# A Probabilistic Learning Method for XML Annotation of Documents

**Boris Chidlovskii**[1]
[1]Xerox Research Centre Europe
6, chemin de Maupertuis
38240 Meylan, France

**Jérôme Fuselier**[1,2]
[2]Université de Savoie - Laboratoire SysCom
Domaine Universitaire
73376 Le Bourget-du-Lac, France

## Abstract

We consider the problem of semantic annotation of semi-structured documents according to a target XML schema. The task is to annotate a document in a tree-like manner where the annotation tree is an instance of a tree class defined by DTD or W3C XML Schema descriptions. In the probabilistic setting, we cope with the tree annotation problem as a *generalized probabilistic context-free parsing* of an observation sequence where each observation comes with a probability distribution over terminals supplied by a probabilistic classifier associated with the content of documents. We determine the most probable tree annotation by maximizing the joint probability of selecting a terminal sequence for the observation sequence and the most probable parse for the selected terminal sequence.

## 1 Introduction

The future of the World Wide Web is often associated with the Semantic Web initiative which has as a target a wide-spread document reuse, re-purposing and exchange, achieved by means of making document markup and annotation more machine-readable. The success of the Semantic Web initiative depends to a large extent on our capacity to move from *rendering-oriented* markup of documents, like PDF or HTML, to *semantic-oriented* document markup, like XML and RDF.

In this paper, we address the problem of *semantic annotation of HTML documents according to a target XML schema*. A tree-like annotation of a document requires that the annotation tree be an instance of the target schema, described in a DTD, W3C XML Schema or another schema language. Annotation trees naturally generalize flat annotations conventionally used in information extraction and wrapper induction for Web sites.

The migration of documents from rendering-oriented formats, like PDF and HTML, toward XML has recently become an important issue in various research communities [2; 3; 11; 12]. The majority of approaches either make certain assumptions about the source and target XML documents, like a conversion through a set of local transformations [3], or entail the transformation to particular tasks, such as the semantic annotation of dynamically generated Web pages in news portals [11] or the extraction of logical structure from page images [12].

In this paper, we consider the general case of tree annotation of semi-structured documents. We make *no assumptions about the structure of the source and target documents* or their possible similarity. We represent the document content as a sequence of observations $\mathbf{x} = \{x_1, \ldots, x_n\}$, where each observation $x_i$ is a content fragment. In the case of HTML documents, such a fragment may be one or multiple leaves, often surrounded with rich contextual information in the form of HTML tags, attributes, etc. The tree annotation of sequence $\mathbf{x}$ is given by a pair $(\mathbf{y}, d)$, where $\mathbf{y}$ and $d$ refer to leaves and internal nodes of the tree, respectively. The sequence $\mathbf{y} = \{y_1, \ldots, y_n\}$ can be seen, on one side, as labels for observations in $\mathbf{x}$, and on the other side, as a terminal sequence for tree $d$ that defines the internal tree structure over $\mathbf{y}$ according to the target XML schema.

In supervised learning, the document annotation system includes selecting the tree annotation model and training the model parameters from a training set $S$ given by triples $(\mathbf{x}, \mathbf{y}, d)$. We adopt a probabilistic setting, by which we estimate the probability of an annotation tree $(\mathbf{y}, d)$ for a given observation sequence $\mathbf{x}$ and address the problem of finding the pair $(\mathbf{y}, d)$ of maximal likelihood.

We develop a modular architecture for the tree annotation of documents that includes two major components. The first component is a probabilistic context-free grammar (PCFG) which is a probabilistic extension to the corresponding (deterministic) XML schema definition. The PCFG rules may be obtained by rewriting the schema's element declarations (in the case of a DTD) or element and type definitions (in the case of a W3C XML Schema) and the rule probabilities are chosen by observing rule occurrences in the training set, similar to learning rule probabilities from tree-bank corpora for NLP tasks. PCFGs offer the efficient inside-outside algorithm for finding the most probable parse for a given sequence $\mathbf{y}$ of terminals. The complexity of the algorithm is $O(n^3 \cdot |N|)$, where $n$ is the length of sequence $\mathbf{y}$

and $|N|$ is the number of non-terminals on the PCFG.

The second component is a probabilistic classifier for predicting the terminals $y$ for the observations $x_i$ in $\mathbf{x}$. In the case of HTML documents, we use the maximum entropy framework [1], which proved its efficiency when combining content, layout and structural features extracted from HTML documents for making probabilistic predictions $p(y)$ for $x_i$.

