Conditional Random Fields for XML Trees

Florent Jousse[1], Rémi Gilleron[2], Isabelle Tellier[2], and Marc Tommasi[2]

INRIA Futurs and Lille University, LIFL, Mostrare Project
http://www.grappa.univ-lille3.fr/mostrare
[1]jousse@grappa.univ-lille3.fr, [2]first.last@univ-lille3.fr

Abstract. We present XML Conditional Random Fields (XCRFs), a framework for building conditional models to label XML data. XCRFs are Conditional Random Fields over unranked trees (where every node has an unbounded number of children). The maximal cliques of the graph are triangles consisting of a node and two adjacent children. We equip XCRFs with efficient dynamic programming algorithms for inference and parameter estimation. We experiment XCRFs on tree labeling tasks for structured information extraction and schema matching. Experimental results show that labeling with XCRFs is suitable for these problems.

1 Introduction

We address the task of labeling XML documents with Conditional Random Fields (CRFs). Many different problems in information science, such as information extraction, data integration, data matching and schema matching, are performed on XML documents and can be dealt with using XML labeling.

Lafferty *et al* have introduced CRFs in [LMP01]. A CRF represents a conditional distribution $p(\boldsymbol{y}|\boldsymbol{x})$ with an associated graphical structure. CRFs have been successfully used in many sequence labeling tasks such as those arising in part-of-speech tagging [SRM04], shallow parsing [SP03], named entity recognition [ML03] and information extraction [PMWC03,SC04]; for an overview, see Sutton and McCallum's survey [SM06]. The idea of defining CRFs for tree structured data has shown up only recently. Basically, the propositions differ in the graphical structure associated with the CRFs. In [RKK+02], the output variables are independent. Other approaches such as [CC04,Sut04] define the graphical structure on rules of context-free or categorial grammars. Viola and Narasimhan in [VN05] consider discriminative context-free grammars, trying to combine the advantages of nongenerative approaches (such as CRFs) and the readability of generative ones. All these approaches apply to ranked rather than unranked trees. As far as we know, their graphical models are limited to edges.

We develop XCRFs, a new instance of CRFs that properly accounts for the inherent tree structure of XML documents. In an XML document, every node has an unlimited number of ordered children, and a possibly unbounded number of unordered attributes. The graphical structure for XCRFs is defined by: for ordered (parts of the) trees, the maximal cliques of the graph are all triangles

consisting of a node and two adjacent children; for unordered (parts of the) trees, the maximal cliques are edges consisting of a node and one child.

We define efficient dynamic programming algorithms for the inference problem and the parameter estimation problem in XCRFs. Because of the unranked property of XML trees, algorithms for XCRFs implement two recursions: an horizontal recursion following the child ordering and a vertical recursion following the sibling ordering.

We have implemented XCRFs in a system for labeling XML trees (`treecrf.gforge.inria.fr`). The system allows to label elements, attributes and text nodes of XML trees. In a first set of experiments, we show that attribute features and triangle features significantly improve the performance of XCRFs in XML tree labeling tasks. To the best of our knowledge, no alternative system for labeling trees exists, because so far only *ad hoc* solutions have been used. Nevertheless, in a second set of experiments, we have applied XCRFs on XML tree labeling tasks for schema matching and structured information extraction. For instance, for schema matching on the real estate domain, we show that the XCRF system performs really well although it does not use domain constraints or integrity constraints like the LSD system [DDH01].

2 Conditional Random Fields

We refer to [SM06] for a complete introduction to CRFs. A CRF is a conditional distribution with an associated graphical structure. Let $\boldsymbol{X}$ and $\boldsymbol{Y}$ be two random fields, let $\mathcal{G}$ be an undirected graph over $\boldsymbol{Y}$. Let $\mathcal{C}$ be the set of all cliques of $\mathcal{G}$. The conditional probability distribution is of the form:

$$p(\boldsymbol{y}|\boldsymbol{x}) = \frac{1}{Z(\boldsymbol{x})} \prod_{c \in \mathcal{C}} \psi_c(\boldsymbol{y}_c, \boldsymbol{x})$$

where ψ_c is the potential function of the clique c and $Z(\boldsymbol{x})$ is a normalization factor. Each potential function has the form:

$$\psi_c(\boldsymbol{y}_c, \boldsymbol{x}) = \exp\Big(\sum_k \lambda_k f_k(\boldsymbol{y}_c, \boldsymbol{x}, c)\Big)$$

for some real-valued parameter vector $\Lambda = \{\lambda_k\}$, and for some set of real-valued feature functions $\{f_k\}$. This form ensures that the family of distributions parameterized by Λ is an exponential family. The feature function values only depend on $\boldsymbol{y}_c$, *i.e.* the assignments of the random variables in the clique c, and the whole observable $\boldsymbol{x}$. The two main problems that arise for CRFs are:

Inference: given an observable $\boldsymbol{x}$, find the most likely labeling $\hat{\boldsymbol{y}}$ for $\boldsymbol{x}$, *i.e.* compute $\hat{\boldsymbol{y}} = \arg\max_{\boldsymbol{y}} p(\boldsymbol{y}|\boldsymbol{x})$.

