in an original variant adapted to tree labelling. The conducted experiments are extensively described and analyzed. With good parameters, a F1-mesure value of over 80% is reached.

Mots-clés : fonctions syntaxiques, Conditional Random Fields, corpus arborés.

Keywords: syntactic functions, Conditional Random Fields, Treebanks.

1 Introduction

Nous nous intéressons dans cet article à l'application de techniques d'apprentissage automatique statistique pour identifier les *fonctions syntaxiques* (comme "sujet", "objet", "modifieur" etc.) présentes à l'intérieur de phrases françaises. Identifier ces fonctions est précieux pour des applications qui requièrent une analyse linguistique de haut niveau comme les systèmes "questions/réponses" ou ceux de traduction automatique (Blaheta, 2004). C'est une tâche très liée à l'analyse syntaxique et nous ne l'abordons ici qu'en partant d'un corpus de phrases déjà analysées. Par exemple, nous nous attendons à ce que la *structure syntaxique* d'une phrase comme "l'oiseau chante chaque matin" reflète le fait que le GN qui suit le verbe est un modifieur de la phrase et non un complément d'objet.

Ce problème est étudié depuis quelques années, principalement pour l'anglais (Blaheta & Charniak, 2000; Blaheta, 2004; Merlo & Musillo, 2005; Musillo & Merlo, 2005). Une tâche comparable, mais requérant l'identification plus fine des *rôles thématiques* (comme "agent", "patient"...) a aussi fait l'objet d'une compétition lors de la conférence CoNLL 2004 et 2005 (Carre-

∗. Ce travail a bénéficié du soutien de l'Agence Nationale de la Recherche ANR-07-MDCO-03

ras & Marquez, 2005) [1], avec des données issues du Penn Treebank (Marcus, 1993). Or, la communauté francophone du TALN dispose maintenant, avec le corpus arboré de Paris 7 (Abeillé, 2003), d'un outil de travail pertinent puisqu'une partie de ce corpus est étiquetée avec des fonctions syntaxiques. Notre objectif est donc d'apprendre un étiqueteur à partir de ces données.

Il existe peu de techniques d'apprentissage automatique capables de prendre en compte directement les structures de données arborescentes. Bien sûr, nous pourrions (comme l'avaient fait la plupart des participants à la compétition CoNLL), considérer les arbres comme des simples tableaux de données et appliquer des méthodes de classification sur chacun des nœuds des arbres indépendamment. A la place, nous voulons mettre en œuvre la théorie des "Champs Aléatoires Conditionnels" (Conditional Random Fields ou CRF). Les CRF (Lafferty *et al.*, 2001; Sutton & McCallum, 2006) sont des modèles statistiques d'annotation très puissants qui obtiennent d'excellentes performances (souvent les meilleures) pour la reconnaissance d'entités nommées (McCallum & Li, 2003), l'extraction d'informations (Pinto *et al.*, 2003), l'étiquetage Part-Of-Speech (Altun *et al.*, 2003) ou le shallow parsing (Sha & Pereira, 2003). Par rapport aux techniques de classification, ils présentent l'intérêt de modéliser des *dépendances entre annotations*. Ils ont aussi récemment été mis en œuvre pour faire de l'analyse syntaxique (Finkel *et al.*, 2008). Mais, dans toutes ces applications, les CRF ne sont employés que pour annoter des *séquences*. Or, ils ont aussi récemment été adaptés à *l'annotation d'arbres* (Gilleron *et al.*, 2006a; Gilleron *et al.*, 2006b; Jousse, 2007). Cette adaptation n'avait jusqu'à présent été testée que pour l'étiquetage de pages Web, en vue d'en extraire de l'information ou d'en transformer la structure (Jousse, 2007). Il est temps de la confronter à des données linguistiques.

Dans la suite de l'article, nous commençons par décrire les données du corpus de Paris 7. Nous présentons ensuite le modèle général des CRF, et son instanciation aux *séquences* ou aux *arbres*. Nous exposons enfin quelques-uns des résultats de nos expériences, avec divers paramétrages mais en ne faisant appel à aucune ressource linguistique externe. Ces expériences montrent que les taux de reconnaissance de l'ordre de 80% de F1-mesure, qui sont l'état de l'art sur les corpus anglais, sont aussi atteignables sur nos données.

2 Le corpus et ses annotations

Le French Treebank (par la suite noté FTB) est décrit dans (Abeillé, 2003). Ce corpus, constitué à partir d'extraits du journal *Le Monde* de 1989 à 1993, est, bien sûr, fortement influencé par le Penn Treebank, mais les objectifs d'application visés en priorité (la constitution d'une grammaire électronique du français) ont entraîné des descriptions syntaxiques de granularité plus fine. (Abeillé, 2003) recense pour le FTB 12 parties principales du discours : A(djectif), Adv, CL, C(onjonction), D(éterminant), ET(ranger), I(nterjections), N, P(réposition), PRO, PREF, et enfin V, auxquelles s'ajoute la ponctuation PONCT. Chacune des 12 catégories majeures possède des sous-catégories, ainsi que, le cas échéant, des indications morphologiques classiques, et des indications de mode/temps pour les verbes. Le FTB présente ainsi un jeu de catégories maximal de 218 éléments, en tenant compte de toutes les combinaisons valides.