With the terminal predictions supplied by the content classifier, the tree annotation problem represents *the generalized case of probabilistic parsing*, where each position $i$ in sequence $\mathbf{y}$ is defined not with a specific terminal, but with a terminal probability $p(y)$. Consequently, we consider the sequential and joint evaluations of the maximum likelihood tree annotation for observation sequences. In the joint case, we develop a generalized version of the inside-outside algorithm that determines the *most probable annotation tree* $(\mathbf{y}, d)$ according to the PCFG and the distributions $p(y)$ for all positions $i$ in $\mathbf{x}$. We show that the complexity of the generalized inside-outside algorithm is $O(n^3 \cdot |N| + n \cdot |T| \cdot |N|)$, where $n$ is the length of $\mathbf{x}$ and $\mathbf{y}$, and where $|N|$ and $|T|$ are the number of non-terminals and terminals in the PCFG.

We also show that the proposed extension of the inside-outside algorithm imposes the *conditional independence requirement*, similar to the Naive Bayes assumption, on estimating terminal probabilities. We test our method on two collections and report an important advantage of the joint evaluation over the sequential one.

## 2 XML annotation and schema

XML annotations of documents are trees where inner nodes determine the tree structure, and the leaf nodes and tag attributes refer to the document content. XML annotations can be abstracted as the class T of *unranked labeled rooted trees* defined over an alphabet $\Sigma$ of tag names [9]. The set of trees over $\Sigma$ can be constrained by a *schema* $D$ that is defined using DTD, W3C XML Schema or other schema languages.

DTDs and an important part of W3C XML Schema descriptions can be modeled as extended context-free grammars [10], where regular expressions over alphabet $\Sigma$ are constructed by using the two basic operations of concatenation $\cdot$ and disjunction $|$ and with occurrence operators $*$ (Kleene closure), ? ($a? = a|\epsilon$) and + ($a+ = a \cdot a*$). An *extended context free grammar* (ECFG) is defined by the 4-tuple $G = (T, N, \mathcal{S}, R)$, where $T$ and $N$ are disjoint sets of terminals and nonterminals in $\Sigma$, $\Sigma = T \cup N$; $\mathcal{S}$ is an initial nonterminal and $R$ is a finite set of production rules of the form $A \rightarrow \alpha$ for $A \in N$, where $\alpha$ is a regular expression over $\Sigma = T \cup N$. The language $L(G)$ defined by an ECFG $G$ is the set of terminal strings derivable from the starting symbol $S$ of $G$. Formally, $L(G) = \{w \in \Sigma^* | S \Rightarrow w\}$, where $\Rightarrow$ denotes the transitive closure of the derivability relation. We represent as a *parse tree* $d$ any sequential form that reflects the derivational steps. The *set of parse trees* for $G$ forms the set $\mathcal{T}(G)$ of unranked labeled rooted trees constrained with schema $G$.

### 2.1 Tree annotation problem

When annotating HTML documents accordingly to a target XML schema, the main difficulty arises from the fact that the source documents are essentially layout-oriented, and the use of tags and attributes is not necessarily consistent with elements of the target schema. The irregular use of tags in documents, combined with complex relationships between elements in the target schema, makes the manual writing of HTML-to-XML transformation rules difficult and cumbersome.

In supervised learning, the content of source documents is presented as a sequence of observations $\mathbf{x} = \{x_1, \ldots, x_n\}$, where any observation $x_i$ refers to a content fragment, surrounded by rich contextual information in the form of HTML tags, attributes, etc. The tree annotation model is defined as a mapping $X \rightarrow (Y, D)$ that maps the observation sequence $\mathbf{x}$ into a pair $(\mathbf{y}, d)$ where $\mathbf{y}=\{y_1, \ldots, y_n\}$ is a terminal sequence and $d$ is a parse tree of $\mathbf{y}$ according to the target schema or equivalent PCFG $G$, $\mathcal{S} \Rightarrow \mathbf{y}$. The training set $S$ for training the model parameters is given by a set of triples $(\mathbf{x}, \mathbf{y}, d)$.

