Tree Representations in Probabilistic Models for Extended Named Entities Detection

Marco Dinarelli
LIMSI-CNRS
Orsay, France
marcod@limsi.fr

Sophie Rosset
LIMSI-CNRS
Orsay, France
rosset@limsi.fr

Abstract

In this paper we deal with Named Entity Recognition (NER) on transcriptions of French broadcast data. Two aspects make the task more difficult with respect to previous NER tasks: i) named entities annotated used in this work have a tree structure, thus the task cannot be tackled as a sequence labelling task; ii) the data used are more noisy than data used for previous NER tasks. We approach the task in two steps, involving Conditional Random Fields and Probabilistic Context-Free Grammars, integrated in a single parsing algorithm. We analyse the effect of using several tree representations. Our system outperforms the best system of the evaluation campaign by a significant margin.

1 Introduction

Named Entity Recognition is a traditinal task of the Natural Language Processing domain. The task aims at mapping words in a text into semantic classes, such like persons, organizations or localizations. While at first the NER task was quite simple, involving a limited number of classes (Grishman and Sundheim, 1996), along the years the task complexity increased as more complex class taxonomies were defined (Sekine and Nobata, 2004). The interest in the task is related to its use in complex frameworks for (semantic) content extraction, such like Relation Extraction applications (Doddington et al., 2004).

This work presents research on a Named Entity Recognition task defined with a new set of named entities. The characteristic of such set is in that named entities have a tree structure. As concequence the task cannot be tackled as a sequence labelling approach. Additionally, the use of noisy data like transcriptions of French broadcast data, makes the task very challenging for traditional NLP solutions. To deal with such problems, we adopt a two-steps approach, the first being realized with Conditional Random Fields (CRF) (Lafferty et al., 2001), the second with a Probabilistic Context-Free Grammar (PCFG) (Johnson, 1998). The motivations behind that are:

- Since the named entities have a tree structure, it is reasonable to use a solution coming from syntactic parsing. However preliminary experiments using such approaches gave poor results.

- Despite the tree-structure of the entities, trees are not as complex as syntactic trees, thus, before designing an ad-hoc solution for the task, which require a remarkable effort and yet it doesn't guarantee better performances, we designed a solution providing good results and which required a limited development effort.

- Conditional Random Fields are models robust to noisy data, like automatic transcriptions of ASR systems (Hahn et al., 2010), thus it is the best choice to deal with transcriptions of broadcast data. Once words have been annotated with basic entity constituents, the tree structure of named entities is simple enough to be reconstructed with relatively simple model like PCFG (Johnson, 1998).

The two models are integrated in a single parsing algorithm. We analyze the effect of the use of

Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics, pages 174–184,
Avignon, France, April 23 - 27 2012.

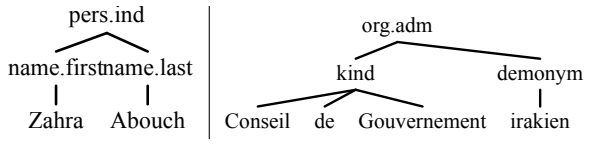

Figure 1: Examples of structured named entities annotated on the data used in this work

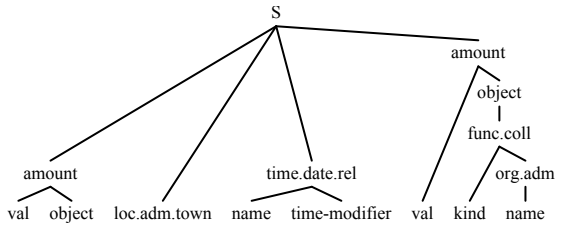

Figure 2: An example of named entity tree corresponding to entities of a whole sentence. Tree leaves, corresponding to sentence words have been removed to keep readability

Quaero	training		dev	
# sentences	43,251		112	
	words	entities	words	entities
# tokens	1,251,432	245,880	2,659	570
# vocabulary	39,631	134	891	30
# components	–	133662	–	971
# components dict.	–	28	–	18
# OOV rate [%]	–	–	17.15	0

Table 1: Statistics on the training and development sets of the Quaero corpus

several tree representations, which result in different parsing models with different performances. We provide a detailed evaluation of our models. Results can be compared with those obtained in the evaluation campaign where the same data were used. Our system outperforms the best system of the evaluation campaign by a significant margin.

The rest of the paper is structured as follows: in the next section we introduce the extended named entities used in this work, in section 3 we describe our two-steps algorithm for parsing entity trees, in section 4 we detail the second step of our approach based on syntactic parsing approaches, in particular we describe the different tree representations used in this work to encode entity trees in parsing models. In section 6 we describe and comment experiments, and finally, in section 7, we draw some conclusions.

