Web-Scale Table Census and Classification

Eric Crestan
Yahoo! Labs
701 First Avenue
Sunnyvale, CA 94089, USA
ecrestan@yahoo.fr

Patrick Pantel
Microsoft Research
One Microsoft Way
Redmond, WA 98052, USA
ppantel@microsoft.com

ABSTRACT
We report on a census of the types of HTML tables on the Web according to a fine-grained classification taxonomy describing the semantics that they express. For each relational table type, we describe open challenges for extracting from them semantic triples, i.e., knowledge. We also present TabEx, a supervised framework for web-scale HTML table classification and apply it to the task of classifying HTML tables into our taxonomy. We show empirical evidence, through a large-scale experimental analysis over a crawl of the Web, that classification accuracy significantly outperforms several baselines. We present a detailed feature analysis and outline the most salient features for each table type.

Categories and Subject Descriptors
H.3.1 [**Information Storage and Retrieval**]: Content Analysis and Indexing; I.2.6 [**Artificial Intelligence**]: Learning – *knowledge acquisition*.

General Terms
Algorithms, Experimentation, Measurement.

Keywords
Information extraction, structured data, web tables, classification.

1. INTRODUCTION
A wealth of knowledge is encoded in the form of tables on the World Wide Web. Mining this knowledge has the potential to power many applications such as query expansion [7] and textual advertising [13]. Recent efforts have focused on teasing apart tables consisting of relational information from those used strictly for multi-column layouts and formatting [13], and other efforts on extracting schemas and knowledge in the form of relational tuples [1][4][5][6][10][14].

Relational tables considered in this paper encode facts, or semantic triples of the form ⟨*p*, *s*, *o*⟩, where *p* is a predicate or relation, *s* is the subject of the predicate and *o* is its object. These tables may be rendered in many different ways as illustrated in Figure 1. Various triples may be extracted from the tables, such as for example ⟨*Price*, *Angels & Demons DVD*, *$22.99*⟩ and ⟨*List Price*, *Angels & Demons Blue-ray*, *$39.95*⟩.

WSDM'11, February 9–12, 2011, Hong Kong, China.

Each relational table consists of a schema, and each table type poses different challenges for extracting the schema and the encoded knowledge (semantic triples). For example, in order to extract the triple ⟨*Price*, *Angels & Demons DVD*, *$22.99*⟩ from Figure 1, a system would first have to recognize that the first column consists of attributes (i.e., the *predicates*) and that the second column consists of the values of these attributes (i.e., the *objects*). The subject of the predicates, "*Angels & Demons DVD*", is completely outside of the table and may only be found in the title of the page or in some anchor text linking to the document.

Although some table types have been identified in the literature (e.g., see [14]), there is a pressing need for characterizing all the types on the Web, for estimating their relative importance, and for ultimately extracting the knowledge contained in these tables.

In this paper, we propose a table type taxonomy, discovered empirically through a large-scale census of tables on a large crawl of the Web. We present the relative frequencies of these tables and show that the most frequent table type, *attribute/value* tables, has been mostly ignored in the extraction literature. Extracting the actual semantic triples is outside of the scope of this paper. However, for each table type that encodes relational knowledge, we discuss the open challenges for extracting from them schemas and semantic triples, in the hope of spawning new research directions for extraction. We then propose machine learning algorithms for classifying all the tables on the web according to the taxonomy, leveraging an extensive set of rich layout features, structural content features, and lexical features. Finally, we empirically show, by means of large-scale classification experimentations, that overall classification accuracy is 75.2%.

The main contributions of this paper are summarized as follows:

- We propose a fine-grained table type taxonomy and their relative frequencies through a census of HTML tables on a large crawl of the Web;
- We outline the main open challenges for extracting semantic triples from each table type;
- We propose a supervised classification model and a rich set of features for automatically classifying Web tables into our table type taxonomy;
- We report on a large empirical study showing that our classification model significantly outperforms several baselines.

The remainder of this paper is organized as follows. In the next section, we present related work in table identification and knowledge extraction. Then, in Section 3, we present our table type taxonomy and describe open challenges for extracting from them semantic triples. Section 4 outlines our supervised classification model and our rich set of features, and Section 5 reports our experimental results. We finally conclude in Section 6.

Figure 1. Example webpage containing multiple table types.

2. RELATED WORK

Web table understanding and extraction has received a lot of interest of late, with most work since the turn of the century (coinciding with the advent of more and more data available on the Internet). There have been very few attempts to characterize the different types of tables on the web. Some researchers have focused on discovering specific types of tables such as listings [1][2], while others have considered coarse-grained classes such as genuine (i.e., relational tables) vs. non-genuine (i.e., formatting tables) [13]. An exception could be found in the work of [14] where they proposed nine types of tables, all of which focus on tables containing attributes and values. We argue in this paper that these types can be collapsed into two classes (vertical listings and horizontal listings), where the other types only capture structural differences between these two classes.

More recently, Cafarella et al. [2] extracted *relational* tables (i.e., tables containing semantic triples) from a very large Web corpus. While filtering out non-relational tables, they listed other frequent types of tables such as *form* and *calendar*. In addition, they also discarded the tables that "*are really simple lists presented in two dimensions*", what we call in this paper *attribute/value* tables, which we show to be the most frequent *relational* table type on the Web. As well as giving some idea of the existing table types, this previous work also gives an idea on the distribution of some of those types, reporting 1.34% of the tables are used as *forms* and 0.04% as *calendars*. In this paper, we conduct a large census of tables on the Web and show that among all our table types, *forms* occur 4.8% of the time, and *calendars* occur 0.4% of the time.

Several methods have been proposed in the past for table classification, but all of them consider it as a binary problem (genuine vs. non-genuine, relational vs. non-relational). Wang and Hu [13] experimented with decision trees and SVMs for separating genuine tables from the non-genuine tables and reported a similar F-measure for the two approaches of 95.9%. An extensive list of features was used tackling both table layout and content consistency. The classification quality is not comparable to what could be achieved on a Web-scale because of the way they carried their data collection. A set of 2851 pages were harvested from Google directory and News using predefined keywords known to have a higher chance to recall *genuine* tables (e.g. *table*, *stocks*, *bonds*…). In the same spirit, Cafarella et al. [2] used a similar machine learning approach in order to filter out the non-relational tables as well as for detecting table headers. They used a similar set of features as Wang and Hu [13]. For the task of filtering out non-relational tables, they report an F-measure of 61% which highlights the difficulty introduced by a web-scale application to table classification when compared with [14]. In this paper, we propose a much finer-grained table-type classification over a large crawl of the Web and report an overall accuracy of 75.2%.