Mais, surtout, le FTB est un corpus arboré : une analyse en constituants principaux est fournie, sous la forme de balises XML, pour chacune des 22 000 phrases du corpus. [2] Les constituants

1. http ://www.lsi.upc.edu/ srlconll/
2. Ces estimations chiffrées sont basées sur des décomptes réalisés sur les fichiers source du corpus.

distingués [3] sont les syntagmes dont la tête est l'une des catégories majeures suivantes : Nom, Verbe, Adjectif, Adverbe, Préposition. S'y ajoutent les propositions relatives, les subordonnées, les "autres subordonnées", les infinitives, les participiales et enfin les syntagmes coordonnés. Il est à noter que seul le noyau verbal est complètement décrit au regard de ses adjoints et de ses arguments, autrement dit les noms et adjectifs prédicatifs ne sont pas identifiés comme tels. Par ailleurs, les têtes de constituants ne sont pas explicitement identifiées.

Enfin, une partie du FTB, soit environ 9 000 phrases, comprend en outre des annotations fonctionnelles pour les constituants majeurs : SUJ (sujet), OBJ (objet), MOD (modifieur), A-OBJ (objet introduit par une préposition "à"), DE-OBJ (idem pour "de"), P-OBJ (objet prépositionnel), ATS (attribut du sujet) et ATO (attribut de l'objet), soit 8 fonctions différentes. Ces étiquettes sont présentes dans le corpus par le biais d'attributs affectés à certaines balises (VN, VP...). Le parti pris pour cet étiquetage fonctionnel a été une annotation de surface. Ainsi, par exemple, les sujets des infinitives ne sont pas notés : dans *je dis <PP> à Marie </PP> de venir*, le groupe prépositionnel PP est marqué comme A-OBJ de "dire", mais pas comme SUJ de "venir". Dans le corpus du challenge CoNLL au contraire, ces fonctions auraient figuré toutes les deux : chaque occurrence de verbe y donne lieu à une annotation spécifique complète, de telle sorte que le sujet d'un verbe peut très bien aussi être l'objet d'un autre. Ce choix a certainement rendu la tâche de CoNLL plus abordable. Nous n'avons pris en compte pour nos expériences que ces 9 000 phrases syntaxiquement analysées et fonctionnellement annotées.

3 Les CRF et leur adaptation aux arbres

3.1 Présentation générale des CRF

Les CRF permettent d'associer à une observation x une annotation y, en se basant sur un ensemble d'exemples étiquetés, c'est-à-dire un ensemble de couples (x, y). Souvent, x est une *séquence d'unités* (par exemple une suite de mots) et y la *séquence des étiquettes correspondante* (par exemple la suite de leur catégorie syntaxique). Mais, dans notre application, x et y proviendront tous les deux des arbres du FTB : x sera un arbre d'analyse syntaxique et y l'arbre de même structure dont les nœuds internes ne sont constitués que de fonctions syntaxiques (ou d'une étiquette $\perp$ signifiant "aucune fonction syntaxique"), comme dans la Figure 1.

Les CRF appartiennent à la famille des *modèles graphiques non dirigés*. Ils sont définis par X et Y, deux champs aléatoires décrivant respectivement l'observation x et son annotation y, et par un graphe $\mathcal{G} = (V, E)$ dont $V = X \cup Y$ est l'ensemble des nœuds (vertices) et $E \subseteq V \times V$ l'ensemble des arcs (edges). On note Y_v la variable aléatoire du nœud $v \in V$ dans Y. On dit que (X, Y) respecte la propriété de Markov si :

$$\forall v, p(Y_v|X, \{Y_w, w \neq v\}) = p(Y_v|X, \{Y_w, (Y_v, Y_w) \in E\})$$

Elle signifie que l'annotation d'un nœud Y_v ne dépend que des annotations Y_w des nœuds avec lesquels il est connecté dans $\mathcal{G}$ et de toute l'observation globale X. Les CRF rentrent dans ce cadre. Dans le graphe correspondant, chaque Y_v est donc toujours implicitement reliée à *toutes les variables du champ* X, ce qui explique qu'on omette la représentation des nœuds de X dans le dessin de $\mathcal{G}$. D'après le théorème de Hammersley-Clifford (Hammersley & Clifford, 1971),