2 Extended Named Entities

The most important aspect of the NER task we investigated is provided by the tree structure of named entities. Examples of such entities are given in figure 1 and 2, where words have been remove for readability issues and are: (*"90 persons are still present at Atambua. It's there that 3 employees of the High Conseil of United Nations for refugees have been killed yesterday morning"*):

> ***90 personnes*** *toujours présentes* ***à Atambua*** *c' est là qu'* ***hier matin*** *ont été tués* ***3 employés du haut commissariat des Nations unies*** *aux réfugiés , le* ***HCR***

Words realizing entities in figure 2 are in bold, and they correspond to the tree leaves in the picture. As we see in the figures, entities can have complex structures. Beyond the use of subtypes, like *individual* in *person* (to give `pers.ind`), or *administrative* in *organization* (to give *org.adm*), entities with more specific content can be constituents of more general entities to form tree structures, like *name.first* and *name.last* for *pers.ind* or *val* (for value) and *object* for *amount*.

These named entities have been annotated on transcriptions of French broadcast news coming from several radio channels. The transcriptions constitute a corpus that has been split into training, development and evaluation sets.The evaluation set, in particular, is composed of two set of data, Broadcast News (BN in the table) and Broadcast Conversations (BC in the table). The evaluation of the models presented in this work is performed on the merge of the two data types. Some statistics of the corpus are reported in table 1 and 2. This set of named entities has been defined in order to provide more fine semantic information for entities found in the data, e.g. a person is better specified by first and last name, and is fully described in (Grouin, 2011) . In order to avoid confusion, entities that can be associated directly to words, like *name.first*, *name.last*, *val* and *object*, are called entity constituents, components or entity pre-terminals (as they are preterminals nodes in the trees). The other entities, like *pers.ind* or *amount*, are called entities or nonterminal entities, depending on the context.

3 Models Cascade for Extended Named Entities

Since the task of Named Entity Recognition presented here cannot be modeled as sequence labelling and, as mentioned previously, an approach

Quaero	test BN		test BC	
# sentences	1704		3933	
	words	entities	words	entities
# tokens	32945	2762	69414	2769
# vocabulary		28		28
# components	–	4128	–	4017
# components dict.	–	21	–	20
# OOV rate [%]	3.63	0	3.84	0

Table 2: Statistics on the test set of the Quaero corpus, divided in Broadcast News (BN) and Broadcast Conversations (BC)

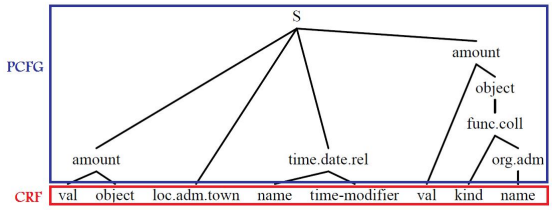

Figure 3: Processing schema of the two-steps approach proposed in this work: CRF plus PCFG

coming from syntactic parsing to perform named entity annotation in "one-shot" is not robust on the data used in this work, we adopt a two-steps. The first is designed to be robust to noisy data and is used to annotate entity components, while the second is used to parse complete entity trees and is based on a relatively simple model. Since we are dealing with noisy data, the hardest part of the task is indeed to annotate components on words. On the other hand, since entity trees are relatively simple, at least much simpler than syntactic trees, once entity components have been annotated in a first step, for the second step, a complex model is not required, which would also make the processing slower. Taking all these issues into account, the two steps of our system for tree-structured named entity recognition are performed as follows:

1. A CRF model (Lafferty et al., 2001) is used to annotate components on words.
2. A PCFG model (Johnson, 1998) is used to parse complete entity trees upon components, i.e. using components annotated by CRF as starting point.

This processing schema is depicted in figure 3. Conditional Random Fields are described shortly in the next subsection. PCFG models, constituting the main part of this work together with the analysis over tree representations, is described more in details in the next sections.

3.1 Conditional Random Fields

CRFs are particularly suitable for sequence labelling tasks (Lafferty et al., 2001). Beyond the possibility to include a huge number of features using the same framework as Maximum Entropy models (Berger et al., 1996), CRF models encode global conditional probabilities normalized at sentence level.

Given a sequence of N words $W_1^N = w_1, ..., w_N$ and its corresponding components sequence $E_1^N = e_1, ..., e_N$, CRF trains the conditional probabilities

$$P(E_1^N|W_1^N) =$$

$$\frac{1}{Z}\prod_{n=1}^{N}\exp\left(\sum_{m=1}^{M}\lambda_m \cdot h_m(e_{n-1}, e_n, w_{n-2}^{n+2})\right) \quad (1)$$

where λ_m are the training parameters. $h_m(e_{n-1}, e_n, w_{n-2}^{n+2})$ are the feature functions capturing dependencies of entities and words. Z is the partition function:

$$Z = \sum_{\tilde{e}_1^N}\prod_{n=1}^{N} H(\tilde{e}_{n-1}, \tilde{e}_n, w_{n-2}^{n+2}) \quad (2)$$

which ensures that probabilities sum up to one. $\tilde{e}_{n-1}$ and $\tilde{e}_n$ are components for previous and current words, $H(\tilde{e}_{n-1}, \tilde{e}_n, w_{n-2}^{n+2})$ is an abbreviation for $\sum_{m=1}^{M}\lambda_m \cdot h_m(e_{n-1}, e_n, w_{n-2}^{n+2})$, i.e. the set of active feature functions at current position in the sequence.