3. WEB OF TABLES: A CENSUS

Tables on the Web are mostly used to visually organize elements on a user's screen, however some also consist of relational knowledge. In this section, we propose a table type taxonomy developed by studying a large crawl of the Web. We identify those types which contain relational knowledge, we quantify their frequencies on the Web, and we describe challenges for extracting semantic triples from this knowledge.

3.1 Extracting HTML Tables

In order to develop the table type taxonomy, we investigate a sample of the HTML tables over a large crawl of 1.2 billion high quality English pages on the Web[1]. We applied a simple parser in order to extract all the *inner tables*, which are the HTML tables that do not contain another table among its cells. Our assumption is that other tables are only used for layout purposes. This filter rejects 30% of the original 12 billion tables, leaving 8.2 billion *inner tables* for our study. Below are some statistics describing our extraction:

- Total documents: **1.2B**
- Hosts with tables: **37M**
- Documents with tables: **896M**
- Tables: **8.2B**
- Unique tables: **2.6B**

From the initial crawl, 75% of the pages contain at least one table with an average of 9.1 tables per document. This shows how extensively tables are used on the web. However, a large percentage of them is used for layout and formatting purposes and is of little interest to us for extracting semantic triples.

3.2 Table Type Taxonomy

The genuineness[2] of tables depends largely on its structure and the semantics it carries. In order to better understand table structure and content, we manually inspected a large sample of our tables and propose a table type taxonomy separating them by whether they contain relational knowledge or whether they are used for layout purposes. Below, we describe our proposed taxonomy, illustrated in Figure 2.

3.2.1 Relational Knowledge Tables

In this section we describe types of tables structuring pieces of knowledge. For each table type, we describe the challenges for extracting from them semantic triples of the form $\langle p, s, o \rangle$, where p is a predicate, s is the subject of the predicate and o is its object. Our goal is not to solve all of these extraction problems in this paper, it is only to identify open research problems.

[1] Quality is ensured thanks to a spam page classifier discarding any spam content and a page quality score, similar to PageRank, computed over a much larger set of pages.

[2] Genuineness was a term introduced in [12] to denote tables containing relational knowledge.

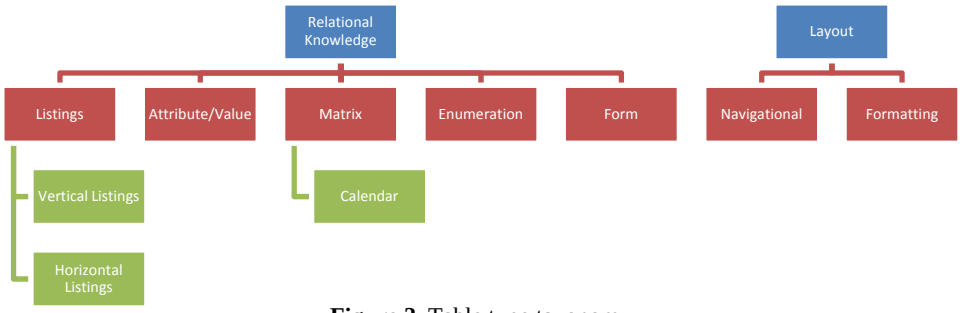

Figure 2. Table type taxonomy.

3.2.1.1 *Vertical Listings*

These tables list one or more attributes for a series of *similar* entities. For example, it could be a table where the capital and population for several countries (one per line) are listed.

The biggest challenge in extracting semantic triples from vertical listings is the identification of which column lists the *subject* and which columns list the *predicates*. Often, the subject is the first column, but header columns or reordering are possible. The *objects* of the triples may be extracted directly from the cell values[3].

3.2.1.2 *Horizontal Listings*

Similar to VERTICAL LISTINGS, horizontal listings present their subjects in one row and their predicates in rows. These tables are often used in order to compare items (e.g., comparisons of product features) or to list elements with additional information (e.g., a list of conference participants with affiliations). An example could be a table comparing the *predicates* "Resolution", "Sensor", etc., for two subjects, digital cameras "Nikon D80" and "Canon Digital Rebel XTi/400D".

Identifying the row containing the *subjects* and *predicates* pose the biggest extraction challenge from this table type. As for VERTICAL LISTINGS, the *objects* of triples may be extracted directly from the cell values3.

3.2.1.3 *Attribute/Value*

ATTRIBUTE/VALUE tables are a specific case of VERTICAL LISTINGS and HORIZONTAL LISTINGS. The distinguishing factor is that they often do not contain the *subjects* in the table. ATTRIBUTE/VALUE tables are often used as factual sheets about an entity. For example: A table containing specifications of a digital camera, where the table does not contain the actual camera name. In this case, the whole webpage was *about* the digital camera Nikon D5000 so it was not necessary to repeat it in the table.

It follows that the biggest challenge in extracting semantic triples from ATTRIBUTE/VALUE tables lies in the detection of the *subject* of the table (normally, all triples extracted from an ATTRIBUTE/VALUE table contain the same *subject*.) We call this particular open research problem *Protagonist Detection*. We show later in this section that this table type is one of the most common relational tables on the Web.

[3] Beyond the discovery of which columns and rows list *subjects*, *attributes*, and *objects*, there are significant challenges in normalizing objects, finding canonical forms for subjects and objects, and fusing triples across tables. This paper does not address the extraction of semantic triples beyond identifying the structural location of the triples.

3.2.1.4 *Matrix*

MATRIX tables have the same value type for each cell at the junction of a row and a column. The row and column headers are normally the *subject*, but the *object* is only obtained by combining the 2 *subjects* together. The *predication* is often not contained explicitly in the table. For example, a table giving the number of traffic accidents per month (rows) and per state (columns) is of type matrix.

3.2.1.5 *Calendar*

This table type is a specific case of the MATRIX type, differing only in its semantics. In CALENDAR tables, the *subject* of the triples is a *date* and the *predicates* are either a generic semantic relation such as "*occurs-on*" or relations such as "*performance-date*" or "*lunch-special*". For example: a calendar listing the performances at Los Angeles concert venues. In this case the *predicate* would be "*performance-date*" and the *objects* are the values of the cells. The hardest extraction challenge from CALENDARS lies in the discovery of the *predicates*, which oftentimes are not explicitly stated in the table.

3.2.1.6 *Enumeration*

ENUMERATION tables list a series of *objects* that have the same ontological relation (e.g., *hyponomys*, *meronymys*, or *siblings*). A table enumerating a list of alphabetic letters or a list of camera models would be considered as ENUMERATION. For example: an enumeration of U.S. States and Territories.