3. Ces informations sont tirées des guides fournis aux annotateurs lors de la constitution du corpus.

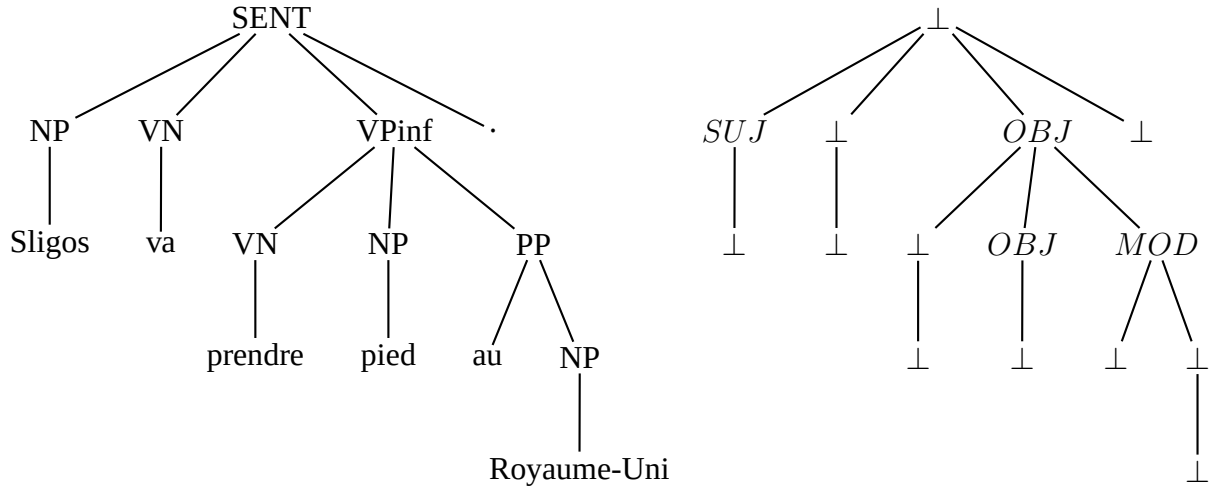

FIGURE 1 – un arbre observé x (à gauche) et son annotation y (à droite)

cette condition permet d'écrire :

$$p(y|x) = \frac{1}{Z(x)} \prod_{c \in \mathcal{C}} \psi_c(y_c, x) \; avec \; Z(x) = \sum_y \prod_{c \in \mathcal{C}} \psi_c(y_c, x)$$

où $\mathcal{C}$ est l'ensemble des cliques (sous-graphes complètement connectés) de $\mathcal{G}$ sur Y, y_c la configuration prise par les variables de Y dans la clique c, les ψ_c sont des "fonctions de potentiels" sur c et $Z(x)$ est un coefficient de normalisation.

Dans ces modèles, on dispose donc directement d'une formule pour calculer $p(y|x)$ sans avoir besoin de passer par le calcul de $p(x, y)/p(x)$, ce qui fait toute la différence entre les *modèles discriminants* (comme les CRF) et les *modèles génératifs* (comme les HMMs : Hidden Markov Models ou les PCFGs : Probabilistic Context-Free Grammars) dans lesquels il est nécessaire d'évaluer $p(x, y)$, c'est-à-dire de modéliser *comment l'observation x est produite conjointement à son annotation y*. Les modèles génératifs doivent modéliser *comment les observations x sont générées*, alors que les modèles discriminants ne font aucune hypothèse sur x.

Pour définir les CRF, (Lafferty *et al.*, 2001) ont proposé de donner aux fonctions de potentiels ψ_c la forme suivante :

$$\psi_c(y_c, x) = \exp\Big(\sum_k \lambda_k f_k(y_c, x, c)\Big)$$

Les fonctions f_k sont appelées *features* : elles sont définies à l'intérieur de chaque clique c et sont à valeurs réelles, mais souvent choisies pour donner un résultat binaire (0 ou 1). C'est à travers ces fonctions, fournies par l'utilisateur, que des ressources ou des connaissances sur le domaine peuvent être intégrées dans le modèle. Par exemple, l'association entre un mot x_i et une catégorie y_i à une même position i peut être testée par une feature $f_k(y_i, x_i, i)$ qui vaut 1 si (x_i, y_i) est présent dans un dictionnaire, 0 sinon. Par définition, la valeur de ces fonctions peut aussi dépendre de la valeur de x *n'importe où dans la donnée* (et pas uniquement à l'intérieur de la clique c), ce qui est impossible à exprimer dans les HMMs. Les poids λ_k, qui permettent d'accorder plus ou moins d'importance à chaque feature f_k, sont les paramètres du modèle : l'enjeu de la phase d'apprentissage est de fixer leur valeur. Dans un CRF, on a donc finalement :

$$p(y|x) = \frac{1}{Z(x)} \exp\Big(\sum_{c \in \mathcal{C}} \sum_k \lambda_k f_k(y_c, x, c)\Big) \; avec \; Z(x) = \sum_y \exp\Big(\sum_{c \in \mathcal{C}} \sum_k \lambda_k f_k(y_c, x, c)\Big)$$

Le premier problème associé aux CRF est celui de *l'inférence ou de l'apprentissage*, qui consiste à estimer les paramètres λ_k qui rendent le mieux compte d'un échantillon S d'observations annotées : $S = \{(x^j, y^j)_{1 \leq j \leq n}\}$. Classiquement, on cherche l'ensemble des paramètres qui maximisent la log-vraisemblance du modèle. Des techniques de descente de gradient sont utilisées pour estimer cet ensemble optimal. Le second problème est celui de *l'annotation* qui consiste, une fois les paramètres du CRF fixés, à trouver la valeur de y la plus probable associée à une nouvelle observation x, autrement dit à trouver $argmax_y p(y|x)$. Il est traité en mettant en œuvre des algorithmes de programmation dynamique.