Training: given a sample set S of pairs $\{(\boldsymbol{x}^{(i)}, \boldsymbol{y}^{(i)})\}$, learn the best real-valued parameter vector Λ according to some criteria. In this paper, the criterion for training is the maximum conditional penalized log-likelihood.

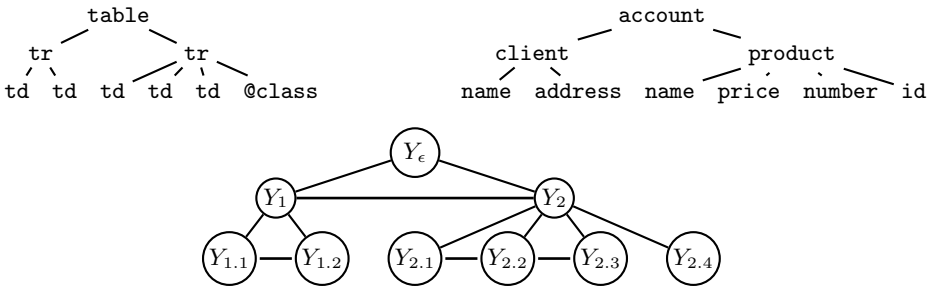


Fig. 1. An ordered unranked tree, its labeling and its graph.

3 CRFs for XML Trees

XML documents are represented by their DOM tree. We only consider element nodes, attribute nodes and text nodes of the DOM representation. Other types of nodes[1] are not concerned by labeling tasks. Attribute nodes are unordered, while element nodes and text nodes are ordered. We identify a node by a position which is a sequence of integers n and we denote by x_n the symbol in position n in the tree $\boldsymbol{x}$. The k ordered children of a node in position n are identified by positions $n.1$ to $n.k$. The unordered children are identified by positions $n.(k+1)$ and higher. As a running example, consider the two XML trees $\boldsymbol{x}$ (on the left) and $\boldsymbol{y}$ (on the right) in Figure 1. The set of nodes for both trees is $\{\epsilon, 1, 1.1, 1.2, 2, 2.1, 2.2, 2.3, 2.4\}$, ϵ being the root. The symbol in position 2.1 in $\boldsymbol{x}$ is `td`; in its labeling $\boldsymbol{y}$, the label in position 2.1 is `name`. $\{2.1, 2.2, 2.3, 2.4\}$ is the set of children of 2, where 2.1, 2.2 and 2.3 are ordered children and 2.4 is an unordered child of 2.

With every set of nodes, we associate a random field $\boldsymbol{X}$ of observable variables X_n and a random field $\boldsymbol{Y}$ of output variables Y_n where n is a position. The realizations of X_n will be the symbols of the input trees, and the realizations of Y_n will be the labels of their labelings. In the following, we freely identify realizations of these random fields with ordered unranked trees.

For their ordered parts, the structure of XML trees is governed by the sibling and the child orderings. We translate this structural property into XCRFs, by defining triangle feature functions that have the form:

$$f_k\big(y_n, y_{n.i}, y_{n.(i+1)}, \boldsymbol{x}, n.i\big) \ . \tag{3.1}$$

Their arguments are the labels assigned to the node n and to two consecutive children of it ($n.i$ and $n.(i+1)$), the whole observable $\boldsymbol{x}$, and the identifier of the clique in the tree $n.i$. For unordered parts of XML trees there is no next-sibling ordering, feature functions are thus only defined over nodes (node features) and pairs of nodes (edge features).