The major extraction challenge for ENUMERATION tables is the discovery of the *predicate*. Oftentimes, it is not explicitly listed and may not even occur anywhere on the web page. In the example, the *predicate* "*is-a*" (or *hyponym-of*) may not be listed anywhere on the page from where it came. Using corpus count statistics of patterns such as "*X is a Y*" and "*X* is part of *Y*" between cell values may give insights into the predicate [11]. The *subjects* of the semantic triples are the cell values and the *object* is oftentimes the header row. Similar to ATTRIBUTE/VALUE tables, ENUMERATION tables don't always explicitly state the *object* in the table (e.g., a webpage titled U.S. States may just contain a table listing them without explicitly stating in the table that they are U.S. States.) This poses another significant extraction challenge.

3.2.1.7 *Form*

Similar to ATTRIBUTE/VALUE tables, FORM tables are composed of input fields for the user to fill or select. A typical example of this type is a user/password input form. The input fields are the missing values of an ATTRIBUTE/VALUE table meant to be filled by the user. For example: a FORM table requesting contact information.

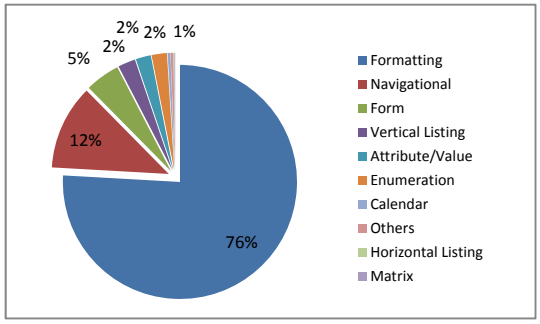

Figure 3. Table type distribution on a large crawl of the Web.

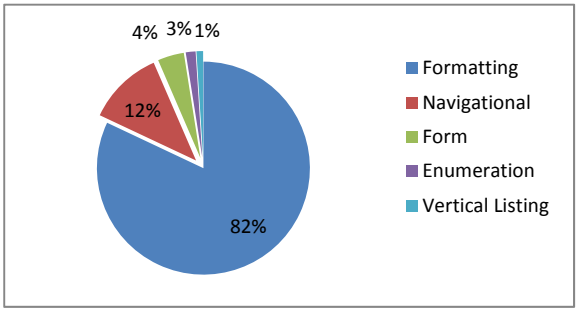

Figure 4. Table type distribution of tables removed during the filtering step of Section 3.4.

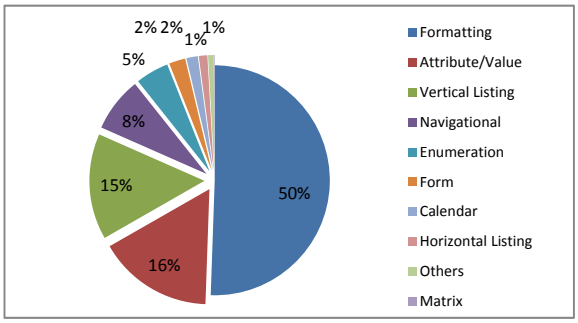

Figure 5. Table type distribution of remaining tables after applying the filter from Section 3.4.

There is typically no *subject* for semantic triples in FORM tables. The *predicates* are most often form labels and the *objects* are what is requested from the user.

3.2.1.8 Other

In some cases, there is an underlying relation between the elements in a table, but it doesn't fit within the previously described types. Since those cases are very rare (a total of 2% of tables for all of them), we do not present them in this paper. An example is a table presenting the score for each period of two basketball teams for a given game, which cannot be classified as VERTICAL LISTING since a game is a pair-wise relationship and another team cannot be added to the same table.

3.2.2 Layout Tables

We now present two table types that do not contain any knowledge (i.e., non-relational tables) and are used strictly for layout purposes.

3.2.2.1 Navigational

These tables are composed of cells organized for navigational purpose (e.g., the product categories on an online shopping site such as Amazon.com). There are no clear relations between the cells, except that for navigating within or outside of the site. While ENUMERATION could also be considered NAVIGATIONAL, we distinguish it since semantic triples (knowledge) may be extracted.

3.2.2.2 Formatting

This table type accounts for a large portion of the tables on the web where the only purpose is to organize visually some elements. It could be used for laying out pictures with text or formatting a webpage. FORMATTING could also be seen as the default type of tables where nothing is of interest for a knowledge extraction task.

3.3 Table Type Distribution

In this section, we estimate the proportion of each table type of our taxonomy, presented in the previous section, in order to understand how tables are used on the Web. We randomly sampled 5000 tables from our 8.2 billion tables described in Section 3.1. For each table, we asked a paid human editor to classify it according to the taxonomy presented in Figure 2. The resulting table type distribution is illustrated in Figure 3.

Not surprisingly, the vast majority of tables are used for layout purposes (i.e., non-relational tables). FORMATTING and NAVIGATIONAL make for 88% of the annotated sample. Comparing with the corpus estimates presented in [1] on *relational* tables, which they define as consisting only of VERTICAL LISTING and HORIZONTAL LISTING, we capitalize a 2.5% tables. This discrepancy might be due to differences in the interpretation of such *relational* tables by the editors. For example, our definition of VERTICAL LISTING includes tables listing blog posts with an associated date or a list of tracks on a CD with track numbers (ignored in [1]). Moreover, while the ATTRIBUTE/VALUE table type was discarded in their work as being non-relational tables, we argue in this paper that they should be considered as single row relational tables (see Section 3.2.1.3). While some might contain uninteresting information such as user profiles, others contain useful knowledge such as the specification of a *digital camera*. This table type represents more than 2% of the tables found on the Web.

In the following section, we propose a very simple filter for eliminating many of the layout tables, losing very few knowledge-rich relational tables. This is important to reduce the amount of data to process downstream.

3.4 Filtering

Although *Layout* tables such as FORMATTING and NAVIGATIONAL may be useful for page structure analysis and segmentation, they are of no interest for information extraction engines. Similar to [1][12], we filter out *Layout* tables by ensuring that accepted tables adhere to the following conditions:

- Minimum of 2 rows
- Minimum of 2 columns
- No cell with more than 100 characters in it.

Applying these simple rules reduces drastically the numbers of tables to process by a factor of more than 80% to 1.3 billion.

In order to estimate the impact of this filter on extraction recall, we annotated a random sample of 200 eliminated tables. The results are presented in Figure 4.

Applying this filter eliminates mainly *Layout* tables (93% for both FORMATTING and NAVIGATIONAL). The most common false negative is the table type FORM, which in itself contains little knowledge without first being filled in by a user.

The filter also reduces the number of pages carrying HTML tables. The average number of tables per document drops from more that 9 to 2.7. About 45% of the original documents contained only discarded tables.