In the last few years different CRF implementations have been realized. The implementation we refer in this work is the one described in (Lavergne et al., 2010), which optimize the following objective function:

$$-log(P(E_1^N|W_1^N)) + \rho_1\|\lambda\|_1 + \frac{\rho_2}{2}\|\lambda\|_2^2 \quad (3)$$

$\|\lambda\|_1$ and $\|\lambda\|_2^2$ are the *l1* and *l2* regularizers (Riezler and Vasserman, 2004), and together in a linear combination implement the elastic net regularizer (Zou and Hastie, 2005). As mentioned in (Lavergne et al., 2010), this kind of regularizers are very effective for feature selection at training time, which is a very good point when dealing with noisy data and big set of features.

4 Models for Parsing Trees

The models used in this work for parsing entity trees refer to the models described in (Johnson, 1998), in (Charniak, 1997; Caraballo and Charniak, 1997) and (Charniak et al., 1998), and which constitutes the basis of the maximum entropy model for parsing described in (Charniak, 2000). A similar lexicalized model has been proposed also by Collins (Collins, 1997). All these models are based on a PCFG trained from data and used in a chart parsing algorithm to find the best parse for the given input. The PCFG model of (Johnson, 1998) is made of rules of the form:

- $X_i \Rightarrow X_j X_k$
- $X_i \Rightarrow w$

where X are non-terminal entities and w are terminal symbols (words in our case).[1] The probability associated to these rules are:

$$p_{i \to j,k} = \frac{P(X_i \Rightarrow X_j, X_k)}{P(X_i)} \quad (4)$$

$$p_{i \to w} = \frac{P(X_i \Rightarrow w)}{P(X_i)} \quad (5)$$

The models described in (Charniak, 1997; Caraballo and Charniak, 1997) encode probabilities involving more information, such as head words. In order to have a PCFG model made of rules with their associated probabilities, we extract rules from the entity trees of our corpus. This processing is straightforward, for example from the tree depicted in figure 2, the following rules are extracted:

S ⇒ amount loc.adm.town time.dat.rel amount
amount ⇒ val object
time.date.rel ⇒ name time-modifier
object ⇒ func.coll
func.coll ⇒ kind org.adm
org.adm ⇒ name

Using counts of these rules we then compute maximum likelihood probabilities of the Right Hand Side (RHS) of the rule given its Left Hand Side (LHS). Also binarization of rules, applied to have all rules in the form of 4 and 5, is straightforward and can be done with simple algorithms not discussed here.

[1] These rules are actually in Chomsky Normal Form, i.e. unary or binary rules only. A PCFG, in general, can have any rule, however, the algorithm we are discussing convert the PCFG rules into Chomsky Normal Form, thus for simplicity we provide directly such formulation.

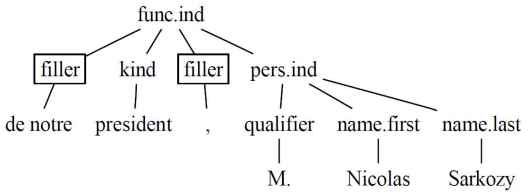

Figure 4: Baseline tree representations used in the PCFG parsing model

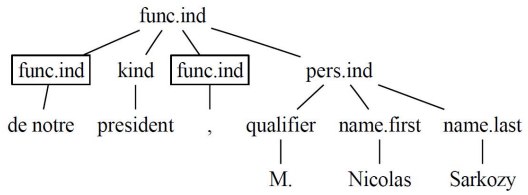

Figure 5: Filler-parent tree representations used in the PCFG parsing model

4.1 Tree Representations for Extended Named Entities

As discussed in (Johnson, 1998), an important point for a parsing algorithm is the representation of trees being parsed. Changing the tree representation can change significantly the performances of the parser. Since there is a large difference between entity trees used in this work and syntactic trees, from both meaning and structure point of view, it is worth performing an analysis with the aim of finding the most suitable representation for our task. In order to perform this analysis, we start from a named entity annotated on the words *de notre president , M. Nicolas Sarkozy*(*of our president, Mr. Nicolas Sarkozy*). The corresponding named entity is shown in figure 4. As decided in the annotation guidelines, fillers can be part of a named entity. This can happen for complex named entities involving several words. The representation shown in figure 4 is the default representation and will be referred to as *baseline*. A problem created by this representation is the fact that fillers are present also outside entities. Fillers of named entities should be, in principle, distinguished from any other filler, since they may be informative to discriminate entities.