To determine the most probable tree annotation $(\mathbf{y}, d)$ for a sequence $\mathbf{x}$, we maximize the joint probability $p(\mathbf{y}, d|\mathbf{x}, G)$, given the sequence $\mathbf{x}$ and PCFG $G$. Using the Bayes theorem and the independence of $x$ and $G$, we have

$$p(\mathbf{y}, d|\mathbf{x}, G) = p(d|\mathbf{y}, G) \cdot p(\mathbf{y}|\mathbf{x}), \tag{1}$$

where $p(\mathbf{y}|\mathbf{x})$ is the probability of terminal sequence $\mathbf{y}$ for the observed sequence $\mathbf{x}$, and $p(d|\mathbf{y}, G)$ is the probability of the parse $d$ for $\mathbf{y}$ according the PCFG $G$. The most probable tree annotation for $\mathbf{x}$ is a pair $(\mathbf{y}, d)$ that maximizes the probability in (1),

$$(\mathbf{y}, d)_{max} = \underset{(y,d)}{argmax}\, p(d|\mathbf{y}, G) \cdot p(\mathbf{y}|\mathbf{x}). \tag{2}$$

In the following, we build a probabilistic model for tree annotation of source documents consisting of two components to get the two probability estimates in (2). The first component is a probabilistic extension of the target XML schema; for a given terminal sequence $\mathbf{y}$, it finds the most probable parse $p(d|\mathbf{y}, G)$ for sequences according to the PCFG $G$, where rule probabilities are trained from the available training set. The second component is a probabilistic content classifier $C$, it estimates the conditional probabilities $p(y|x_i)$ for annotating observations $x_i$ with terminals $y \in T$. Finally, for a given sequence of observations $\mathbf{x}$, we develop two methods for finding a tree annotation $(\mathbf{y}, d)$ that maximizes the joint probability $p(\mathbf{y}, d|\mathbf{x}, G)$ in (1).

## 3 Probabilistic context-free grammars

PCFGs are probabilistic extensions of CFGs, where each rule $A \rightarrow \alpha$ in $R$ is associated with a real number $p$ in the half-open interval (0; 1]. The values of $p$ obey the

restriction that for a given non-terminal $A \in N$, all rules for $A$ must have $p$ values that sum to 1,

$$\forall A \in N : \sum_{r=A\rightarrow\alpha, r\in R} p(r) = 1. \quad (3)$$

PCFGs have a normal form, called the Chomsky Normal Form (CNF), according to which any rule in $R$ is either $A \rightarrow B\ C$ or $A \in b$, where $A$, $B$ and $C$ are non-terminals and $b$ is a terminal. The rewriting of XML annotations requires the *binarization* of source ranked trees, often followed by an extension of the nonterminal set and the underlying set of rules. This is a consequence of rewriting nodes with multiple children as a sequence of binary nodes. The binarization rewrites any rule $A \rightarrow B\ C\ D$ as two rules $A \rightarrow BP$ and $P \rightarrow C\ D$, where $P$ is a new non-terminal.

A PCFG defines a joint probability distribution over $Y$ and $D$, random variables over all possible sequences of terminals and all possible parses, respectively. $Y$ and $D$ are clearly not independent, because a complete parse specifies exactly one or few terminal sequences. We define the function $p(\mathbf{y}, d)$ of a given terminal sequence $\mathbf{y} \in Y$ and a parse $d \in D$ as the product of the $p$ values for all of the rewriting rules $R(\mathbf{y}, d)$ used in $\mathcal{S} \Rightarrow \mathbf{y}$. We also consider the case where $d$ does not actually correspond to $\mathbf{y}$,

$$p(\mathbf{y}, d) = \begin{cases} \prod_{r\in R(\mathbf{y},d)} p(r), & \text{if } d \text{ is a parse of } y \\ 0, & \text{otherwise.} \end{cases}$$

The values of $p$ are in the closed interval [0; 1]. In the cases where $d$ is a parse of $\mathbf{y}$, all $p(r)$ values in the product will lie in the half open interval (0; 1], and so will the product. In the other case, 0 is in [0; 1] too. However, it is not always the case that $\sum_{d,\mathbf{y}} p(\mathbf{y}, d) = 1$.