Among the 1.3B remaining tables (representing 560M unique tables), a sample of 5000 was judged again according to the table type taxonomy in Figure 2. The results are illustrated in Figure 5.

Now, only half the tables were judged as FORMATTING. The two next most frequent table types are knowledge-rich: ATTRIBUTE/VALUE at 16% and VERTICAL LISTING at 15%.

4. FINE-GRAINED TABLE TYPE CLASSIFICATION

4.1 Modeling Approach

Our classification approach adopts a supervised machine learning model. Specifically, we use a *Gradient Boosted Decision Tree* classification model - GBDT [8], which consists of an ensemble of decision trees for each class, fitted in a forward step-wise manner to current residuals. Friedman [8] shows that by drastically easing the problem of overfitting on training data (which is common in boosting algorithms), GDBT competes with state-of-the-art machine learning algorithms such as SVM [9] with much smaller resulting models and faster decoding time, both critical for our production system. The model is trained on a manually annotated random sample of entities taken from the full list of tables, using the features described in the next section.

4.2 Features Classes

In previous work, the problem of table type classification has been mostly tackled as a binary classification problem. The goal was to separate *genuine* vs. *non-genuine* tables [Wang and Hu] or *relational* vs. *non-relational tables* [Cafarella et al.][4]. Since we are proposing a much finer grained classification, our model requires a much larger and refined set of features in order to distinguish between the different table types. Also, the deeper classification problem makes direct comparison over state of the art difficult.

Instead of only considering global features for the table as a whole, each (non-global) feature was extracted per row and per column. The intuition is that some table types will present a high consistency in one of those dimensions. For example, VERTICAL LISTINGS are likely to have consistent values in columns, while variance should be higher on the rows. However, since table size is variable, it was not possible to generate features for every row and column. This would have caused a training data sparsity problem for high dimensional tables and could result in thousands of features to account for in the model. Instead, features where only generated for the two first rows and columns, as well as the last row and column. When tables only contain 2 rows (or columns), the last row (or column) is equal to the second one, which ensures a denser feature space.

Features are grouped into three distinct classes. The *global layout feature* class is the only one accounting for the structure of the table as a whole. The two remaining classes, *layout features* and *content features*, are generated over rows and columns. Below we describe each feature class and list the individual features.

4.2.1 Global Layout Features

Below we list *global* features that capture structural statistics about the table:

- ***max_rows***: Maximum number of rows for each column.
- ***max_cols***: Maximum number of columns for each row.
- ***max_cell_length***: Maximum cell content length in characters (this exclude any *html* tags it could contain).

Note that these features are also used in the filtering step in the first phase of the extraction (see Section 3.4).

4.2.2 Layout Features

Layout features are applied per column and per row. They are solely based on the size of the cells and their variance.

- ***avg_length***: Average length of cell cleaned of tags along a column or a row.
- ***length_variance***: Variance in cell length for a column or a row.
- ***ratio_colspan***: Ratio of cells in a column or a row generated by a colspan attribute. For example, if a row has 2 cells and one of those has a colspan=4 (illustrated in the table below), the ratio_colspan would be equal to 3/5=0.6, meaning that 3 of the columns displayed are generated by a *colspan.*

col#1	col#2 colspan=4	*span*	*span*	*span*

- ***ratio_rowspan***: Ratio of cells in a column or a row generated by a rowspan attribute.

4.2.3 Content Features

The following set of features focus on cell content. Two subdivisions can be distinguished based on whether the feature involves html tags (*html feature*s) of textual content (*lexical features*).

4.2.3.1 Html features

These features focus on the html tags contained in cells.

- ***dist_tags***: Ratio of distinct tags in the row/column. After removing tag attributes and all text, this feature assesses how different the cell tag structure is for a given row/column.
- ***ratio_th***: Ratio of cells containing table header <*th*> tags. Table header tags are more likely to be used in tables containing structured data.
- ***ratio_anchor***: Ratio of cells containing an anchor text. This is a useful feature in order to identify tables used for NAVIGATIONAL purposes.
- ***ratio_img***: Ratio of cells containing an image tag <*img*>. Tables containing several images are more likely to be FORMATTING tables.
- ***ratio_input***: Ratio of cells containing an <*input*> tag. This feature is a good clue in order to identify the FORM table type.

[4] *Genuineness* and *relational* referring to the same concept as *relational tables* defined in Section 1.

Table 1. Classification performance of TabEx on each table type compared with various baselines.

	Formatting			Navigational			Attribute/Value			Vertical Listing			Horizontal Listing		
	P	R	F	P	R	F	P	R	F	P	R	F	P	R	F
Baseline	-	-	-	-	-	-	0.457	0.562	0.510	-	-	-	-	-	-
Global Features	0.597	0.842	0.720	0.300	0.045	0.173	0.390	0.267	0.329	0.397	0.273	0.335	-	-	-
Layout Features	0.781	0.866	0.824	0.456	0.216	0.336	0.643	0.661	0.652	0.598	0.582	0.590	0.231	0.088	0.160
Html Features	0.804	0.841	0.823	0.474	0.282	0.378	0.618	0.704	0.661	0.620	0.543	0.582	0.265	0.097	0.181
Lexical Features	0.810	0.883	0.847	0.473	0.269	0.371	0.777	0.721	0.749	0.691	0.694	0.693	0.287	0.132	0.210
TabEx	0.836	0.873	**0.855**	0.545	0.353	**0.449**	0.767	0.764	**0.766**	0.717	0.720	**0.719**	0.357	0.126	**0.242**

	Enumeration			Calendar			Matrix			Form			Others		
	P	R	F	P	R	F	P	R	F	P	R	F	P	R	F
Baseline	-	-	-	-	-	-	-	-	-	0.2902	0.9912	0.6407	-	-	-
Global Features	0.125	0.016	0.071	0.586	0.806	0.696	0.333	0.333	0.333	0.167	0.019	0.093	0.050	0.050	0.050
Layout Features	0.319	0.164	0.242	0.919	0.888	0.904	0.500	1.000	**0.750**	0.362	0.116	0.239	0.431	0.225	0.328
Html Features	0.456	0.281	0.369	0.709	0.697	0.703	0.500	0.500	0.500	0.680	0.630	0.655	0.500	0.201	0.351
Lexical Features	0.419	0.266	0.343	0.842	0.872	0.857	0.667	0.667	0.667	0.513	0.265	0.389	0.625	0.399	**0.512**
TabEx	0.441	0.302	**0.372**	0.941	0.882	**0.912**	0.200	0.400	0.300	0.781	0.567	**0.674**	0.639	0.319	0.479

- ***ratio_select***: Ratio of cells containing a drop down *<select>* tag. Like the previous feature, this is a good clue in order to identify the FORM table type.
- ***ratio_f***: Ratio of cells containing a font change. This includes *<b>*, *<u>*, *<font>* and *<i>* html tags.
- ***ratio_br***: Ratio of cells containing a line break tag *
*.