Following this intuition, we designed two different representations where entity fillers are con-

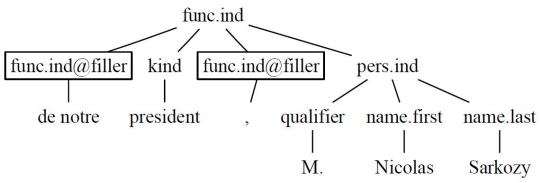

Figure 6: Parent-context tree representations used in the PCFG parsing model

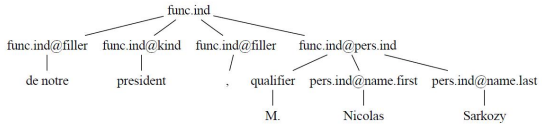

Figure 7: Parent-node tree representations used in the PCFG parsing model

textualized so that to be distinguished from the other fillers. In the first representation we give to the filler the same label of the parent node, while in the second representation we use a concatenation of the filler and the label of the parent node. These two representations are shown in figure 5 and 6, respectively. The first one will be referred to as *filler-parent*, while the second will be referred as *parent-context*. A problem that may be introduced by the first representation is that some entities that originally were used only for non-terminal entities will appear also as components, i.e. entities annotated on words. This may introduce some ambiguity.

Another possible contextualization can be to annotate each node with the label of the parent node. This representation is shown in figure 7 and will be referred to as *parent-node*. Intuitively, this representation is effective since entities annotated directly on words provide also the entity of the parent node. However this representation increases drastically the number of entities, in particular the number of components, which in our case are the set of labels to be learned by the CRF model. For the same reason this representation produces more rigid models, since label sequences vary widely and thus is not likely to match sequences not seen in the training data.

Finally, another interesting tree representation is a variation of the *parent-node* tree, where entity fillers are only distinguished from fillers not in an entity, using the label *ne-filler*, but they are not contextualized with entity information. This representation is shown in figure 8 and it will be

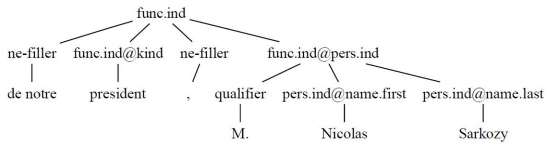

Figure 8: Parent-node-filler tree representations used in the PCFG parsing model

referred to as *parent-node-filler*. This representation is a good trade-off between contextual information and rigidity, by still representing entities as concatenation of labels, while using a common special label for entity fillers. This allows to keep lower the number of entities annotated on words, i.e. components.

Using different tree representations affects both the structure and the performance of the parsing model. The structure is described in the next section, the performance in the evaluation section.

4.2 Structure of the Model

Lexicalized models for syntactic parsing described in (Charniak, 2000; Charniak et al., 1998) and (Collins, 1997), integrate more information than what is used in equations 4 and 5. Considering a particular node in the entity tree, not including terminals, the information used is:

- s: the head word of the node, i.e. the most important word of the chunk covered by the current node
- h: the head word of the parent node
- t: the entity tag of the current node
- l: the entity tag of the parent node

The head word of the parent node is defined percolating head words from children nodes to parent nodes, giving the priority to verbs. They can be found using automatic approaches based on words and entity tag co-occurrence or mutual information. Using this information, the model described in (Charniak et al., 1998) is $P(s|h,t,l)$. This model being conditioned on several pieces of information, it can be affected by data sparsity problems. Thus, the model is actually approximated as an interpolation of probabilities:

$$P(s|h,t,l) = \lambda_1 P(s|h,t,l) + \lambda_2 P(s|c_h,t,l) + \lambda_3 P(s|t,l) + \lambda_4 P(s|t) \quad (6)$$

where $\lambda_i, i = 1, ..., 4$, are parameters of the model to be tuned, and c_h is the cluster of head words for a given entity tag t. With such model, when not all pieces of information are available to estimate reliably the probability with more conditioning, the model can still provide a probability with terms conditioned with less information. The use of head words and their percolation over the tree is called lexicalization. The goal of tree lexicalization is to add lexical information all over the tree. This way the probability of all rules can be conditioned also on lexical information, allowing to define the probabilities $P(s|h,t,l)$ and $P(s|c_h,t,l)$. Tree lexicalization reflects the characteristics of syntactic parsing, for which the models described in (Charniak, 2000; Charniak et al., 1998) and (Collins, 1997) were defined. Head words are very informative since they constitute keywords instantiating labels, regardless if they are syntactic constituents or named entities. However, for named entity recognition it doesn't make sense to give priority to verbs when percolating head words over the tree, even more because head words of named entities are most of the time nouns. Moreover, it doesn't make sense to give priority to the head word of a particular entity with respect to the others, all entities in a sentence have the same importance. Intuitively, lexicalization of entity trees is not straightforward as lexicalization of syntactic trees. At the same time, using not lexicalized trees doesn't make sense with models like 6, since all the terms involve lexical information. Instead, we can use the model of (Johnson, 1998), which define the probability of a tree τ as:

$$P(\tau) = \prod_{X \to \alpha} P(X \to \alpha)^{C_\tau(X \to \alpha)} \quad (7)$$

here the RHS of rules has been generalized with α, representing RHS of both unary and binary rules 4 and 5. $C_\tau(X \to \alpha)$ is the number of times the rule $X \to \alpha$ appears in the tree τ. The model 7 is instantiated when using tree representations shown in Fig. 4, 5 and 6. When using representations given in Fig. 7 and 8, the model is:

$$P(\tau|l) \quad (8)$$

where l is the entity label of the parent node. Although non-lexicalized models like 7 and 8 have shown less effective for syntactic parsing than their lexicalized couter-parts, there are evidences showing that they can be effective in our task. With reference to figure 4, considering the entity *pers.ind* instantiated by *Nicolas Sarkozy*, our algorithm detects first *name.first* for *Nicolas* and *name.last* for *Sarkozy* using the CRF model. As mentioned earlier, once the CRF model has detected components, since entity trees have not a complex structure with respect to syntactic trees, even a simple model like the one in equation 7 or 8 is effective for entity tree parsing. For example, once *name.first* and *name.last* have been detected by CRF, *pers.ind* is the only entity having *name.first* and *name.last* as children. Ambiguities, like for example for *kind* or *qualifier*, which can appear in many entities, can affect the model 7, but they are overcome by the model 8, taking the entity tag of the parent node into account. Moreover, the use of CRF allows to include in the model much more features than the lexicalized model in equation 6. Using features like word prefixes (P), suffixes (S), capitalization (C), morpho-syntactic features (MS) and other features indicated as F[2], the CRF model encodes the conditional probability:

$$P(t|w, P, S, C, MS, F) \quad (9)$$

where w is an input word and t is the corresponding component.

The probability of the CRF model, used in the first step to tag input words with components, is combined with the probability of the PCFG model, used to parse entity trees starting from components. Thus the structure of our model is:

$$P(t|w, P, S, C, MS, F) \cdot P(\tau) \quad (10)$$

or

$$P(t|w, P, S, C, MS, F) \cdot P(\tau|l) \quad (11)$$

depending if we are using the tree representation given in figure 4, 5 and 6 or in figure 7 and 8, respectively. A scale factor could be used to combine the two scores, but this is optional as CRFs can provide normalized posterior probabilities.

[2]The set of features used in the CRF model will be described in more details in the evaluation section.

5 Related Work

While the models used for named entity detection and the set of named entities defined along the years have been discussed in the introduction and in section 2, since CRFs and models for parsing constitute the main issue in our work, we discuss some important models here.

Beyond the models for parsing discussed in section 4, together with motivations for using or not in our work, another important model for syntactic parsing has been proposed in (Ratnaparkhi, 1999). Such model is made of four Maximum Entropy models used in cascade for parsing at different stages. Also this model makes use of head words, like those described in section 4, thus the same considerations hold, moreover it seems quite complex for real applications, as it involves the use of four different models together. The models described in (Johnson, 1998), (Charniak, 1997; Caraballo and Charniak, 1997), (Charniak et al., 1998), (Charniak, 2000), (Collins, 1997) and (Ratnaparkhi, 1999), constitute the main individual models proposed for constituent-based syntactic parsing. Later other approaches based on models combination have been proposed, like e.g. the reranking approach described in (Collins and Koo, 2005), among many, and also evolutions or improvements of these models.

More recently, approaches based on log-linear models have been proposed (Clark and Curran, 2007; Finkel et al., 2008) for parsing, called also "*Tree CRF*", using also different training criteria (Auli and Lopez, 2011). Using such models in our work has basically two problems: one related to scaling issues, since our data present a large number of labels, which makes CRF training problematic, even more when using "*Tree CRF*"; another problem is related to the difference between syntactic parsing and named entity detection tasks, as mentioned in sub-section 4.2. Adapting "*Tree CRF*" to our task is thus a quite complex work, it constitutes an entire work by itself, we leave it as feature work.

Concerning linear-chain CRF models, the one we use is a state-of-the-art implementation (Lavergne et al., 2010), as it implements the most effective optimization algorithms as well as state-of-the-art regularizers (see sub-section 3.1). Some improvement of linear-chain CRF have been proposed, trying to integrate higher order target-side features (Tang et al., 2006). An integration of the same kind of features has been tried also in the model used in this work, without giving significant improvements, but making model training much harder. Thus, this direction has not been further investigated.

6 Evaluation

In this section we describe experiments performed to evaluate our models. We first describe the settings used for the two models involved in the entity tree parsing, and then describe and comment the results obtained on the test corpus.