The PCFG training takes as evidence the corpus of terminal sequences $\mathbf{y}$ with corresponding parses $d$ from the training set $S$. It associates with each rule an expected probability of using the rule in producing the corpus. In the presence of parses for all terminal sequences, each rule probability is set to the expected count normalized so that the PCFG constraints (3) are satisfied:

$$p(A \rightarrow \alpha) = \frac{count(A \rightarrow \alpha)}{\sum_{A\rightarrow\beta\in R} count(A \rightarrow \beta)}.$$

### 3.1 Generalized probabilistic parsing

PCFGs are used as probabilistic models for natural languages, as they naturally reflect the "deep structure" of language sentences rather than the linear sequences of words. In a PCFG language model, a finite set of words serve as a terminal set and production rules for non-terminals express the full set of grammatical constructions in the language. Basic algorithms for PCFGs that find the most likely parse $d$ for a given sequence $\mathbf{y}$ or choose rule probabilities that maximize the probability of sentence in a training set, represent [6] efficient extensions of the Viterbi and Baum-Welsh algorithms for hidden Markov models.

The tree annotation model processes sequences of observations $\mathbf{x} = \{x_1, \ldots, x_n\}$ from the infinite set $X$, where the observations $x_i$ are not words in a language (and therefore terminals in $T$) but complex instances, like HTML leaves or groups of leaves.

Content fragments are frequently targeted by various probabilistic classifiers that produce probability estimates for labeling an observation with a terminal in $T$, $p(y|x_i)$, $y \in T$, where $\sum_y p(y|x_i) = 1$. The tree annotation problem can therefore be seen as *a generalized version of probabilistic context-free parsing*, where the input sequence is given by the probability distribution over a terminal set and the most probable annotation tree requires maximizing the joint probability in (2).

A similar generalization of probabilistic parsing takes place in speech recognition. In the presence of a noisy channel for speech streams, parsing from a sequence of words is replaced by parsing from a word lattice, which is a compact representation of a set of sequence hypotheses, given by conditional probabilities obtained by special acoustic models from acoustic observations [4].

## 4 Content classifier

To produce terminal estimates for the observations $x_i$, we adopt the maximum entropy framework, according to which the best model for estimating probability distributions from data is the one that is consistent with certain constraints derived from the training set, but otherwise makes the fewest possible assumptions [1]. The distribution with the fewest possible assumptions is one with the highest entropy, and closest to the uniform distribution. Each constraint expresses some characteristic of the training set that should also be present in the learned distribution. The constraint is based on a binary feature, it constrains the expected value of the feature in the model to be equal to its expected value in the training set.

One important advantage of maximum entropy models is their flexibility, as they allow the extension of the rule system with additional syntactic, semantic and pragmatic features. Each feature $f$ is binary and can depend on $y \in T$ and on any properties of the input sequence $\mathbf{x}$. In the case of tree annotation, we include the *content features* that express properties on content fragments, like $f_1(x, y)$ ="1 if $y$ is **title** and $x$'s length is less then 20 characters, 0 otherwise", as well as the *structural and layout features* that capture the HTML context of the observation $x$, like $f_2(x, y)$="1 if $y$ is **author** and $x$'s father is `span`, 0 otherwise".

With the constraints based on the selected features $f(x, y)$, the maximum entropy method attempts to maximize the conditional likelihood of $p(y|x)$ which is represented as an exponential model:

$$p(y|x) = \frac{1}{Z_\alpha(x)} exp\left(\sum_\alpha \lambda_\alpha \cdot f_\alpha(x, y)\right), \quad (4)$$

where $Z_\alpha(x)$ is a normalizing factor to ensure that all the probabilities sum to 1,

$$Z_\alpha(x) = \sum_y exp\left(\sum_\alpha \lambda_\alpha f_\alpha(x,y)\right). \quad (5)$$

For the iterative parameter estimation of the Maximum Entropy exponential models, we use one of the quasi Newton methods, namely the Limited Memory BFGS method, which is observed to be more effective than the Generalized Iterative Scaling (GIS) and Improved Iterative Scaling (IIS) for NLP and IE tasks [7].

## 5 Sequential tree annotation

We use pairs $(\mathbf{x}, \mathbf{y})$ from triples $(\mathbf{x}, \mathbf{y}, d)$ of the training set $S$ to train the content classifier $C$ and pairs $(\mathbf{y}, d)$ to choose rule probabilities that maximize the likelihood for the instances in the training set. $C$ predicts the terminal probabilities $p(\mathbf{y}|\mathbf{x})$ for any observation $\mathbf{x}$, while the inside-outside algorithm can find the parse $d$ of the highest probability for a given terminal sequence $\mathbf{y}$.