In Figure 1 (bottom), we show the graph for our running example. We denote by $\mathcal{C}$ the set of cliques in the dependency graph. We use f_k to index every

[1] comments, processing instructions...

feature function. To shorten the presentation, node, edge and triangle feature are all written like in (3.1). Each f_k is associated with a real-valued parameter λ_k, defining the vector $\Lambda = \{\lambda_k\}$. It is worth pointing out that the same set of feature functions with the same parameters is used for every clique in the graph. The conditional probability distribution for an XCRF can be written as:

$$p(\boldsymbol{y}|\boldsymbol{x}) = \frac{1}{Z(\boldsymbol{x})} \prod_{n.i \in \mathcal{C}} \exp\big(\sum_k \lambda_k f_k(y_n, y_{n.i}, y_{n.(i+1)}, \boldsymbol{x}, n.i)\big)$$

$$\text{where } Z(\boldsymbol{x}) = \sum_{\boldsymbol{y}} \Big(\prod_{n.i \in \mathcal{C}} \exp\big(\sum_k \lambda_k f_k(y_n, y_{n.i}, y_{n.(i+1)}, \boldsymbol{x}, n.i)\big)\Big)$$

Inference XCRFs are a particular case of graphical models. The treewidth of undirected graphs for XCRFs is 2. For every graph associated with an XCRF, a junction tree can be computed in linear time. Then the belief propagation algorithm can be applied [TAK02,SRM04]. However using the knowledge of the tree-shaped graphical structures associated with XCRFs, we propose a dynamic programming algorithm for inference in XCRFs. The algorithm is based on the dynamic programming algorithm for probabilistic context-free grammars but introduces another set of variables due to the possibily large number of children.

Training Training an XCRF means learning its parameter vector Λ. We are given iid training data S of pairs of the form (observable tree, labeled tree). Parameter estimation is performed by penalized maximum likelihood. The conditional log-likelihood, defined as $\mathcal{L}_\Lambda = \sum_{(\boldsymbol{x},\boldsymbol{y})\in S} \log p(\boldsymbol{y}|\boldsymbol{x};\Lambda)$, is used. This function is concave and the global optimum is the vector of parameters with which the first derivative is null. However, finding analytically this derivative with respect to all the model parameters is impossible. The L-BFGS gradient ascent [BLNZ95], which requires the computation of the partial derivatives of $\mathcal{L}_\Lambda$ for each parameter, is therefore used. To make these computations tractable, we introduce a dynamic programming algorithm using both forward-backward variables and inside-outside variables.

$Z(\boldsymbol{x})$ can be computed in $O(N \times M^3)$ where N is the number of nodes of $\boldsymbol{x}$ and M is the number of distinct labels in $\mathcal{Y}$. This result can be extended to the computation of the marginal probabilities in the gradient. This leads to an overall complexity for training in $O(N \times M^3 \times G)$ where N is the total number of nodes of the trees in the input sample S, M is the number of distinct labels in $\mathcal{Y}$, and G is the number of gradient steps. For linear chain CRFs only a factor M^2 occurs.

4 Experiments with the XCRF System

4.1 The XCRF System

The XCRF model is implemented by a freely available JAVA library[2]. For training, the parameters are estimated by maximizing the penalized log-likelihood. This implementation is a stochastic system which allows to label element, attribute and text nodes of XML trees. It provides the ability to not label some nodes by assigning them a label which means "not labeled". Labeling a medium-sized (about 1000 nodes) XML tree with an XCRF built on 100 features is almost immediate. An XCRF is specified by an XML file. Feature functions are 0-1 valued functions defined by XPATH expressions. There are node features, edge features, attribute features (edge features for unordered children) and triangle features.

4.2 Feature Generation

Users can easily introduce domain knowledge via the definition of feature functions in XCRFs. But to be fair, in all our experiments, feature functions are automatically generated from the training set, using a syntactic and domain-independent procedure. **Structure features** are node features, edge features and triangle features which are based on node symbols and labels. For instance, let 1_p be 1 if and only if p is true. Consider a tree $\boldsymbol{x} = \texttt{tr}(\texttt{td},\texttt{td})$ and its labeling $\boldsymbol{y} = 0(1,2)$. The triangle feature $f(y_n, y_{n.i}, y_{n.(i+1)}, \boldsymbol{x}, n.i) = 1_{\{y_n=0\}}\ 1_{\{y_{n.i}=1\}}\ 1_{\{y_{n.(i+1)}=2\}}\ 1_{\{x_n=tr\}}\ 1_{\{x_{n.i}=td\}}\ 1_{\{x_{n.(i+1)}=td\}}$ is generated together with two edge features and three node features. **Attribute features** are based on attribute values. We also preprocess documents: additional information on the structure (number of children, depth, etc.) or on the textual content of a leaf (ContainsComma, ContainsColon, IsANumber, etc) are encoded into attributes. Attribute features are therefore generated from both original and preprocessed attributes. Consider a node n of x which is labeled with 0 and has an attribute a at position i whose value is 2 labeled with 1. An attribute feature $f(y_n, y_{n.i}, \boldsymbol{x}, n.i) = 1_{\{y_n=0\}}\ 1_{\{y_{n.i}=1\}}\ 1_{\{x_{n.i}=2\}}$ is generated.