6.1 Settings

The CRF implementation used in this work is described in (Lavergne et al., 2010), named *wapiti*.[3] We didn't optimize parameters ρ_1 and ρ_2 of the elastic net (see section 3.1), although this improves significantly the performances and leads to more compact models, default values lead in most cases to very accurate models. We used a wide set of features in CRF models, in a window of $[-2, +2]$ around the target word:

- A set of standard features like word prefixes and suffixes of length from 1 to 6, plus some *Yes/No* features like *Does the word start with capital letter?*, etc.
- Morpho-syntactic features extracted from the output of the tool *tagger* (Allauzen and Bonneau-Maynard, 2008)
- Features extracted from the output of the semantic analyzer (Rosset et al., (2009)) provided by the tool *WMatch* (Galibert, 2009).

This analysis morpho-syntactic information as well as semantic information at the same level of named entities. Using two different sets of morpho-syntactic features results in more effective models, as they create a kind of agreement for a given word in case of match. Concerning the PCFG model, grammars, tree binarization and the different tree representations are created with our own scripts, while entity tree parsing is performed with the chart parsing algorithm described in (Johnson, 1998).[4]

[3] available at `http://wapiti.limsi.fr`

[4] available at `http://web.science.mq.edu.au/~mjohnson/Software.htm`

	CRF		PCFG
Model	**# features**	**# labels**	**# rules**
baseline	3,041,797	55	29,611
filler-parent	3,637,990	112	29,611
parent-context	3,605,019	120	29,611
parent-node	3,718,089	441	31,110
parent-node-filler	3,723,964	378	31,110

Table 3: Statistics showing the characteristics of the different models used in this work

	DEV		TEST	
Model	**SER**	**F1**	**SER**	**F1**
baseline	20.0%	73.4%	14.2%	79.4%
filler-parent	16.2%	77.8%	12.5%	81.2%
parent-context	15.2%	78.6%	11.9%	81.4%
parent-node	6.6%	96.7%	5.9%	96.7%
parent-node-filler	6.8%	95.9%	5.7%	96.8%

Table 4: Results computed from oracle predictions obtained with the different models presented in this work

	DEV		TEST	
Model	**SER**	**F1**	**SER**	**F1**
baseline	33.5%	72.5%	33.4%	72.8%
filler-parent	31.3%	74.4%	33.4%	72.7%
parent-context	30.9%	74.6%	33.3%	72.8%
parent-node	31.2%	77.8%	31.4%	79.5%
parent-node-filler	28.7%	78.9%	30.2%	80.3%

Table 5: Results obtained with our combined algorithm based on CRF and PCFG

6.2 Evaluation Metrics

All results are expressed in terms of Slot Error Rate (SER) (Makhoul et al., 1999) which has a similar definition of word error rate for ASR systems, with the difference that substitution errors are split in three types: i) correct entity type with wrong segmentation; ii) wrong entity type with correct segmentation; iii) wrong entity type with wrong segmentation; here, i) and ii) are given half points, while iii), as well as insertion and deletion errors, are given full points. Moreover, results are given using the well known $F1$ measure, defined as a function of precision and recall.

6.3 Results

In this section we provide evaluations of the models described in this work, based on combination of CRF and PCFG and using different tree representations of named entity trees.

6.3.1 Model Statistics

As a first evaluation, we describe some statistics computed from the CRF and PCFG models using the tree representations. Such statistics provide interesting clues of how difficult is learning the task and which performance we can expect from the model. Statistics for this evaluation are presented in table 3. Rows corresponds to the different tree representations described in this work, while in the columns we show the number of features and labels for the CRF models (**# features** and **# labels**), and the number of rules for PCFG models (# rules).

As we can see from the table, the number of rules is the same for the tree representations **baseline**, **filler-parent** and **parent-context**, and for the representations **parent-node** and **parent-node-filler**. This is the consequence of the contextualization applied by the latter representations, i.e. **parent-node** and **parent-node-filler** create several different labels depending from the context, thus the corresponding grammar will have more rules. For example, the rule `pers.ind` ⇒ `name.first name.last` can appear as it is or contextualized with `func.ind`, like in figure 8. In contrast the other tree representations modify only fillers, thus the number of rules is not affected.

Concerning CRF models, as shown in table 3, the use of the different tree representations results in an increasing number of labels to be learned by CRF. This aspect is quite critical in CRF learning, as training time is exponential in the number of labels. Indeed, the most complex models, obtained with **parent-node** and **parent-node-filler** tree representations, took roughly 8 days for training. Additionally, increasing the number of labels can create data sparseness problems, however this problem doesn't seem to arise in our case since, apart the **baseline** model which has quite less features, all the others have approximately the same number of features, meaning that there are actually enough data to learn the models, regardless the number of labels.

6.3.2 Evaluations of Tree Representations

In this section we evaluate the models in terms of the evaluation metrics described in previous section, Slot Error Rate (SER) and F1 measure.