By analogy with speech recognition, there exists a naive, sequential method to combine the two components $C$ and $G$ for computing a tree annotation for sequence $\mathbf{x}$. First, from $C$'s estimates $p(\mathbf{y}|\mathbf{x})$, we determine the (top $k$) most probable sequences $\mathbf{y}_{max,j}$ for $\mathbf{x}$, $j = 1, \ldots, k$. Second, we find the most probable parses for all $\mathbf{y}_{max,j}$, $d_{max,j} = \underset{d}{argmax}\, p(d|\mathbf{y}_{max,j}, G)$; and finally, we choose the pair $(\mathbf{y}_{max,j}, d_{max,j})$ that maximizes the product $p(\mathbf{y}_{max,j}) \times p(d_{max,j})$.

The sequential method works well if the noise level is low (in speech recognition) or if the content classifier (in the tree annotation) is accurate enough in predicting terminals $y$ for $x_i$. Unfortunately, it gives poor results once the classifier $C$ is far from 100% accuracy in $\mathbf{y}$ predictions, as it faces the impossibility of finding any parse for all the top $k$ most probable sequences $\mathbf{y}_{max,j}$.

**Example.** Consider an example target schema given by the following DTD:

```
<!ELEMENT  Book      (author, Section+)>
<!ELEMENT  Section   (title, (para | footnote)+)>
<!ELEMENT  author    (#PCDATA)>
<!ELEMENT  title     (#PCDATA)>
<!ELEMENT  para      (#PCDATA)>
<!ELEMENT  footnote  (#PCDATA)>
```

The reduction of the above schema definition to the Chomsky Normal Form will introduce extra nonterminals, so we get the PCFG $G = (T, N, \mathcal{S}, R)$, where the terminal set is $T$={`author, title, para, footnote`}, the nonterminal set is $N$= {Book, Author, SE, Section, TI, ELS, EL}, $\mathcal{S}$= Book, and $R$ includes twelve production rules.

Assume that we have trained the content classifier $C$ and the PCFG $G$ and have obtained the following probabilities for the production rules in $R$:

(0.3) Book → AU Section
(0.4) SE → Section Section
(0.8) Section → TI ELS
(0.4) ELS → EL EL
(1.0) AU → `author`
(0.8) EL → `para`
(0.7) Book → AU SE
(0.6) SE → Section SE
(0.2) Section → TI EL
(0.6) ELS → EL ELS
(1.0) TI → `title`
(0.2) EL → `footnote`.

Assume now that we test the content classifier $C$ and PCFG $G$ on a sequence of five unlabeled observations $\mathbf{x} = \{x_1, \ldots, x_5\}$. Let the classifier $C$ estimate the probability for terminals in $T$ as given in the following table:

| | $x_1$ | $x_2$ | $x_3$ | $x_4$ | $x_5$ |
|---|---|---|---|---|---|
| `author` | 0.3 | 0.2 | 0.1 | 0.1 | 0.2 |
| `title` | **0.4** | **0.4** | 0.3 | 0.3 | **0.3** |
| `para` | 0.1 | 0.2 | **0.5** | 0.2 | 0.2 |
| `footnote` | 0.2 | 0.2 | 0.1 | **0.4** | 0.2 |

According to the above probability distribution, the most probable terminal sequence $\mathbf{y}_{max}$ is composed of the most probable terminals for all $x_i$, $i = 1, \ldots, 5$. It is `'title title para footnote title'` with probability $p(\mathbf{y}_{max}) = p(\mathbf{y}_{max}|x) = \Pi_i \cdot p(\mathbf{y}_i^{max}|x_i) = 0.4 \cdot 0.4 \cdot 0.5 \cdot 0.4 \cdot 0.3 = 0.0096$. However, $\mathbf{y}_{max}$ has no corresponding parse tree in $G$. Instead, there exist two valid annotation trees for $\mathbf{x}$, $(\mathbf{y}_1, d_1)$ and $(\mathbf{y}_2, d_2)$, as shown in Figure 1. In Figure 1.b, the terminal sequence $\mathbf{y}_2$=`'author title para title para'` with the parse $d_2$=Book(AU SE(Section (TI EL) Section (TI EL))) maximizes the joint probability $p(\mathbf{y}, d|\mathbf{x}, G)$, with $p(\mathbf{y}_2) = 0.3 \cdot 0.4 \cdot 0.5 \cdot 0.3 \cdot 0.2 = 0.0036$, and

$$\begin{aligned} p(d_2) = & p(\text{Book} \to \text{AU SE}) \cdot p(\text{AU} \to \texttt{author}) \times \\ & p(\text{SE} \to \text{Section Section}) \cdot p(\text{Section} \to \text{TI EL}) \times \\ & p(\text{TI} \to \texttt{title})^2 \cdot p(\text{EL} \to \texttt{para})^2 \times \\ & p(\text{Section} \to \text{TI EL}) \\ = & 0.7 \cdot 1.0 \cdot 0.4 \cdot 0.2 \cdot 1.0^2 \cdot 0.8^2 \cdot 0.2 = 0.007172. \end{aligned}$$