3.2 Les CRF sur les séquences et sur les arbres

Les CRF ont pour l'instant surtout été utilisés pour annoter des séquences. Dans ce cas, le graphe utilisé est une "chaîne linéaire du premier ordre" dans laquelle chaque variable Y_v est reliée uniquement à sa voisine droite et à sa voisine gauche. Toutes les distributions de probabilités exprimables par un HMM peuvent être reproduites dans un CRF de cette forme (Sutton & McCallum, 2006). Plusieurs bibliothèques sont disponibles pour les mettre en œuvre : "crf.source.net" de Sarawagi, "Mallet" de McCallum et "CRF++" de Taku Kado.

Des travaux récents proposent d'adapter le modèle général des CRF au cas de l'annotation d'arbres dans lesquels chaque nœud interne peut avoir un nombre quelconque de fils ordonnés. Cette adaptation a été initialement définie pour annoter des documents XML. Elle a ainsi été employée avec succès dans des tâches d'extraction d'information à partir de pages Web, et de transformation de documents (Gilleron *et al.*, 2006a; Gilleron *et al.*, 2006b; Jousse, 2007). Elle peut aussi s'appliquer aux arbres présents dans le FTB.

Nos champs aléatoires X et Y sont donc désormais identifiés à l'ensemble des nœuds possibles d'un arbre. Nous devons, dans un premier temps, proposer un graphe pour connecter les variables de Y. Nous avons en fait envisagé 3 variantes possibles, de complexité croissante :

– dans la variante appelée 1-CRF, le graphe se réduit à un ensemble de singletons non connectés. L'annotation y_v d'un nœud quelconque v ne dépend alors que de sa position dans l'arbre, et de l'ensemble de l'arbre observé x. La tâche d'annotation par 1-CRF est donc ramenée à une tâche de *classification de nœuds*. Ce modèle, aussi connu sous le nom de *maximum d'entropie indépendant sur chacun des nœuds*, servira de baseline à nos expériences.
– dans la variante 2-CRF, nous prenons en compte les relations hiérarchiques spécifiques des arbres : nous relions entre eux dans le graphe chaque couple de nœuds de variables sur Y en relation *père-fils* dans l'arbre initial.
– enfin, dans la variante 3-CRF, nous tenons en plus compte de l'ordre des annotations portées par les fils successifs d'un même père. Le graphe relie donc alors à la fois les couples en relation *père-fils* et les couples en relation *frères successifs d'un même père*. Dans de tels graphes, les cliques maximales sont "triangulaires" : elles sont composées d'un père et de deux de ses fils consécutifs.

La Figure 2 montre les graphes sur les annotations correspondant à l'arbre droit de la Figure 1, pour un 2-CRF et un 3-CRF. La variante 2-CRF ramène en quelque sorte un arbre à l'ensemble des chemins allant de sa racine à chacune de ses feuilles : elle peut être simulée avec les outils mettant en œuvre les CRF sur les séquences. C'est l'approche adoptée par (Cohn & Blusom, 2005) pour la tâche de CoNLL. La variante 3-CRF, en revanche, est vraiment originale et seule l'application XCRF [4], issue des travaux précédemment évoqués, l'autorise. L'équipe qui