4.2.3.2 Lexical features

Below we list features capturing textual patterns within the cells.

- ***dist_string***: Ratio of distinct strings in the row/column. After replacing all digits by the character "#", this feature assesses how different the cell content is for a given row/column.
- ***ratio_colon***: Ratio of cells ending with the colon character. This feature is a useful indicator for the ATTRIBUTE/VALUE table type.
- ***ratio_contain_number***: Ratio of cells containing a digit. This feature is a good indication for columns or rows listing measures (e.g. *temperature*, *area*…).
- ***ratio_is_number***: Ratio of cells where the content is a number. A high ratio is likely to indicate a VERTICAL or HORIZONTAL LISTING.
- ***ratio_nonempty***: Ratio of non-empty cells.

5. EXPERIMENTS ON TABLE TYPE CLASSIFICATION

Depending on the type of table, the knowledge it contains will be structured in different ways. An accurate table type classification is therefore necessary in order to extract knowledge from these tables.

5.1 Setup and baselines

Section 3.1 describes our Web crawl and our extraction methodology of obtaining all the tables from this crawl. After applying the filtering algorithm from Section 3.4, we were left with 1.3 billion tables. We randomly sampled 5000 tables from these and had 10 paid editors classify them into our table type taxonomy illustrated in Figure 2. After the first pass of annotation, two separate paid expert editors went over and adjudicated the tables with disagreements (slightly over 10% of the overall annotations). This results in a high quality test data set, forming the base of our experiments reported in this section.

Prior art mostly focused on a very coarse-grained table classification between two classes: genuine (those containing relational knowledge) and non-genuine (those tables used for layout purposes). In Section 5.5, we specifically discuss a comparison of our work with prior art. For our classification analysis on the full table type taxonomy presented in this paper, we use the following heuristic baselines:

- FORM: The most singular elements for this table type are *input* and *select* html tags. If we find these tags then we classify the table as FORM.
- ATTRIBUTE/VALUE: After excluding tables matching the previous heuristic, this table type typically contains a colon at the end of attribute cells. This heuristic classifies tables as ATTRIBUTE/VALUE if we find colons in each cell of a column.

For the other table types, there is no obvious single feature that can be used as evidence for creating a heuristic. For comparisons on these types, we defined several baseline versions of our classifier using the different feature families presented in Section 4.2, separately.

5.2 Classification Analysis

20-fold cross-validation was used in order to evaluate each system and measure statistical significance. We randomly generated 20 non-overlapping test sets composed of 250 examples each, while the remaining 4750 examples are used for training purposes. This setting allows computing the confidence bounds for each table type classification. GBDT model parameters were set globally on a separate held-out development set and are as follows:

- *Number of trees* = 100: The number of decision trees in the boosting procedure;
- *Minimum number of samples per node* = 2: The minimum number of training samples per node;
- *Best-first search nodes* = 6: The number of nodes for best-first search in tree growing.

For each system, we report 3 measures: *precision* (P), *recall* (R) and *F-measure* (F). The results reported in Table 1 are an average over the 20 runs for each table type. They correspond to the maximum recall point over all classes (every test example gets classified). The overall TABEX accuracy was 75.2%. Note that a comparison against prior art is discussed later in Section 5.5.

The first observation regarding the two heuristic-based baselines is that FORM can be identified with very high recall using a simple

Table 2. Top-5 most important features per table type.

Formatting		Navigational		Attribute/Value		Vertical Listing		Horizontal Listing	
C$_RATIO_COLSPAN	100	*C$_RATIO_ANCHOR*	100	R2_RATIO_COLON	100	R2_DIST_STRING	100	MAX_COLS	100
C1_RATIO_NONEMPTY	84.1	*C2_RATIO_ANCHOR*	66.6	C1_LENGTH_VARIANCE	59.6	C1_RATIO_NONEMPTY	62.2	R1_DIST_STRING	59.9
C1_DIST_TAGS	82.3	C1_RATIO_NONEMPTY	39.5	C1_RATIO_NONEMPTY	41.6	MAX_COLS	42.6	R2_RATIO_CONTAIN_NUMBER	36.1
C$_AVG_LENGTH	59.9	C2_LENGTH_VARIANCE	18.9	C2_DIST_STRING	30.3	R2_RATIO_COLON	29.0	*R2_RATIO_F*	12.6
C$_DIST_STRING	46.6	MAX_CELL_LENGTH	16.9	C1_RATIO_COLON	26.2	R$_DIST_STRING	27.8	R1_LENGTH_VARIANCE	9.3

Enumeration		Calendar		Matrix		Form		Others	
C2_RATIO_ANCHOR	100	MAX_COLS	100	MAX_COLS	100	*R2_RATIO_INPUT*	100	C2_RATIO_IS_NUMBER	100
C2_DIST_STRING	58.1	C2_RATIO_CONTAIN_NUMBER	12.5	*R2_RATIO_TH*	30.9	*C1_RATIO_INPUT*	67.2	R1_RATIO_IS_NUMBER	61.7
C1_RATIO_ANCHOR	20.8	C2_RATIO_IS_NUMBER	12.0	R$_RATIO_IS_NUMBER	18.5	*C2_RATIO_INPUT*	37.8	MAX_COLS	43.3
R$_LENGTH_VARIANCE	19.1	C1_RATIO_IS_NUMBER	7.5	*C1_RATIO_TH*	8.9	*R2_RATIO_ANCHOR*	14.9	C1_LENGTH_VARIANCE	16.7
C$_DIST_STRING	17.8	C$_DIST_STRING	6.3	*R$_RATIO_F*	3.1	*C1_RATIO_SELECT*	12.6	C2_AVG_LENGTH	8.7

rule based on <*input*> and <*select*> tags. However, this heuristic lacks precision (only 29%).

From the results obtained using only the *Global Features*, the lack of modeling power is clearly exposed. Nevertheless, the CALENDAR type seems to benefit from such features with almost 70% F-measure. This observation follows the intuition that calendars should have a pretty standard number of columns and rows (representing the days of the week and the number of weeks in a month). Using only the *Layout Features* improves greatly over the simpler *Global Features* with the most dramatic improvement for the MATRIX type. However, because of the low representation of the MATRIX type in the full set, the improvements are not statistically significant.