Jointly, we have $p(\mathbf{y}_2) \times p(d_2) \approx 2.58 \cdot 10^{-5}$. Similarly, for the annotation tree in Figure 1.a, we have $p(\mathbf{y}_1) \times p(d_1) = 0.0048 \cdot 0.0018432 \approx 8.85 \cdot 10^{-6}$.

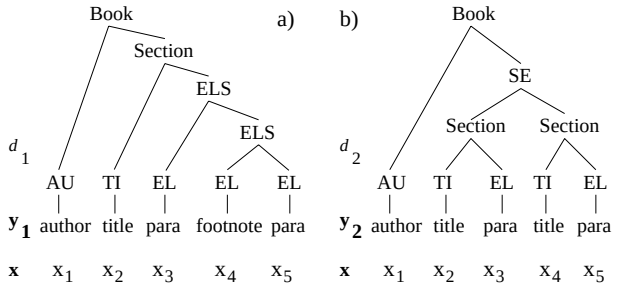


Figure 1: Tree annotations for the example sequence.

## 6 The most probable annotation tree

As the sequential method fails to find the most probable annotation tree, we try to couple the selection of terminal sequence $\mathbf{y}$ for $\mathbf{x}$ with finding the most probable parse $d$ for $\mathbf{y}$, such that $(\mathbf{y}, d)$ maximizes the probability product $p(d|\mathbf{y}, G) \cdot p(\mathbf{y}|\mathbf{x})$ in (2). For this goal, we extend the

basic inside-outside algorithm for terminal PCFGs [6]. We redefine the *inside probability* as the most probable joint probability of the subsequence of $\mathbf{y}$ beginning with index $i$ and ending with index $j$, and the most probable partial parse tree spanning the subsequence $\mathbf{y}_i^j$ and rooted at nonterminal $A$:

$$\beta_A(i,j) = max_{A,\mathbf{y}_i^j}\ p(A_{i,j} \Rightarrow \mathbf{y}_i^j) \cdot p(\mathbf{y}_i^j|\mathbf{x}). \quad (6)$$

The inside probability is calculated recursively, by taking the maximum over all possible ways that the nonterminal $A$ could be expanded in a parse,

$$\beta_A(i,j) = \quad max_{i\le q\le j}\ p(A \rightarrow BC) \cdot p(B \Rightarrow \mathbf{y}_i^q) \times p(C \Rightarrow \mathbf{y}_{q+1}^j) \cdot p(\mathbf{y}_i^j|\mathbf{x}).$$

To proceed further, we make the *independence assumption* about $p(\mathbf{y}|\mathbf{x})$, meaning that for any $q$, $i \le q \le j$, we have $p(\mathbf{y}_i^j|\mathbf{x}) = p(\mathbf{y}_i^q|\mathbf{x}) \cdot p(\mathbf{y}_{q+1}^j|\mathbf{x})$. Then, we can rewrite the above as follows

$$\beta_A(i,j) \quad = max_{i\le q\le j} p(A \rightarrow BC) \cdot p(B \Rightarrow \mathbf{y}_i^q) \times \quad (7)$$
$$p(C \Rightarrow \mathbf{y}_{q+1}^j) \cdot p(\mathbf{y}_i^q|\mathbf{x}) \cdot p(\mathbf{y}_{q+1}^j|\mathbf{x}) \quad (8)$$
$$= max_{i\le q\le j} p(A \rightarrow BC) \cdot \beta_B(i,q) \cdot \beta_C(q+1,j) \quad (9)$$

The recursion is terminated at the $\beta_{\mathcal{S}}(1,n)$ which gives the probability of the most likely tree annotation $(\mathbf{y}, d)$,

$$\beta_{\mathcal{S}}(1,n) = max\ p(\mathcal{S} \Rightarrow \mathbf{y}_1^n) \cdot p(\mathbf{y}_1^n|\mathbf{x}),$$

where $n$ is the length of both sequences $\mathbf{x}$ and $\mathbf{y}$.