4.3 Interest of Triangle Features

We first want to experimentally evaluate the significance of the triangle features, and to show that learning is successful with small datasets.

To evaluate our system we provide recall, precision and F1-measure over the number of nodes. Precision and recall are the ratio of the number of correctly labeled nodes to respectively the total number of labeled nodes and the total number of nodes which had to be labeled. Using a simple accuracy measure would take into account the nodes which are not labeled. The results would therefore be biased.

Triangle features. The "Courses" dataset[4] collected by Doan consists of 960 XML documents of about 20 nodes each, containing course information from five

[2] `http://treecrf.gforge.inria.fr/`

Features	F1	%docs
Edge	52.19	0
Edge & Attribute	84.91	26.51
Triangle	93.62	62.24
Triangle & Attribute	99.84	98.93

Nb Docs	F1	%docs
5	82.15	50.97
10	91.34	69.14
20	96.65	82
50	98.77	93.82

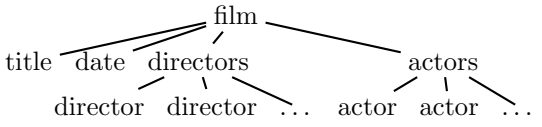
Table 1. Left: Triangle features versus edge features. **Right:** Varying the size of the training set

universities. All XML tags are removed and replaced with a unique and therefore uninformative one. The task consists in relabeling the XML documents with the original 14 distinct tags. We train XCRFs with node and edge structure features then with node, edge and triangle features. In both cases, we also train the XCRFs with or without attribute features. Results of Table 1 are means of 5 iterations in which XCRFs are trained on one fifth of the corpus and evaluated on the other 4 fifths. They show the F1-measure on document nodes and the percentage of XML documents that were completely correctly labeled. They confirm the relevance of triangle features. Indeed, results are far better with triangle features than when attribute features (containing also structural information) are added to edge features. Labeling is almost perfect with triangle and attribute features.

Number of examples. We consider the same task using structure features and attribute features. This time, we want to evaluate the impact of the number of examples in the learning set. Experimental results in Table 1 show that with only 20 documents for learning (192 in the previous experiment), the system already achieves more than 96% in F1-measure. When using more documents, the performances still rise, but very slowly. It is also interesting to note that when the XCRF has about 300 feature functions, the labeling of these documents is still immediate.

4.4 Experiments on Applicative Domains

Now, we apply our tree-labeling learning system to two applicative domains. The first one is structured information extraction, the second one is schema matching, both considered as a labeling task.



Structured Information Extraction. For this experiment, we take 558 XML documents from the MovieDB corpus used for the XML Document Mining Challenge[5]. The average number of nodes in these documents is 170. The task is to extract data according to the target DTD shown on the right. This purpose can be achieved by labeling the input XML documents according to this DTD. Note that two nodes with the same input tag might be labeled with different output tags, depending on the context. For instance, the tag `name` can be labeled either by `actor` or by `director`. To evaluate the XCRF

[4] `http://anhai.cs.uiuc.edu/archive`

Nb docs	Recall	Precision	F1	% docs
5	100	97.91	98.94	71.32
10	100	99.55	99.77	89.84
20	100	99.99	99.99	99.63

Table 2. Results on MovieDB Structured IE.

system on this task, we run 10 experiments, each time randomly choosing 5, 10 or 20 labeled documents on which an XCRF is trained and tested on the remaining documents. The results, very promising, are provided in Table 2.

Schema Matching. For the problem of schema matching, we evaluate XCRFs on the "Real Estate I" dataset[4], collected by Doan. This corpus, built from five real estate websites, describes house listing information and contains about 10000 documents of about 35 nodes each. Each of the five sources has its own schema. A unique mediated schema with 16 tags is also known. The task thus consists in labeling the nodes of the documents in their source schema with their corresponding tag in the mediated schema. As in [DDH01], for every experiment, it is supposed that mappings are known for three sources out of the five ones and that documents labeled according to the mediated schema are the training set. We run 20 experiments, each time choosing at random 3 sources with which an XCRF with structure and attribute features is learnt. We took 5 labeled documents from each source. All the documents from the remaining 2 sources are used to evaluate the XCRF. The system achieved an excellent recall of 99%, and 88% of F1-measure. Unfortunately, these results can not be compared to the ones achieved by the LSD system given in [DDH01]. Indeed, [DDH01] measure the ratio of correct mappings between a tag in the source schema and its corresponding tag in the mediated schema. Since we are using a conditional model to label tree nodes, the same tag in the source schema can be mapped to different tags in the mediated schema depending on the context. Therefore, we can not provide this measure. However, it is still worth noting that we achieve very good results without using the domain knowledge, such as domain constraints or integrity constraints, that was used in the LSD system.