The two baselines using only *Html Features* or *Lexical Features* show similar performance for most of the classes. As an exception, using *Html Features* improves dramatically the classification quality over the other feature sets for the class FORM. On the other hand, ATTRIBUTE/VALUE and VERTICAL LISTING types benefit the most from *Lexical Features*. This observation follows the intuition that those tables are supposed to contain knowledge offering in most of the case certain regularity in its content (e.g. columns of numbers or columns containing colon characters).

Finally, TABEX, our system using all the previously presented features performs the best overall in F-measure. Higher scores achieved using *Layout Features* only for MATRIX and the one for OTHERS using *Lexical Features* are not statistically significant because there are too few examples of each class (respectively 7 and 42). But all other results are statistically significant over the baseline, with 95% confidence estimated by 20-fold cross-validation.

For four of the most preponderant table types in the corpus, Figure 6 illustrates the classifier precision variation over the classification threshold. We observed that classification quality for ATTRIBUTE/VALUE and FORM reached nearly 90% precision for a recall of 50%.

While classification of the ATTRIBUTE/VALUE type looks solid, the VERTICAL LISTING type falls behind by an average of almost 10% at the same recall point. There is in fact quite a lot of overlap between both table types in term of structure and content. This is generating some confusion for the classifier (shown in Table 3). Moreover, several tables in the training set are borderline cases and offer clues from several types at the same time. We will discuss this in more detail in Section 5.4.

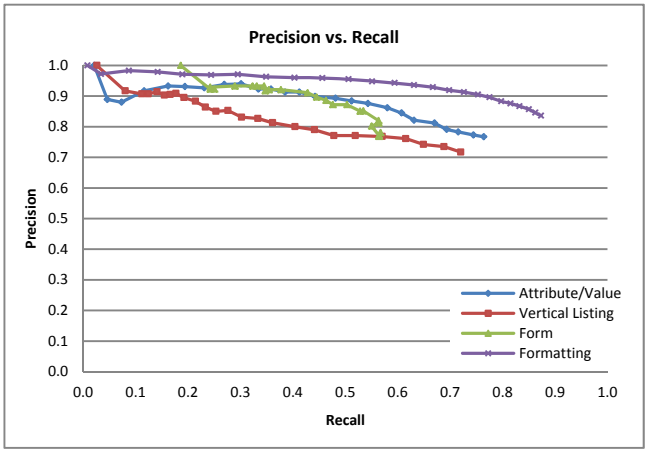

Figure 6. Precision vs. Recall graphs for four table types: ATTRIBUTE/VALUE, VERTICAL LISTING, FORM, and FORMATTING.

5.3 Feature Importance

The features proposed in Section 4.2 were chosen based on our intuition and their perceived ability to help differentiating one table type against another. Using GBDT as our machine learning model for table type classification, we have the added benefit of being able to compute the feature importance based on the overall gain yield by this feature for a given class. Table 2 lists the top-5 most important features for each table type. The feature importance is the normalized sum of the gains, defined as the improvement in squared error, from every split point across all features as describe in [Friedman, 1999]. In this table, feature names from Section 4.2 are prefixed with the extraction point: *C* denotes column features, *R* denotes row features¸ 1 denotes the first column/row, 2 denotes the second column/row, and $ denotes the last column/row.

Features based on lexical content matter a great deal for identifying relational tables. Among the top-5 most important features for ATTRIBUTE/VALUE and VERTICAL LISTING table types, four are from the *Lexical features* family. As predicted by our second baseline, the presence of a "colon" in an ATTRIBUTE/VALUE table is an extremely important feature. It is however surprising that the second row is the center of attention for this feature and not the first column. The reason might be that the presence of a cell ending with ":" in the second row is more indicative of a repetition of such a character in the following rows. Moreover, the ratio should be 0.5 in the case of an ideal ATTRIBUTE/VALUE formatting (the number of columns containing the attributes and

Table 3. TABEX classification confusion matrix.

Reference \ System	Formatting	Navigational	Attribute/Value	Vertical Listing	Horizontal Listing	Enumeration	Calendar	Matrix	Form	Others	Total
Formatting	2235	55	101	67	3	44	0	1	23	3	2532
Navigational	183	135	5	13	2	41	1	0	1	2	383
Attribute/Value	85	5	622	84	7	4	0	1	1	2	811
Vertical Listing	79	10	69	535	14	20	2	3	1	10	743
Horizontal Listing	14	2	14	17	8	0	1	1	0	1	58
Enumeration	70	50	9	26	2	75	0	0	1	0	233
Calendar	1	0	0	7	0	0	69	1	0	0	78
Matrix	0	0	1	3	0	0	1	2	0	0	7
Form	38	2	4	1	0	0	0	0	67	1	113
Others	13	0	4	13	0	0	0	0	0	12	42
Total	2718	259	829	766	36	184	74	9	94	31	5000

the values should be identical). Also as predicted, when casting the baseline classifier for FORM type tables, the presence of <*input*> and <*select*> tags is a highly reliable clue in order to identify the FORM table type.

The high precision classification of CALENDAR seems to rely mainly on *MAX_COLS* which shows a large gap with the next important feature. CALENDARS tend to have a fixed number of columns, one for each day of the week, which is a precise signature. Finally, NAVIGATIONAL type classification intuitively uses anchor tag features as the main component for its model (since navigational tables are used to hyperlink to other documents).

5.4 Error Analysis

There is a great deal of overlap between the table types defined in Figure 2. The <*table*> container primary use is to structure visually some elements which could be knowledge or/and design/layout features. For this reason, classification between certain types of tables might be harder than others. In order to assess how hard it is to distinguish between some types of tables, we report in Table 3 the confusion matrix based on the gold standard truth given by our editors and the TABEX system classification for all of the classes.

For the VERTICAL/LISTING class we see a fair amount (~11%) of misclassification to ATTRIBUTE/VALUE and vice-versa (~8%). It underlines the similarities between the 2 table types. Moreover, 24% of HORIZONTAL LISTINGS have been wrongly assigned the ATTRIBUTE/VALUE class, which emphasizes the fact that the latter is a specific case of the former (i.e., ATTRIBUTE/VALUE tables are defined as horizontal listings with only one column of values). Since FORMATTING is a default case for any table that does not fall in the other categories (other classes are specifically for relational knowledge), there are a fair amount of examples from other classes falling into this category (12% across classes). The most errors come from the NAVIGATIONAL table type, where 48% are misclassified. Once again, it shows how difficult it is to distinguish <*table*> formatting to structure navigational elements from generic layout formatting. However, those errors are not critical since we are targeting knowledge extraction from relational tables (both FORMATTING and NAVIGATIONAL don't contain any semantic triples).