The initialization step requires some extra work, as we select among all terminals in $T$ being candidates for $y_k$:

$$\beta_A(k,k) = max_{y_k} p(A \rightarrow y_k) \cdot p(y_k|\mathbf{x}). \quad (10)$$

It can be shown that the redefined inside function converges to a local maximum in the $(Y, D)$ space. The extra work during the initialization step takes $O(n \cdot |T| \cdot |N|)$ time which brings the total complexity of the extended IO algorithm to $O(n^3 \cdot |N| + n \cdot |T| \cdot |N|)$.

The independence assumption established above represents the *terminal conditional independence*, $p(\mathbf{y}|\mathbf{x}) = \prod_{i=1}^{n} p(y_i|\mathbf{x})$ and matches the Naive Bayes assumption. The assumption is frequent in text processing; it simplifies the computation by ignoring the correlations between terminals. Here however it becomes a requirement for the content classifier. While PCFGs are assumed to capture all (short- and long- distance) relations between terminals, the extended inside algorithm (9)-(10) imposes the terminal conditional independence when building the probabilistic model. This impacts the feature selection for the maximum entropy model, by disallowing features that include terminals of neighbor observations $y_{i-1}$, $y_{i+1}$, etc, as in the maximum entropy extension with HMM and CRF models [8; 5].

## 7 Experimental results

We have tested our method for XML annotation on two collections. One is the collection of 39 Shakespearean plays available in both HTML and XML format.[1] 60 scenes with 17 to 189 leaves were randomly selected for the evaluation. The DTD fragment for scenes consists of 4 terminals and 6 non-terminals. After the binarization, the PCFG in CNF contains 8 non-terminals and 18 rules.

The second collection, called TechDoc, includes 60 technical documents from repair manuals. [2] The target documents have a fine-grained semantic granularity and are much deeper than in the Shakespeare collection; the longest document has 218 leaves. The target schema is given by a complex DTD with 27 terminals and 35 nonterminals. The binarization increased the number of non-terminals to 53. For both collections, a content observation refers to a PCDATA leaf in HTML.

To evaluate the annotation accuracy, we use two metrics. The *terminal error ratio* (TER) is similar to the word error ratio used in natural language tasks; it measures the percentage of correctly determined terminals in test documents. The second metric is the *non-terminal error ratio* (NER) which is the percentage of correctly annotated sub-trees.

As content classifiers, we test with the maximum entropy (ME) classifier. For the ME model, we extract 38 *content features* for each observation, such as the number of words in the fragment, its length, POS tags, textual separators, etc. Second, we extract 14 *layout and structural features* include surrounding tags and all associated attributes. Beyond the ME models, we use the maximum entropy Markov models (MEMM) which extends the ME with hidden Markov structure and terminal conditional features [8]. The automaton structure used in MEMM has one state per terminal.

In all tests, a cross-validation with four folds is used. ME and MEMM were first tested alone on both collections. The corresponding TER values for the most probable terminal sequences $\mathbf{y}_{max}$ serve a reference for methods coupling the classifiers with the PCFG. When coupling the ME classifier with the PCFG, we test both the sequential and joint methods. Additionally, we included a special case MEMM-PCFG where the content classifier is MEMM and therefore the terminal conditional independence is not respected.

The results of all the tests are collected in Table 1. The joint method shows an important advantage over the sequential method, in particular in the TechDoc case, where the ME content classifier alone achieves 86.23% accuracy and the joint method reduces the errors in terminals by 1.36%. Instead, coupling MEMM with the PCFG reports a decrease of TER values and a much less important NER increase.

[1] http://metalab.unc.edu/bosak/xml/eg/shaks200.zip.
[2] Available from authors on request.

| Method | TechDoc | | Shakespeare | |
|---|---|---|---|---|
| | TER | NER | TER | NER |
| ME | 86.23 | – | **100.0** | – |
| MEMM | 78.16 | – | 99.91 | – |
| Seq-ME-PCFG | 86.23 | 9.38 | **100.0** | 82.87 |
| Jnt-ME-PCFG | **87.59** | **72.95** | 99.97 | **99.79** |
| Jnt-MEMM-PCFG | 75.27 | 56.25 | 98.09 | 94.01 |

Table 1: Evaluation results.

