A TREE LEARNING APPROACH TO WEB DOCUMENT SECTIONAL HIERARCHY EXTRACTION

F. Canan Pembe
Department of Computer Engineering, Boğaziçi University, Bebek, İstanbul, Turkey
Department of Computer Engineering, Istanbul Kultur University, Ataköy, İstanbul, Turkey
canan.pembe@boun.edu.tr

Tunga Gungör
Department of Computer Engineering, Boğaziçi University, Bebek, İstanbul, Turkey
gungori@boun.edu.tr

Keywords: Machine learning, document structure, World Wide Web, Hypertext Markup Language.

Abstract: There is an increasing availability of documents in electronic form due to the widespread use of the Internet. Hypertext Markup Language (HTML) which is mostly concerned with the presentation of documents is still the most commonly used format on the Web, despite the appearance of semantically richer markup languages such as XML. Effective processing of Web documents has several uses such as the display of content on small-screen devices and summarization. In this paper, we investigate the problem of identifying the sectional hierarchy of a given HTML document together with the headings in the document. We propose and evaluate a learning approach suitable to tree representation based on Support Vector Machines.

1 INTRODUCTION

The automatic processing of Web documents is becoming increasingly important as the number of such documents grows drastically on the Internet. HTML is the most commonly used document format on the Web; however, it is mainly concerned with the presentation of contents rather than a semantic representation. Web documents usually consist of several sections and subsections with various formatting. Automatic processing of the document structure can be useful for various applications including search engines and summarization.

In this paper, we consider the problem of automatically extracting the sectional hierarchy of a Web document. We developed a machine learning approach because it can be more flexible by combining several features using a corpus rather than predefined rules. In general, a document can be represented as a tree with order and containment relations between its physical and logical components (Mao et al., 2003). Therefore, a learning approach suitable to a tree representation is needed rather than the simpler case of classification. We developed an incremental algorithm making a sequence of locally optimal choices to approximate a globally optimal solution using Support Vector Machines. We evaluated the system based on the accuracy of heading and hierarchy extraction tasks.

2 RELATED WORK

In the literature, there are some studies on the extraction of the main title from documents in electronic form using machine learning (Xue et al., 2007). In another study, the aim is to segment a Web document into blocks using classification (Feng et al., 2005). For this purpose, a set of features is defined to represent the difference between each pair of text nodes; e.g. the difference in formatting of two nodes. The document is segmented in a flat way without considering the hierarchical structure.

In general, document structure analysis can be considered as a syntactic analysis problem (Mao et al., 2003). In one of the studies, transformation-based learning is used in the conversion of HTML documents into XML format (Curran and Wong, 1999). The logical structure of a document may also be represented with a generalized n-gram model (Brugger et al., 1997) or by means of a probabilistic grammar (Shilman et al., 2005). There is some related work on incremental parsing using machine learning (Collins and Roark, 2004). An application
of such an approach is to automatically generate a table-of-contents for a book (Branavan et al., 2007).

3 WEB DOCUMENT SECTIONAL HIERARCHY EXTRACTION

3.1 The Model

The general problem we consider may be defined as learning a mapping from inputs $x \in X$ to outputs $y \in Y$. In syntactic parsing, $X$ is a set of sentences and $Y$ is a set of possible parse trees (Collins and Roark, 2004). Analogously, we define $X$ as a set of documents and $Y$ as a set of possible sectional hierarchies using the following framework:

- Training examples $(x_i, y_i)$ for $i = 1 \ldots n$
- A function $\text{GEN}(x)$ which enumerates a set of possible outputs for an input $x$
- A representation $\Phi$ mapping each $(x_i, y_i) \in X \times Y$ to a feature vector $\Phi(x_i, y_i)$
- A parameter vector $\alpha$

The learning task is to estimate the parameter vector $\alpha$ using the training examples such that it will give highest scores to correct outputs:

$$F(x) = \arg \max_{y \in \text{GEN}(x)} \Phi(x, y) \cdot \alpha$$  \hspace{1cm} (1)

In the proposed system, each document is modeled as a sequence of text units. We define a text unit $u_i$ as a text fragment delimited by a new line (Figure 1). The output of the proposed system is the sectional hierarchy where headings and subheadings are at the intermediate nodes and other text units are at the leaves (Figure 2). A graph corresponding to an example hierarchy is given in Figure 3. The nodes are arranged from left to right according to their order of appearance in the document.