5.5 Comparison Against Prior Art

Several methods have been proposed in the past for table classification, but most consider it as a binary problem (over coarse-grained classes such as genuine (i.e., relational tables) vs. non-genuine (i.e., formatting tables)), making a direct comparison with our work difficult.

Penn et al. [12] presented a rule-based system for finding genuine tables. This system is described in Section 3.4 and is referred to in this section as "Penn et al."

Table 4 compares the performance of our system on the simpler binary classification task (genuine vs. non-genuine tables). We used the test set sampled from our 8.2 billion tables described in Section 3.1. Note that the main fine-grained classification experiments reported in Table 2 were conducted on a smaller 1.3 billion table sample after applying the filter described in Section 3.4 (which is a reimplementation of Penn et al. [12]). In this section, we want to compare directly against [12]. Since Web tables are highly skewed towards non-genuine tables (see Figure 3), we implemented a simple baseline, *Baseline NG*, where the class non-genuine is always predicted.

Penn et al. has good precision and recall on the non-genuine class, however it fails at identifying genuine classes. The system is a very good initial filter (as used in our approach) for trimming out many non-genuine tables. A more specialized system is needed however for teasing out the genuine tables from the remaining tables.

The reported accuracy results show small gains as we add our features to the Penn et al. system. This is misleading since our system focuses only on teasing out genuine tables from those tables not caught by the Penn et al. filter. It is therefore better to analyze the results on the Genuine class, where we see large gains with TabEx.

Another commonly cited binary classification engine is the machine learned model proposed by Wang and Hu [13][5]. Direct comparison against their results is not possible because of the way in which they extracted their test set of candidate tables. A set of 2851 pages were harvested from Google directory and News

[5] Cafarella et al. [2] is also often cited, however they use the exact same model as Wang and Hu [13].

Table 4. Comparative analysis on the simpler binary classification task of genuine vs. non-genuine tables.

	Genuine			Non-genuine			Accuracy
	P	R	F	P	R	F	
Baseline NG	-	-	-	0.873	1.000	0.932	0.762
Penn et al. [10]	0.440	0.542	0.486	0.931	0.882	0.906	0.869
Global Features	0.340	0.236	0.279	0.884	0.864	0.874	0.815
Layout Features	0.570	0.533	0.551	0.907	0.869	0.887	0.864
Html Features	0.595	0.567	0.581	0.909	0.868	0.888	0.870
Lexical Features	0.677	0.618	0.647	0.910	0.872	0.891	0.881
TabEx	0.704	0.662	**0.683**	0.914	0.872	**0.893**	**0.888**

using predefined keywords known to have a higher chance to recall *genuine* tables (e.g. *table*, *stocks*, *bonds*…), thus skewing the data set. In our case, we extracted a random sample of tables from a large 1.2 billion page web crawl. As evidence of incomparability, Wang and Hu [13] report *Precision*, *Recall*, *F-measure* scores of 0.482, 0.757, and 0.619, respectively on the Genuine class for the Penn et al. system, whereas on our dataset we report 0.440, 0.542, and 0.486, respectively.

We can, however, compare our features to those identified by Wang and Hu [13]. They identified three feature families: *Layout* features, *Content* features, and *Word group* features. The latter showed no gains in their results so we did not replicate them in our work. The new features that we added in TabEx include: *ratio_colspan*, *ratio_rowspan*, *dist_tags*, *ratio_th*, *ratio_f*, *ratio_br*, *dist_string*, and *ratio_colon* (see Section 4.2 for feature descriptions). We used the features *ratio_input* and *ratio_select* instead of the tag <*form*> used by Wang and Hu since we found that many tables that contain form elements have the <*form*> tag outside of the <*table*> container. Also, we split their *digit* feature into *ratio_contain_number* and *ratio_is_number*.

Because of our fine-grained taxonomy, many of these new features were the most discriminative for various classes in our taxonomy, as evidenced in Table 3. For example, *ratio_colon* and *dist_string* were important for discriminating the ATTRIBUTE/VALUE tables, and *ratio_th* and *ratio_f* are important for discriminating the MATRIX tables.

Also different in our approach is the extraction mechanism that we used for our features, resulting in larger feature vectors without a loss in feature density. For each table, we extracted our features for the first, second, and last columns and rows, as opposed to the table as a whole. Inspection of Table 3 reveals the importance of teasing out this structure.

6. CASE STUDY: EXTRACTION FROM ATTRIBUTE/VALUE TABLES

Section 3.2.1 outlines the challenges in extracting semantic triples from relational tables in our table type taxonomy. Extraction of these triples is outside of the scope of this paper, however, in this section, we propose a solution to a key challenge in extracting semantic triples from the most common relational table type, ATTRIBUTE/VALUE.

Tables of type ATTRIBUTE/VALUE have been mostly discarded in the prior art as non-relational tables [2], however there is a great deal of information that can be extracted from them. While their structure easily allows extracting the *predicates* (relations/attributes) and the *objects* (values) of semantic triples, they present a great challenge for extracting the *subject*. One of the major issues for extracting triples $\langle p, s, o \rangle$ is to identify the first argument of the relation, namely, the subject. We call this task *protagonist detection*. It is very common for protagonists to be absent from the ATTRIBUTE/VALUE table itself because they appear in context of the protagonist and there is no ambiguity to a user what/who the protagonist is. For example, consider a Web page describing the specifications of a particular digital camera. Since the whole page is *about* that digital camera, the name of the camera may never appear in the specification table (it may just occur as the title of the page or in an anchor text referring to this page).

The task of building semantic triples from attributes and values extracted from these tables falls back on the identification of the protagonist, which is typically unique for a given table. There are mainly three different places where the protagonist could be found:

- **Within the table**: In some cases it is present in the table with a generic attribute (e.g. *name*, *model*…)
- **Within the document**: It appears in the document body or the html <*title*>.
- **Outside of the document**: Anchor texts pointing to the page often contain the protagonist.

While table cells and anchor texts offer well defined boundaries for identifying protagonist candidates, the document body proposes fewer clues. There is however a series of html fields that could help in defining entity boundaries such as the headers and the font tags.

6.1 Experiments on Protagonist Detection

We have carried some preliminary experiments in trying to identify the protagonist of ATTRIBUTE/VALUE tables using a corpus of 200 manually annotated tables[6]. For each table, an editor identified the valid set[7] of protagonists among the content of the document or the anchor text pointing to it. None of the cases presented to the editors lacked a protagonist, highlighting that most often ATTRIBUTE/VALUE tables do indeed contain relational knowledge.

In order to identify all possible candidates, even if it is present in a paragraph of the document, we took an N-gram based approach. All possible 1 to 12-grams were extracted from the document and

[6] It is important to note that extracting semantic triples, and hence protagonist detection, is outside of the scope of this paper. This section proposes initial steps towards this goal, which we leave as future work.

