Identifying Web Tables – Supporting a Neglected Type of Content on the Web

Mikhail Galkin
University of Bonn, Germany
ITMO University, Saint Petersburg, Russia
galkin@iai.uni-bonn.de

Dmitry Mouromtsev
ITMO University, Saint Petersburg, Russia
mouromtsev@mail.ifmo.ru

Sören Auer
University of Bonn, Germany
auer@cs.uni-bonn.de

ABSTRACT

The abundance of the data in the Internet facilitates the improvement of extraction and processing tools. The trend in the open data publishing encourages the adoption of structured formats like CSV and RDF. However, there is still a plethora of unstructured data on the Web which we assume contain semantics. For this reason, we propose an approach to derive semantics from web tables which are still the most popular publishing tool on the Web. The paper also discusses methods and services of unstructured data extraction and processing as well as machine learning techniques to enhance such a workflow. The eventual result is a framework to process, publish and visualize linked open data. The software enables tables extraction from various open data sources in the HTML format and an automatic export to the RDF format making the data linked. The paper also gives the evaluation of machine learning techniques in conjunction with string similarity functions to be applied in a tables recognition task.

Categories and Subject Descriptors

D.2 [**Software**]: Software Engineering; D.2.8 [**Software Engineering**]: Metrics—*complexity measures, performance measures*

General Terms

Algorithms

Keywords

Machine Learning, Linked Data, Semantic Web

1. INTRODUCTION

The Web contains various types of content, e.g. text, pictures, video, audio as well as tables. Tables are used everywhere in the Web to represent statistical data, sports results, music data and arbitrary lists of parameters. Recent research [2, 3] conducted on the *Common Crawl* census[1] indicated that an average Web page contains at least nine tables. In this research about 12 billion tables were extracted from a billion of HTML pages, which demonstrates the popularity of this type of data representation. Tables are a natural way how people interact with structured data and can provide a comprehensive overview of large amounts and complex information. The prevailing part of structured information on the Web is stored in tables. Nevertheless, we argue that table is still a neglected content type regarding processing, extraction and annotation tools.

For example, even though there are billions of tables on the Web search engines are still not able to index them in a way that facilitates data retrieval. The annotation and retrieval of pictures, video and audio data is meanwhile well supported, whereas on of the most widespread content types is still not sufficiently supported. Assumption that an average table contains on average 50 facts it is possible to extract more than 600 billion facts taking into account only the 12 billion sample tables found in the Common Crawl. This is already *six* times more than the whole *Linked Open Data Cloud*[2]. Moreover, despite a shift towards semantic annotation (e.g. via RDFa) there will always be plain tables abundantly available on the Web. With this paper we want turn a spotlight on the importance of tables processing and knowledge extraction from tables on the Web.

The problem of deriving knowledge from tables embedded in an HTML page is a challenging research task. In order to enable machines to understand the meaning of data in a table we have to solve certain problems:

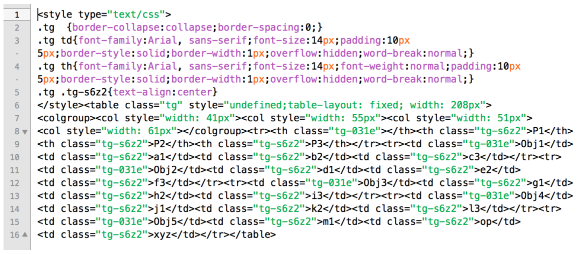
1. Search for relevant Web pages to be processed;
2. Extraction of the information to work with;
3. Determining relevance of the table;
4. Revealing the structure of the found information;
5. Identification of the data range of the table;
6. Mapping the extracted results to existing vocabularies and ontologies.

The difference in recognizing a simple table by a human and a machine is depicted in Fig. 1. Machine are not easily able to derive formal knowledge about the content of the table.

The paper describes current methodologies and services to tackle some crucial Web table processing challenges and in-

[1] Web: `http://commoncrawl.org/`
[2] Web: `http://stats.lod2.eu/`

	P1	P2	P3
Obj1	a1	B2	C3
Obj2	D1	E2	F3
Obj3	G1	H2	I3
Obj4	J1	K2	L3
Obj5	MkS	NxA	xyz



```
<style type="text/css">
.tg  {border-collapse:collapse;border-spacing:0;}
.tg td{font-family:Arial, sans-serif;font-size:14px;padding:10px
5px;border-style:solid;border-width:1px;overflow:hidden;word-break:normal;}
.tg th{font-family:Arial, sans-serif;font-size:14px;font-weight:normal;padding:10px
5px;border-style:solid;border-width:1px;overflow:hidden;word-break:normal;}
.tg .tg-s6z2{text-align:center}
</style><table class="tg" style="undefined;table-layout: fixed; width: 208px">
<colgroup><col style="width: 41px"><col style="width: 55px"><col style="width: 51px">
<col style="width: 61px"></colgroup><tr><th class="tg-031e"></th><th class="tg-s6z2">P1</th>
<th class="tg-s6z2">P2</th><th class="tg-s6z2">P3</th></tr><tr><td class="tg-031e">Obj1</td>
<td class="tg-s6z2">a1</td><td class="tg-s6z2">b2</td><td class="tg-s6z2">c3</td></tr><tr>
<td class="tg-031e">Obj2</td><td class="tg-s6z2">d1</td><td class="tg-s6z2">e2</td>
<td class="tg-s6z2">f3</td></tr><tr><td class="tg-031e">Obj3</td><td class="tg-s6z2">g1</td>
<td class="tg-s6z2">h2</td><td class="tg-s6z2">i3</td></tr><tr><td class="tg-031e">Obj4</td>
<td class="tg-s6z2">j1</td><td class="tg-s6z2">k2</td><td class="tg-s6z2">l3</td></tr><tr>
<td class="tg-031e">Obj5</td><td class="tg-s6z2">m1</td><td class="tg-s6z2">op</td>
<td class="tg-s6z2">xyz</td></tr></table>
```

Figure 1: Different representations of one table.

troduces a new approach of table data processing which combines advantages of Semantic Web technologies with robust machine learning algorithms. Our approach allows machines to distinguish certain types of tables (genuine tables), recognize their structure (orientation check) and dynamically link the content with already known sources.