5 Conclusion

We have shown that XCRFs are a very relevant model for labeling XML trees. Experimental results show the significance of attribute and triangle features. Also, preliminary experimental results show that XCRFs perform very well on tasks such as structured information extraction or schema matching. Various extensions are being considered.

First, when labeling an XML tree, one can be interested in labeling inside the text nodes, *i.e.* assigning different labels to different parts of text. To do so, the use of both the tree structure and the text sequence is needed. Thus, combining

[5] `http://xmlmining.lip6.fr/Corpus`

linear chain CRFs and XCRFs could be a good way of taking advantage of both the structured and the linear view of XML documents.

Second, in the XCRF system, parameter estimation is done by maximizing the conditional probability $p(\boldsymbol{y}|\boldsymbol{x})$, meaning we try to maximize the number of completely correctly labeled XML documents. Sometimes, for instance in schema matching, one might instead prefer to maximize the number of correctly labeled nodes. To do so, another criterion for learning could be the maximum pseudo-likelihood, which consists in maximizing the marginal probabilities $p(y_n|\boldsymbol{x})$.

References

[BLNZ95] Richard H. Byrd, Peihuang Lu, Jorge Nocedal, and Ciyou Zhu. A limited memory algorithm for bound constrained optimization. *SIAM Journal on Computing*, 16(5):1190–1208, 1995.

[CC04] Stephen Clark and James R. Curran. Parsing the WSJ using CCG and log-linear models. In *Proc. of ACL*, pages 103–110, 2004.

[DDH01] AhHai Doan, Pedro Domingos, and Alon Halevy. Reconciling schemas of disparate data sources: A machine-learning approach. In *Proc. of the ACM SIGMOD Conference*, pages 509–520, 2001.

[LMP01] John D. Lafferty, Andrew McCallum, and Fernando C. N. Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proc. of ICML*, pages 282–289, 2001.

[ML03] A. McCallum and W. Li. Early results for named entity recognition with conditional random fields. In *Proc. of CoNLL'2003*, 2003.

[PMWC03] David Pinto, Andrew McCallum, Xing Wei, and W. Bruce Croft. Table extraction using conditional random fields. In *Proc. of the ACM SIGIR*, pages 235–242, 2003.

[RKK+02] S. Riezler, T. King, R. Kaplan, R. Crouch, J. Maxwell, and M. Johnson. Parsing the wall street journal using a lexical-functional grammar and discriminative estimation techniques. In *Proc. of ACL*, pages 271–278, 2002.

[SC04] Sunita Sarawagi and William W. Cohen. Semi-markov conditional random fields for information extraction. In *Proc. of NIPS*, pages 1185–1192, 2004.

[SM06] Charles Sutton and Andrew McCallum. *Introduction to Statistical Relational Learning*, chapter An Introduction to Conditional Random Fields for Relational Learning. MIT Press, lise getoor and ben taskar edition, 2006.

[SP03] Fei Sha and Fernando Pereira. Shallow parsing with conditional random fields. In *Proc. of HLT-NAACL*, pages 213–220, 2003.

[SRM04] Charles Sutton, Khashayar Rohanimanesh, and Andrew McCallum. Dynamic conditional random fields: Factorized probabilistic models for labeling and segmenting sequence data. In *Proc. of ICML*, pages 783–790, 2004.

[Sut04] Charles Sutton. Conditional probabilistic context-free grammars. Master's thesis, University of Massachusetts, 2004.

[TAK02] Ben Taskar, Pieter Abbeel, and Daphne Koller. Discriminative probabilistic models for relational data. In *Proc. of UAI02*, pages 485–492, 2002.

[VN05] Paul Viola and Mukund Narasimhan. Learning to extract information from semistructured text using a discriminative context free grammar. In *Proc. of the ACM SIGIR*, pages 330–337, 2005.

[Wal02] Hanna Wallach. Efficient training of conditional random fields. Master's thesis, University of Edinburgh, 2002.