4. disponible librement sur : treecrf.gforge.inria.fr/

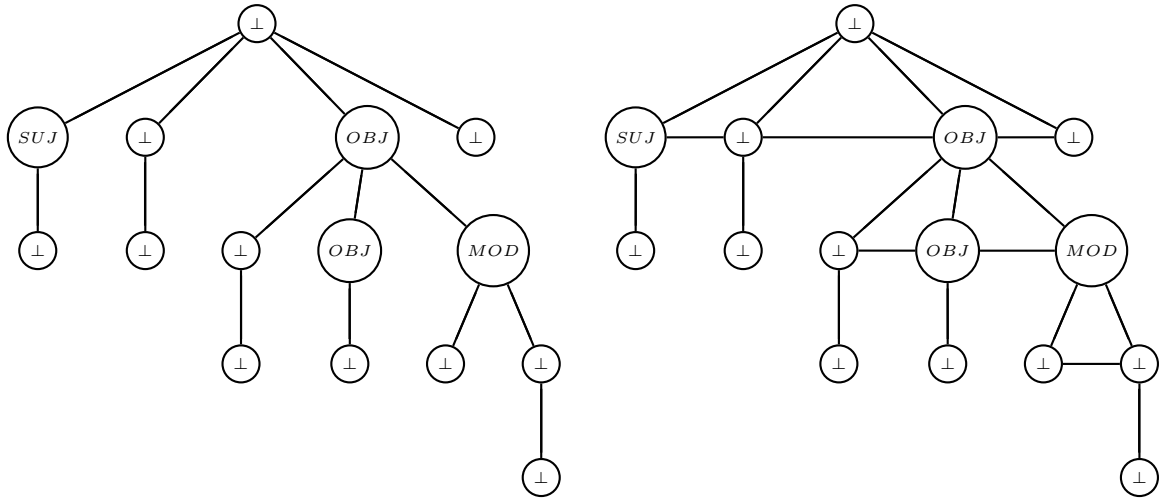

FIGURE 2 – graphes sur Y pour un 2-CRF (à gauche) et un 3-CRF (à droite)

l'a produite a dû redéfinir entièrement les algorithmes d'apprentissage et d'annotation adaptés à ce modèle "arboré". Elle a aussi montré que toutes les distributions de probabilités exprimables par une PCFG sous forme normale de Chomsky peuvent être reproduites par un 3-CRF (Gilleron *et al.*, 2006a; Jousse, 2007).

4 Expériences

4.1 Pré-traitements des données

Avant de lancer nos expériences, des prétraitements ont été nécessaires. En effet, alors que le corpus de CoNLL est centré sur les verbes, aucune "fonction syntaxique" spécifique n'est attachée aux verbes dans le FTB. Or, les fonctions syntaxiques des groupes nominaux dépendent essentiellement de leur position par rapport au verbe principal de la phrase : le sujet aura tendance à être à sa gauche, le ou les objet(s) à sa droite. Nous avons donc commencé par ajouter systématiquement des étiquettes PRED (pour "prédicat") à tous les nœuds VN qui n'ont pas déjà d'étiquette. Sans cet ajout, similaire à celui proposé dans (Schluter & van Genabith, 2008), il n'y aurait pas grand sens à prendre en compte des dépendances "horizontales" entre étiquettes de fonctions, comme c'est le cas dans un 3-CRF. De plus, parmi les VN déjà étiquetés, certains (par exemples des verbes à l'infinitifs), occupaient une fonction SUJ ou OBJ : ils n'ont pas été modifiés. Mais d'autres avaient pour étiquette la concaténation des fonctions des clitiques qui s'y rattachent et sont leurs fils dans l'arbre. Dans ce cas, nous avons remplacé cette annotation au niveau du VN par l'étiquette PREDC (pour "prédicat complexe") et nous avons automatiquement "fait descendre" les fonctions des clitiques aux fils concernés, en se fondant sur des informations présentes dans certains attributs de l'arbre initial. Certaines ambiguités résiduelles ont été traitées à la main. Après ce traitement, chaque nœud du corpus n'a au plus qu'une seule étiquette de fonction syntaxique parmi 11 possibles (les 8 initiales plus PRED, PREDC et $\perp$).

Nous avons compté dans le corpus ainsi enrichi 8 588 phrases contenant 439 370 nœuds parmi lesquels 62 390 ont une vraie étiquette de fonction syntaxique (cf. la table en section 4.3 pour voir leur répartition). Il y a 97 différents types de cliques "père-fils" annotées (correspondant

à 430 782 occurrences) et 474 différentes cliques “triangulaires” (correspondant à 261 098 occurrences). Ces nombres apparaissent comme suffisants pour espérer trouver des régularités dans les données, même si toutes les catégories ne sont bien sûr pas également représentées.

4.2 Sélection des features

Le choix de features pertinentes est essentiel à la construction d'un modèle fiable. La technique utilisée dans (Jousse, 2007) pour les arbres XML/HTML s'est avérée inadaptée pour le FTB : elle menait à une F1-mesure inférieure à 50%. Nous avons donc redéfini totalement le mode de sélection de ces features, sans pour autant faire appel à des ressources linguistiques externes. Dans nos expériences, chaque feature est caractérisée par une clique c et un couple (C, T) :

– C énumère l'ensemble y_c des valeurs de Y sur la clique c du graphe. Les features de type 1 (ou FT1) sont réduites à une seule valeur, elles correspondent aux 1-CRF. Les FT2 (resp. FT3) contiennent les annotations d'un couple de nœuds père-fils (resp. les triplets d'annotations d'une clique triangulaire) et correspondent aux 2-CRF (resp. 3-CRF). Par exemple, pour la clique triangulaire [5] identifiée par $c_0 = (3, 3.2, 3.3)$ dans l'arbre à droite de la Figure 1, on a une FT3 qui donne : $C_0 = \{OBJ, OBJ, MOD\}$

– $T = \{t_1, t_2, ..., t_n\}$ est un ensemble (éventuellement vide) de tests booléens portant sur les valeurs de l'observation x. Par exemple, pour l'arbre gauche de la Figure 1 où $x_{3.1}$ désigne le premier fils du troisième fils de la racine de l'observation, on peut définir $T_0 = \{x_{3.1} = VN\}$.