## 8 Relevant Work

Since the importance of semantic annotation of documents has been widely recognized, the migration of documents from rendering-oriented formats, like PDF and HTML, toward XML has become an important research issue in different research communities [2; 3; 11; 12]. The majority of approaches either constrain the XML conversion to a domain specific problem, or make different kinds of assumptions about the structure of source and target documents. In [11], the source HTML documents are assumed to be dynamically generated through a form filling procedure, as in Web news portals, while a portal subject ontology permits the semantic annotation of the generated documents.

Transformation-based learning is used for automatic translation from HTML to XML in [3]. It assumes that source documents can be transformed into target XML documents through a series of proximity tag operations, including insert, replace, remove and swap. The translation model trains a set of transformation templates that minimizes an error driven evaluation function.

In document analysis research, Ishitani in [12] applies OCR-based techniques and the XY-cut algorithm in order to extract the logical structure from page images and to map it into a pivot XML structure. While logical structure extraction can be automated to a large extent, the mapping from the pivot XML to the target XML schema remains manual.

In natural language tasks, various information extraction methods often exploit the sequential nature of data to extract different entities and extend learning models with grammatical structures, like HMM [8] or undirected graphical models, like Conditional Random Fields [5]. Moreover, a hierarchy of HMMs is used in [13] to improve the accuracy of extracting specific classes of entities and relationships among entities. A hierarchical HMM uses multiple levels of states to describe the input on different level of granularity and achieve a richer representation of information in documents.

## 9 Conclusion

We propose a probabilistic method for the XML annotation of semi-structured documents. The tree annotation problem is reduced to the *generalized probabilistic context-free parsing* of an observation sequence. We determine the most probable tree annotation by maximizing the joint probability of selecting a terminal sequence for the observation sequence and the most probable parse for the selected terminal sequence.

We extend the inside-outside algorithm for probabilistic context-free grammars. We benefit from the available tree annotation that allows us to extend the inside function in a rigorous manner, and avoid the extension of the outside function which might require approximation.

The experimental results are promising. In future work, we plan to address different challenges in automating the HTML-to-XML conversion. We are particularly interested in extending the annotation model with the source tree structures that have been ignored so far.

## 10 Acknowledgement

This work is supported by VIKEF Integrated Project co-funded under the EU 6th Framework Programme.

## References

[1] A. L. Berger, S. Della Pietra, and V. J. Della Pietra. A maximum entropy approach to natural language processing. *Computational Linguistics*, 22(1):39–71, 1996.

[2] N. Sundaresan, Ch. Chung, M. Gertz. Reverse engineering for web data: From visual to semantic structures. In *Proc. 18th ICDE'02, San Jose, California*, 2002.

[3] J.R. Curran and R.K. Wong. Transformation-based learning for automatic translation from HTML to XML. In *Proc. 4th Australas. Doc. Computing Symp.*, 1999.

[4] K. Hall and M. Johnson. Language modeling using efficient best-first bottomup parsing. In *IEEE Autom. Speech Recogn. and Understanding Workshop*, 2003.

[5] J. Lafferty, A. McCallum, and F. Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proc. of 18th ICML'01, San Francisco, CA*, pages 282–289, 2001.

[6] K. Lari and S. J. Young. The estimation of stochastic context-free grammars using the inside-outside algorithm. *Computer Speech and Language*, 4:35–56, 1990.

[7] R. Malouf. A comparison of algorithms for maximum entropy parameter estimation. In *Proc. of 6th Conf. on Natural Language Learning*, pages 49–55, 2002.

[8] A. McCallum, D. Freitag, and F. Pereira. Maximum entropy Markov models for information extraction and segmentation. In *Proc. 17th ICML'00, San Francisco, CA*, pages 591–598, 2000.

[9] Frank Neven. Automata Theory for XML Researchers. *SIGMOD Record*, 31(3):39–46, 2002.

[10] Yannis Papakonstantinou and Victor Vianu. DTD Inference for Views of XML Data. In *Proc. of 19th ACM PODS, Dallas, Texas, USA*, pages 35–46, 2000.

[11] I.V. Ramakrishnan, S. Mukherjee, G. Yang. Automatic annotation of content-rich web documents: Structural and semantic analysis. In *Intern. Sem. Web Conf.*, 2003.

[12] Y. Ishitani. Document transformation system from papers to xml data based on pivot document method. In *Proc. of 7th ICDAR'03*, pages 250–255, 2003.

[13] M. Skounakis, M. Craven, and S. Ray. Hierarchical hidden markov models for information extraction. In *Proc. of 18th IJCAI, Acapulco, Mexico*, 2003.