In order to evaluate PCFG models alone, we performed entity tree parsing using as input reference transcriptions, i.e. manual transcriptions and reference component annotations taken from development and test sets. This can be considered a kind of oracle evaluations and provides us an upper bound of the performance of the PCFG models. Results for this evaluation are reported in

Participant	SER
P1	48.9
P2	41.0
parent-context	33.3
parent-node	31.4
parent-node-filler	30.2

Table 6: Results obtained with our combined algorithm based on CRF and PCFG

table 4. As it can be intuitively expected, adding more contextualization in the trees results in more accurate models, the simplest model, **baseline**, has the worst oracle performance, **filler-parent** and **parent-context** models, adding similar contextualization information, have very similar oracle performances. Same line of reasoning applies to models **parent-node** and **parent-node-filler**, which also add similar contextualization and have very similar oracle predictions. These last two models have also the best absolute oracle performances. However, adding more contextualization in the trees results also in more rigid models, the fact that models are robust on reference transcriptions and based on reference component annotations, doesn't imply a proportional robustness on component sequences generated by CRF models.

This intuition is confirmed from results reported in table 5, where a real evaluation of our models is reported, using this time CRF output components as input to PCFG models, to parse entity trees. The results reported in table 5 show in particular that models using **baseline**, **filler-parent** and **parent-context** tree representations have similar performances, especially on test set. Models characterized by **parent-node** and **parent-node-filler** tree representations have indeed the best performances, although the gain with respect to the other models is not as much as it could be expected given the difference in the oracle performances discussed above. In particular the best absolute performance is obtained with the model **parent-node-filler**. As we mentioned in subsection 4.1, this model represents the best trade-off between rigidity and accuracy using the same label for all entity fillers, but still distinguishing between fillers found in entity structures and other fillers found in words not instantiating any entity.

6.3.3 Comparison with Official Results

As a final evaluation of our models, we provide a comparison of official results obtained at the 2011 evaluation campaign of extended named entity recognition (Galibert et al., 2011; 2) Results are reported in table 6, where the other two participants to the campaign are indicated as $P1$ and $P2$. These two participants *P1* and *P2*, used a system based on CRF, and rules for deep syntactic analysis, respectively. In particular, $P2$ obtained superior performances in previous evaluation campaign on named entity recognition. The system we proposed at the evaluation campaign used a **parent-context** tree representation. The results obtained at the evaluation campaign are in the first three lines of Table 6. We compare such results with those obtained with the **parent-node** and **parent-node-filler** tree representations, reported in the last two rows of the same table. As we can see, the new tree representations described in this work allow to achieve the best absolute performances.

7 Conclusions

In this paper we have presented a Named Entity Recognition system dealing with extended named entities with a tree structure. Given such representation of named entities, the task cannot be modeled as a sequence labelling approach. We thus proposed a two-steps system based on CRF and PCFG. CRF annotate entity components directly on words, while PCFG apply parsing techniques to predict the whole entity tree. We motivated our choice by showing that it is not effective to apply techniques used widely for syntactic parsing, like for example tree lexicalization. We presented an analysis of different tree representations for PCFG, which affect significantly parsing performances.

We provided and discussed a detailed evaluation of all the models obtained by combining CRF and PCFG with the different tree representation proposed. Our combined models result in better performances with respect to other models proposed at the official evaluation campaign, as well as our previous model used also at the evaluation campaign.

Acknowledgments

This work has been funded by the project Quaero, under the program Oseo, French State agency for innovation.

References

Ralph Grishman and Beth Sundheim. 1996. Message Understanding Conference-6: a brief history. In *Proceedings of the 16th conference on Computational linguistics - Volume 1*, pages 466–471, Stroudsburg, PA, USA. Association for Computational Linguistics.

Satoshi Sekine and Chikashi Nobata. 2004. Definition, Dictionaries and Tagger for Extended Named Entity Hierarchy. In *Proceedings of LREC*.

G. Doddington, A. Mitchell, M. Przybocki, L. Ramshaw, S. Strassel, and R. Weischedel. 2004. The Automatic Content Extraction (ACE) Program–Tasks, Data, and Evaluation. *Proceedings of LREC 2004*, pages 837–840.

Cyril Grouin, Sophie Rosset, Pierre Zweigenbaum, Karn Fort, Olivier Galibert, Ludovic Quintard. 2011. Proposal for an extension or traditional named entities: From guidelines to evaluation, an overview. In *Proceedings of the Linguistic Annotation Workshop (LAW)*.

J. Lafferty, A. McCallum, and F. Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of the Eighteenth International Conference on Machine Learning (ICML)*, pages 282–289, Williamstown, MA, USA, June.

Mark Johnson. 1998. Pcfg models of linguistic tree representations. *Computational Linguistics*, 24:613–632.

Stefan Hahn, Marco Dinarelli, Christian Raymond, Fabrice Lefèvre, Patrick Lehen, Renato De Mori, Alessandro Moschitti, Hermann Ney, and Giuseppe Riccardi. 2010. Comparing stochastic approaches to spoken language understanding in multiple languages. *IEEE Transactions on Audio, Speech and Language Processing (TASLP)*, 99.