Ainsi, le couple (C_0, T_0) précédent caractérise la feature de la clique c_0 suivante :
$f_{(C_0,T_0)}(y_{c_0}, x, c_0) = 1\ si\ (y_3 = OBJ) \wedge (y_{3.2} = OBJ) \wedge (y_{3.3} = MOD) \wedge (x_{3.1} = VN)$
$f_{(C_0,T_0)}(y_{c_0}, x, c_0) = 0\ sinon$

En théorie, alors que les éléments de C sont restreints à une clique c de $\mathcal{G}$, ceux de T peuvent porter sur tout l'arbre observé x. Pour générer les features qui seront employées dans nos expériences, nous pouvons faire varier les paramètres suivants sur les tests de T :

– la nature de l'information disponible sur x prise en compte : nous nous sommes contentés ici d'utiliser les étiquettes syntaxiques et aucun autre attribut accessible dans l'arbre (comme les lemmes, les genres/nombres..., mais aussi le nombre de fils d'un nœud, sa profondeur, etc.)

– le voisinage autour de la position du nœud courant de la clique (en autorisant à aller dans toutes les directions : père, fils, frère gauche, frère droit) jusqu'où peuvent porter les tests. Dans notre exemple, le nœud courant qui identifie la clique c_0 est en position 3.2, et les tests sont limités à un voisinage de 1 (le nœud $x_{3.1}$ est bien à une distance 1 de ce nœud courant).

– le nombre maximal de tests autorisés dans un même ensemble T

Pour une clique et un FT donnés, un voisinage et un nombre de tests fixés, nous générons toutes les features représentées dans l'ensemble d'apprentissage. Pour limiter la combinatoire, nous nous restreignons à un voisinage de 2 et à un nombre maximal de tests égal à 2. Cependant, dans ce cas, nous avons réalisé les expériences en gardant ou non les features issues des configurations à occurrence unique. Les différences de performance sont minimes (jamais plus de 0.5%) mais ne pas tenir compte de ces features permet d'accélérer l'apprentissage.

4.3 Résultats et analyse

Dans les résultats qui suivent, la précision et le rappel ne sont calculés que pour les “vraies” étiquettes, c'est-à-dire sans tenir compte de $\perp$, trop fréquent : 85% des nœuds n'ont pas de

5. où le nœud numéroté $i.j$ est le j-ème fils du i-ème fils de la racine

fonction syntaxique et sont presque toujours correctement identifiés (99% de F1-mesure quel que soit le modèle utilisé). C'est la façon habituelle de mesurer la performance dans la tâche considérée (Blaheta, 2004). Pour nos expériences, nous avons découpé le corpus en 5 parties égales : 1/5 (c'est-à-dire 20%) est utilisé comme données d'apprentissage, pendant que les étiquettes de fonctions syntaxiques sont retirées des 4/5 (ou 80%) restant, utilisés comme corpus de test. Nous avons procédé à une validation croisée et fait la moyenne des expériences. Les tables qui suivent donnent un panorama des résultats obtenus : à gauche en faisant varier certains paramètres, à droite en détaillant les résultats du meilleur modèle sur chaque catégorie.

Config.	voisinage=1 Feat.	voisinage=1 F1	voisinage=2 Feat.	voisinage=2 F1
FT1 / 1T	103	44.3	508	78.8
FT1 / 2T	123	44.5	3,716*	79.14*
FT2 / 1T	579	52.9	2,358	81.3
FT2 / 2T	704	52.9	12,292*	80.4*
FT3 / 1T	1,766	79.1	6,570	81.6
FT3 / 2T	2,131	78.8	24,436*	80.3*

* : sans les features à occurrence unique
Feat. / F1 : nb de features / F1-mesure en %
FTn / mT : Feature type n avec m tests

étiquette	Prop.	P	R	F1
NO TAG	-	99.61	99.64	99.62
A-OBJ	2.75	20.22	8.72	11.88
ATO	0.24	55.66	23.08	28.81
ATS	3.72	73.86	49.96	59.57
DE-OBJ	2.56	36.27	16.87	22.95
MOD	23.27	71.31	81.47	75.96
OBJ	17.67	78.21	82.03	80.03
P-OBJ	1.53	13.62	5.72	6.12
PRED	26.27	95.72	97.54	96.62
PREDC	0.26	41.22	13.39	17.61
SUJ	21.72	88.70	91.07	89.87
Total	100.00	81.67	81.54	81.60