3.2 The Learning Approach

The training set consists of $(x_i, y_i)$ pairs where $x_i$ is $i^{th}$ document and $y_i$ is the golden standard hierarchy (i.e. tree) for that document. For each document in the set, the learning algorithm (Figure 4) works on the units one by one and considers the attachment of a unit to its parent unit as a positive example (regular lines in Figure 3). The negative examples are the potential attachments not realized in the golden standard hierarchy (dashed lines in Figure 3). Here, two constraints due to the document flow are applied. First, a unit cannot be attached to a heading unit coming after it in the document order. Second, the connections cannot cross each other according to the projectivity rule (Covington, 2001).
Each unit is associated with the following features:

(1) **Formatting features**: These are related to the formatting of the unit; e.g. font size and boldness.

(2) **DOM tree features**: These features are related to the DOM tree of the document (W3C, 2005); e.g. the DOM address of the unit (Feng et al., 2005).

(3) **Content features**: These features specify whether a unit contains certain cue words or phrases; e.g. “back to top”, “login”, etc.

(4) **Other features**: These include the number of characters and the punctuation mark of the unit.

We utilize contextual information in the document hierarchy. We use \( u_i \) to denote the unit with \( l \) levels above a unit \( u \) and \( j \) units to its left; e.g., \( u_{10} \) denotes the parent and \( u_{01} \) denotes the preceding sibling. We define **composite features** of two units \( u \) and \( u_j (F_i) \) as the difference of their features. Such information can be useful in determining the sectional hierarchy. For example, a heading unit (i.e. parent) is usually more emphasized than the underlying text unit in terms of formatting. Similarly, units under the same heading, i.e. siblings, usually have similar formatting. Finally, we define **global features** such as the depth of the tree.

### 3.3 Features

In testing, we adopted an incremental approach similar to a previous work on syntactic parsing (Collins and Roark, 2004). Initially, the set of partial trees is empty. Whenever the next unit in the document is processed, its potential attachments to the partial trees are considered and the set is updated (ADVANCE). Restrictions due to the document flow are also applied. To prevent the exponential growth of the set, we maintain only the top \( k \) (i.e. beam width) highest scored trees at each step (FILTER). In the pre-processing, each heading \( u_i \) is connected with \( u_{i+1} \) because a heading is always followed by a child unit in the hierarchy. Also, a unit can only attach to a heading. We developed the following modifications of the algorithm:

**M1**: The score obtained from the SVM model is converted into a probability value using a sigmoid function (Platt, 1999; Mayfield et al., 2003). We use \( parent(i)=j \) to denote that there is a parent-child dependency between the units \( i \) and \( j \). If we assume that the probabilities of such dependencies are mutually independent, the probability of building a hierarchy with \( n \) units is:

\[
\prod_{i=1}^{n} P(parent(i) = j)
\]

**M2**: The testing algorithm is run in two levels. The first level runs on only heading units and the second level runs on non-heading units.

**M3 - M4**: The partial trees are given ranks based on the output of SVM. In M3, the times a tree has obtained rank “1” are summed, whereas in M4, the ranks at each step are summed to obtain the score.

## 4 EVALUATION

We selected 12 TREC queries (TREC, 2004) and built a corpus with the top 25 results of Google for each query. In Table 1, some statistics are given for the corpus. We implemented the system using the GATE framework (GATE, 2009), Cobra Java HTML Renderer and Parser Toolkit (The Lobo Project, 2009) and SVM-light (Joachims, 1999).

First, we evaluated the classification of units as heading or not (Model 1). We used five different feature sets (Table 2) including features of the current unit \( (F_v) \) and neighbouring units. The most accurate results were obtained when all the features were included. We evaluated the effect of different kernel types for SVM on that feature set (Table 3).

The second model is the tree learning approach for sectional hierarchy extraction (Model 2). We used different feature sets, including composite features of the current unit with candidate parent \( (F_{10}) \), sibling \( (F_{01}, F_{00}) \) and grandparent \( (F_{20}) \) as in Table 4. The accuracy is defined as the ratio of correctly identified parent-child relationships to the total number of such relationships in the hierarchy. We obtained the best accuracy when the feature set \( F_{10}, F_{01} \) was used. In Table 5, the results for the main algorithm (M0) and the modifications (M1 to M4) are given for polynomial kernel and a beam width of 100. We compared the effect of using the headings identified with Model 1 and using manually marked headings. The best accuracy result was obtained (0.76) when manually marked headings were used.
information can be used as features to improve the accuracy of the system. Also, the corpus of Web documents will be extended with additional queries.

ACKNOWLEDGMENTS

This work was supported by the Bogazici University Research Fund, Grant No. 07A106 and Istanbul Kultür University.

REFERENCES