Adam L. Berger, Stephen A. Della Pietra, and Vincent J. Della Pietra. 1996. A maximum entropy approach to natural language processing. *COMPUTATIONAL LINGUISTICS*, 22:39–71.

Thomas Lavergne, Olivier Cappé, and François Yvon. 2010. Practical very large scale CRFs. In *Proceedings the 48th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 504–513. Association for Computational Linguistics, July.

Stefan Riezler and Alexander Vasserman. 2004. Incremental feature selection and l1 regularization for relaxed maximum-entropy modeling. In *Proceedings of the International Conference on Empirical Methods for Natural Language Processing (EMNLP)*.

Hui Zou and Trevor Hastie. 2005. Regularization and variable selection via the Elastic Net. *Journal of the Royal Statistical Society B*, 67:301–320.

Eugene Charniak. 1997. Statistical parsing with a context-free grammar and word statistics. In *Proceedings of the fourteenth national conference on artificial intelligence and ninth conference on Innovative applications of artificial intelligence*, AAAI'97/IAAI'97, pages 598–603. AAAI Press.

Eugene Charniak. 2000. A maximum-entropy-inspired parser. In *Proceedings of the 1st North American chapter of the Association for Computational Linguistics conference*, pages 132–139, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

Sharon A. Caraballo and Eugene Charniak. 1997. New figures of merit for best-first probabilistic chart parsing. *Computational Linguistics*, 24:275–298.

Michael Collins. 1997. Three generative, lexicalised models for statistical parsing. In *Proceedings of the 35th Annual Meeting of the Association for Computational Linguistics and Eighth Conference of the European Chapter of the Association for Computational Linguistics*, ACL '98, pages 16–23, Stroudsburg, PA, USA. Association for Computational Linguistics.

Eugene Charniak, Sharon Goldwater, and Mark Johnson. 1998. Edge-based best-first chart parsing. In *In Proceedings of the Sixth Workshop on Very Large Corpora*, pages 127–133. Morgan Kaufmann.

Alexandre Allauzen and Héléne Bonneau-Maynard. 2008. Training and evaluation of pos taggers on the french multitag corpus. In *Proceedings of the Sixth International Language Resources and Evaluation (LREC'08)*, Marrakech, Morocco, may.

Olivier Galibert. 2009. *Approches et méthodologies pour la réponse automatique à des questions adaptées à un cadre interactif en domaine ouvert*. Ph.D. thesis, Université Paris Sud, Orsay.

Rosset Sophie, Galibert Olivier, Bernard Guillaume, Bilinski Eric, and Adda Gilles. The LIMSI multilingual, multitask QAst system. In *Proceedings of the 9th Cross-language evaluation forum conference on Evaluating systems for multilingual and multimodal information access*, CLEF'08, pages 480–487, Berlin, Heidelberg, 2009. Springer-Verlag.

Azeddine Zidouni, Sophie Rosset, and Hervé Glotin. 2010. Efficient combined approach for named entity recognition in spoken language. In *Proceedings of the International Conference of the Speech Communication Assosiation (Interspeech)*, Makuhari, Japan

John Makhoul, Francis Kubala, Richard Schwartz, and Ralph Weischedel. 1999. Performance measures for information extraction. In *Proceedings of DARPA Broadcast News Workshop*, pages 249–252.

Adwait Ratnaparkhi. 1999. Learning to Parse Natural Language with Maximum Entropy Models. Journal of Machine Learning, vol. 34, issue 1-3, pages 151–175.

Michael Collins and Terry Koo. 2005. Discriminative Re-ranking for Natural Language Parsing. Journal of Machine Learning, vol. 31, issue 1, pages 25–70.

Clark, Stephen and Curran, James R. 2007. Wide-Coverage Efficient Statistical Parsing with CCG and Log-Linear Models. Journal of Computational Linguistics, vol. 33, issue 4, pages 493–552.

Finkel, Jenny R. and Kleeman, Alex and Manning, Christopher D. 2008. Efficient, Feature-based, Conditional Random Field Parsing. Proceedings of the Association for Computational Linguistics, pages 959–967, Columbus, Ohio.

Michael Auli and Adam Lopez 2011. Training a Log-Linear Parser with Loss Functions via Softmax-Margin. Proceedings of Empirical Methods for Natural Language Processing, pages 333–343, Edinburgh, U.K.

Tang, Jie and Hong, MingCai and Li, Juan-Zi and Liang, Bangyong. 2006. Tree-Structured Conditional Random Fields for Semantic Annotation. Proceedgins of the International Semantic Web Conference, pages 640–653, Edited by Springer.

Olivier Galibert; Sophie Rosset; Cyril Grouin; Pierre Zweigenbaum; Ludovic Quintard. 2011. Structured and Extended Named Entity Evaluation in Automatic Speech Transcriptions. IJCNLP 2011.

Marco Dinarelli, Sophie Rosset. Models Cascade for Tree-Structured Named Entity Detection IJCNLP 2011.