Le tableau de gauche montre qu'il vaut mieux augmenter les voisinages que le nombre de tests dans la génération des features. Celui de droite montre que les étiquettes les plus fréquentes (PRED, MOD, SUJ, OBJ) sont, sans surprise, plus faciles à retrouver que les autres. Comme on pouvait s'y attendre, on obtient de meilleures performances en utilisant des FT2 qu'en utilisant des FT1, et de meilleures encore avec des FT3, mais l'écart tend à diminuer quand le voisinage augmente. Il est difficile de comparer ces résultats avec d'autres, puisque ces expériences sont les premières menées sur le FTB. Pourtant, les meilleurs travaux portant sur le Penn Trebank (Blaheta & Charniak, 2000; Blaheta, 2004; Merlo & Musillo, 2005; Musillo & Merlo, 2005) donnent des valeurs comparables. Notons aussi que les baselines "de bon sens", faisant appel aux mêmes informations mais sans aucun apprentissage, que nous avons essayé de programmer directement (avec des règles du genre "le GN principal à gauche du verbe est SUJ") se comportaient nettement moins bien (F1-mesure inférieure à 60% pour SUJ, par exemple).

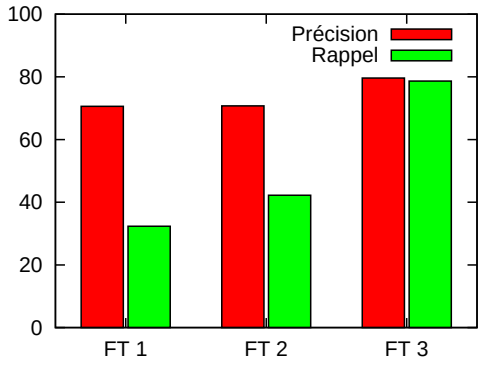

Fig. 3 : précision et rappel en fonction de FT cas d'un voisinage 1, avec 1 test.

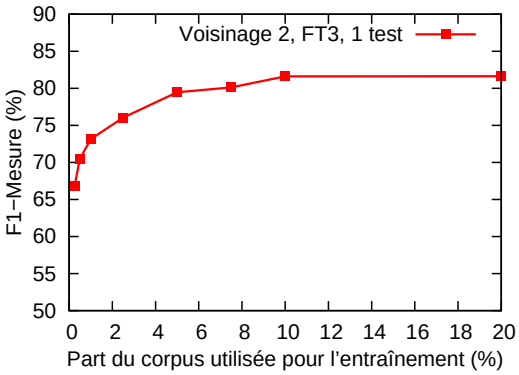

Fig. 4 : influence du nombre d'exemples proportion utilisé en apprentissage (%)

Les Figures 3 et 4 illustrent d'autres propriétés. On voit notamment sur la Figure 3 que le gain apporté par les modèles plus complexes concerne surtout le rappel. Un autre avantage, moins visible, des FT3 sur les FT2 (et les FT1) est qu'ils se comportent mieux sur les catégories moins représentées. Par exemple, DE-OBJ (2.5% des annotations), est reconnu avec seulement 14%/1% de précision/rappel avec un modèle de voisinage 1/FT1, mais avec 41%/13% de précision/rappel avec un modèle de voisinage 1/FT3. Avec la Figure 4, on a la confirmation que les CRF nécessitent peu d'exemples pour être performants : avec seulement 0.25% du corpus (21 phrases), on obtient déjà 66% de F1-mesure (validé sur des exemples nouveaux représentant 80% du corpus). Cette propriété est intéressante en termes d'efficacité : ainsi, avec 7.5% des données, l'apprentissage avec un 3-CRF prend environ 40mn et atteint 80% de F1-mesure.

Nous avons aussi réalisé quelques expériences qui montrent qu'en intégrant un minimum de connaissances linguistiques dans nos features, il est possible d'améliorer encore ces scores. Par exemple, nos modèles ont du mal à discriminer les étiquettes A-OBJ, DE-OBJ et P-OBJ, peu fréquentes et apparaissant dans des contextes très similaires, surtout quand on ne s'autorise à regarder que les catégories syntaxiques. Nous avons ainsi construit des tests *ad hoc* qui regardent si le *lemme* du premier fils du constituant est "de" ou "à" et nous avons généré toutes les features possibles avec les cliques présentes dans le corpus d'apprentissage et un de ces tests. Dans les mêmes conditions que celles de la table des résultats par catégorie, la précision et le rappel pour l'étiquette DE-OBJ (resp. A-OBJ) atteignent alors 63.9% et 64.1% (resp. 56.1% et 57.1%), soit un gain de plus 40%. La F1-mesure globale monte alors à 83.2%.