[7] There might be more than one valid protagonist for a given table because of lexical variations(e.g., *Smith, Reid C.* and *Reid C. Smith*)

the anchor text (obtained from a commercial search engine's web link graph). For each N-gram, its frequency combined with its position (e.g. *anchor text*, *title*, *header*, *body*, *table*, *font…*) was used as features for our GBDT regression model (see Section 4.1). For some tables, as many as 1700 candidates were extracted.

We ran a 20-fold cross-validation experiment and present the results in Figure 8. Our system is labeled *ProIde* and it is compared against a simple baseline system that ranks the candidate protagonists according to their anchor text frequencies. This baseline achieves a surprisingly high precision of 40%. Although our system performs statistically significantly better than the baseline (by more than 25%), *ProIde* concedes 35% errors when looking at only the top suggestion and 12% errors when considering the top-10.

Our approach must be improved, but it is a good starting point for reducing the set of candidates in a first pass (97% chance to find the correct protagonist in the top-100 ranked candidates). Then, more expensive approaches could be used in order to verify whether the triples hold in other contexts using other extractors.

7. CONCLUSION

In this paper, we presented a large census of tables on a crawl of the web and proposed a table type taxonomy along with the frequencies of each table type. We showed that 88% of the tables on the Web are used strictly for layout purposes (i.e., they contain no relational knowledge). For each relational table in our taxonomy (i.e., tables containing relational knowledge), we outlined the major challenges for extracting from them semantic triples.

We presented a Web-scale supervised classification model (GBDT) for classifying Web tables into our fine-grained taxonomy. We proposed a set of rich layout features, structural content features, and lexical features for our model. Empirical results over a large experimental classification task showed that our model achieves high overall accuracy (75.2%) and high F-scores on the frequently occurring relational tables.

We then proposed a supervised GBDT regression model for automatically detecting the *subject* (or *protagonist*) of ATTRIBUTE/VALUE tables. In these tables, the *subject* of semantic triples is typically not found in the table itself, occurring in places such as the page title or even anchor text pointing to the page. Our method outperformed a simple baseline and showed that it can find the correct protagonist in 90% of the cases in its top-20 ranked candidates, and in 79% of the cases in its top-3.

The goal of this paper was twofold: i) provide a table type taxonomy along with a census of their frequencies on the Web; ii) propose a rich fine-grained classification model for assigning Web tables to the taxonomy. In future work, we hope to tackle each relational table type, one by one, and extract with high accuracy all the semantic triples they contain.

8. REFRENCES

[1] Cafarella, M.J.; Halevy, A.; Wang, D. Z.; Wu, E.; and Zhang, Y. 2008.. WebTables: Exploring the Powerpower of Tablestables on the Web. In *Proceedings of VLDB-08*. Auckland, New Zealandthe 34th International Conf. on Very Large Data Bases, pages 538-549, 2008.

[2] Cafarella, M. J.; Halevy, A.; Zhang, Y.; Wang, D. Z.; and Wu, E. Uncovering the Relational Web. In *WebDB*, Vancouver, Canada, 2008.

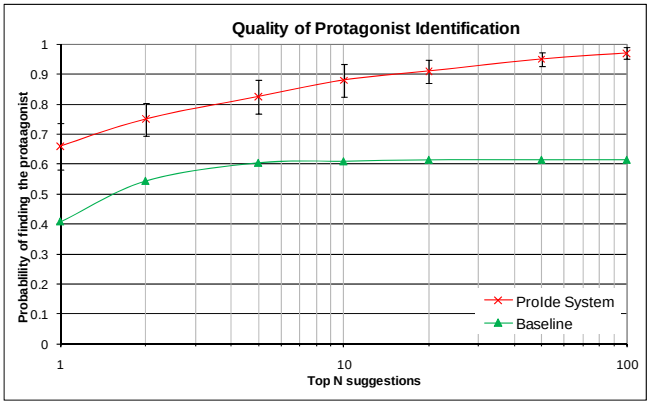

Figure 7. Probability of finding a correct protagonist vs. rank.

[3] Chang, W.; Pantel, P.; Popescu, A.-M.; and Gabrilovich, E. 2009. Towards intent-driven bidterm suggestion. In *Proceedings of WWW-09 (Short Paper)*, Madrid, Spain.

[4] Chen, H.; Tsai, S.; and Tsai, J. 2000. Mining Tables from Large-Scale HTML Texts. *In Proceedings of COLING-00*. Saarbrücken, Germany.

[5] Elmeleegy, H.; Madhavan, J.; and Halevy, A. 2009. Harvesting Relational Tables from Lists on the Web. In *Proceedings of the VLDB Endowment (PVLDB)*. pp. 1078-1089.

[6] Gazen, B. and Minton, S.; 2006. Overview of Autofeed: An Unsupervised Learning System for Generating Webfeeds. In *Proceedings of AAAI-06*. Boston, MA.

[7] Huanhuan, J. H.; Jiang, D.; Pei, J.; He, Q.; Liao, Z.; Chen, E.; and Li, H. 2008. Context-aware query suggestion by mining click-through and session data. In *Proceedings of KDD-08*. pp. 875–883.

[8] Friedman, J.H. 2001. Greedy function approximation: A gradient boosting machine. *Annals of Statistics*, 29(5):1189–1232.

[9] Friedman, J.H. 2006. Recent advances in predictive (machine) learning. *Journal of Classification*, 23(2):175–197.

[10] Gatterbauer, W.; Bohunsky, P.; Herzog, M.; Krupl, B.; and Pollak, B. 2007. Towards Domain-Independent Information Extraction from Web Tables. *In Proceedings WWW-07*. pp. 71–80. Banff, Canada.

[11] Lin, D.; Zhao, S.; Qin, L.; and Zhou, M. 2003. Identifying Synonyms among Distributionally Similar Words. In *Proceedings of IJCAI-03*, pp.1492-1493. Acapulco, Mexico.

[12] Penn, G.; Hu, J.; Luo, H.; and McDonald, R. 2001. Flexible Web Document Analysis for Delivery to Narrow-Bandwidth Devices. In *Proceedings of the Sixth International Conference on Document Analysis and Recognition*. pp. 1074-1078. Seattle, WA.

[13] Wang, Y. and Hu, J. 2002. A Machine Learning Based Approach for Table Detection on the Web. In Proceedings of WWW-02. Honolulu, Hawaii.

[14] Yoshida, M.; Torisawa, K.; and Tsujii, J. 2001. A Method to Integrate Tables of the World Wide Web. In Proceedings of Workshop on Web Document Analysis. pp. 31–34.