5 Conclusion et perspectives

Dans cet article, nous avons décrit le modèle général des CRF, et comment il peut s'instancier selon diverses variantes, en fonction du graphe que l'on se fixe entre annotations. Nous avons ensuite montré expérimentalement que les modèles les plus complexes permettent, comme on pouvait l'espérer, d'atteindre de meilleurs taux de reconnaissance sur la tâche que nous nous sommes fixée. Le modèle des 3-CRF, qui n'avait encore jamais été testé sur des données linguistiques, offre des perspectives intéressantes, même si son paramétrage reste encore sensible (notamment pour le mode de génération et de sélection des features). Nous avons aussi grâce à lui participé à la campagne CoNLL 2009, dans une tâche où il s'agissait d'affecter des rôles thématiques dans des arbres de dépendances provenant de corpus multilingues [6], avec des résultats très honorables (Moreau & Tellier, 2009).

Les CRF peuvent donc prendre directement en compte des structures arborées et ils nécessitent peu d'exemples pour apprendre. Ils sont aussi génériques : le programme d'apprentissage que nous avons employé (génération des features inclus) est totalement indépendant de la langue du corpus. D'un autre côté, les CRF sont aussi suffisamment flexibles pour intégrer facilement des connaissances ou des ressources linguistiques externes sous la forme de dictionnaires ou de règles, comme l'illustre l'exemple du traitement des étiquettes DE-OBJ et A-OBJ. Dans le même esprit nous comptons aussi, par exemple, traduire sous forme de features des schémas de sous-catégorisation de verbes, et évaluer leur apport. Les CRF pourraient ainsi contribuer à réconcilier apprentissage symbolique (via la génération de features) et apprentissage statistique [7].

6. http ://ufal.mff.cuni.cz/conll2009-st/
7. Ce travail résulte du projet ANR CRoTAL, CRF pour le TAL : http ://crotal.gforge.inria.fr/pmwiki-2.1.27/

Références

A. ABEILLÉ, Ed. (2003). *Treebanks, Building and Using Parsed Corpora*. Dordrecht/Boston/London : Kluwer Academic Publishers.

ALTUN Y., JOHNSON M. & HOFMANN T. (2003). Investigating loss functions and optimization methods for discriminative learning of label sequences. In *Proceedings of EMNLP*.

BLAHETA D. (2004). *Function Tagging*. PhD thesis, Brown University.

BLAHETA D. & CHARNIAK E. (2000). Assigning function tags to parsed text. In *Proceedings of NAACL-00*, p. 234–240.

X. CARRERAS & L. MARQUEZ, Eds. (2005). *actes de CoNNL 2005*.

COHN T. & BLUSOM P. (2005). Semantic role labelling with tree conditional random fields. In (Carreras & Marquez, 2005).

FINKEL J. R., KLEEMAN A. & MANNING C. D. (2008). Efficient, feature-based, conditional random field parsing. In *Proceedings of ACL-08 :HLT*, p. 959–967, Columbus, Ohio : ACL.

GILLERON R., JOUSSE F., TELLIER I. & TOMMASI M. (2006a). Conditional random fields for xml trees. In *ECML workshop on Mining and Learning in Graphs*.

GILLERON R., JOUSSE F., TELLIER I. & TOMMASI M. (2006b). Xml document transformation with conditional random fields,. In S. L. 4518, Ed., *INEX 2006*.

HAMMERSLEY J. & CLIFFORD P. (1971). Markov fields on finite graphs and lattices. Unpublished.

JOUSSE F. (2007). *Transformations d'Arbres XML avec des Modèles Probabilistes pour l'Annotation*. PhD thesis, Université Charles de Gaulle - Lille 3.

LAFFERTY J., MCCALLUM A. & PEREIRA F. (2001). Conditional random fields : Probabilistic models for segmenting and labeling sequence data. In *Proceedings of ICML'01*, p. 282–289.

MARCUS M. (1993). Building a large annotated corpus : the penn treebank. In *Computational Linguistics*, p. 313–330.

MCCALLUM A. & LI W. (2003). Early results for named entity recognition with conditional random fields. In *Proceedings of CoNLL 2003*.

MERLO P. & MUSILLO G. (2005). Accurate function parsing. In *proceedings of HLT '05*, p. 620–627 : ACL.

MOREAU E. & TELLIER I. (2009). The crotal srl system : a generic tool based on tree-structured crf. In *proceedings of CoNNL 2009*.

MUSILLO G. & MERLO P. (2005). Lexical and structural biases for function parsing. In *Proceedings of IWPT 2005*, p. 83–93.

PINTO D., MCCALLUM A., LEE X. & CROFT W. (2003). Table extraction using conditional random fields. In *SIGIR'03 : Proceedings of the 26th ACM SIGIR*.

SCHLUTER N. & VAN GENABITH J. (2008). Treebank-based acquisition of lfg parsing resources for french. In *Proceedings of LREC 08*, Marrakech, Morocco.

SHA F. & PEREIRA F. (2003). Shallow parsing with conditional random fields. In *Technical Report CIS TR MS-CIS-02-35, University of Pennsylvania, 2003.*

SUTTON C. & MCCALLUM A. (2006). *An Introduction to Conditional Random Fields for Relational Learning*, In L. GETOOR & B. TASKAR, Eds., *Introduction to Statistical Relational Learning*. MIT Press.