The paper is structured as follows: Section 2 gives an overview of related studies in the field of unstructured data processing. Section 3 presents Web services which provide the user with table extraction functions. Section 4 describes the approach and establishes a mathematical ground for a further research. Section 5 presents used machine learning algorithms and string distance functions. Section 6 showcases the evaluation of the approach. Finally, we derive conclusions and share plans for future work.

2. RELATED WORK

The Linked Open Data concept raises the question of automatic tables processing as one of the most crucial. Open Government Data is frequently published in simple HTML tables that are not well structured and lack semantics. Thus, the problems discussed in the paper [12] concern methods of acquiring datasets related to roads repair from the government of Saint Petersburg. There is also a raw approach [10] in information extraction, which is template-based and effective in processing of web sites with unstable markup. The crawler was used to create a LOD dataset of CEUR Workshop[3] proceedings.

A.C. e Silva et al. in their paper [14] suggest and analyze an algorithm of table research that consists of five steps: location, segmentation, functional analysis, structural analysis and interpretation of the table. The authors provide a comprehensive overview of the existing approaches and designed a method for extracting data from ASCII tables. However, smart tables detection and distinguishing is not considered.

J. Hu et al. introduced in the paper [8] the methods for table detection and recognition. Table detection is based on the idea of edit-distance while table recognition uses random graphs approach. M. Hurst takes into consideration ASCII tables [9] and suggests an approach to derive an abstract geometric model of a table from a physical representation. A graph of constraints between cells was implemented in order to determine position of cells. Nevertheless, the results are rather high which indicates the efficiency of the approach. The authors of the papers achieved significant success in structuring a table, but the question of the table content and its semantic is still opened.

D. Embley et al. tried [4] to solve the table processing problem as an extraction problem with an introduction of machine learning algorithms. However, the test sample was rather small which might have been resulted in overfitting [1].

W. Gatterbauer et al. in the paper [6] developed a new approach towards information extraction. The emphasis is made on a visual representation of the web table as rendered by a web browser. Thus the problem becomes a question of analysis of the visual features, such as 2-D topology and typography.

Y. A. Tijerino et al. introduced in [15] TANGO approach (Table Analysis for Generating Ontologies) which is mostly based on WordNet with a special procedure of ontology generation. The whole algorithm implies 4 actions: table recognition, mini-ontology generation, inter-ontology mappings discovery, merging of mini-ontologies. During the table recognition step search in WordNet support the process of table segmentation su that no machine learning algorithms were applied.

V. Crescenzi et al. introduced in [18] an algorithm RoadRunner for an automatic extraction of the HTML tables data. The process is based on regular expressions and matches/mismatches handling. Eventually a DOM-like tree is built which is more convenient for an information extraction. However, there are still cases when regular expressions are of little help to extract any data.

To sum up, there are different approaches to information extraction developed last ten years. In our work we introduce an effective extraction and analyzing framework built on top of those methodologies combining tables recognition techniques, machine learning algorithms and Semantic Web methods.

2.1 Existing Data Processing Services

Automatic data extraction has always been given a lot of attention from the Web community. There are numerous web-services that provide users with sophisticated instruments useful in web scraping, web crawling and tables processing. Some of them are presented below.

2.1.1 ScraperWiki

ScraperWiki[4] is a powerful tool based on subscription model that is suitable for software engineers and data scientists whose work is connected with processing of large amounts of data. Being a platform for interaction between business, journalists and developers, ScraperWiki allows users to solve

[3] Web: http://ceur-ws.org/

[4] Web: https://scraperwiki.com/

extracting and cleaning tasks, helps to visualize acquired information and offers tools to manage retrieved data. Some of the features of the service:

- Dataset subscription makes possible the automatic tracking, update and processing of the specified dataset in the Internet.
- A wide range of data processing instruments. For instance, information extraction from PDF documents

ScraperWiki allows one to parse web tables in CSV format, but processes all the tables on the page even thought they do not contain relevant data, e.g. layout tables. Also the service does not provide any Linked Data functionality.

2.1.2 Scrapy.

Scrapy[5] is a fast high-level framework written in Python for web-scraping and data extraction. Scrapy is spread under BSD license and available on Windows, Linux, MacOS and BSD. Merging performance, speed, extensibility and simplicity Scrapy is a popular solution in the industry. A lot of services are based on Scrapy, such as ScraperWiki or PriceWiki[6].

2.1.3 Bitrake.

Bitrake[7] is a subscription based tool for scraping and processing the data. Bitrake offers a special service for those who are not acquainted with programming and provides an API for experienced developers written in Ruby and JavaScript. One of the distinctive features of the service is a self–developed scraping algorithm with simple filtering and configuration options. Bitrake is also used in monitoring and data extraction tasks.

2.1.4 Apache Nutch.

Nutch[8] is an open source web crawler system based on Apache Lucene and written in Java. Main advantages of Nutch are performance, flexibility and scalability. The system has a modular architecture which allows developers to create custom extensions, e.g. extraction, transforming extensions or distribute computing extensions. On the other hand, the tool does not have built-in Linked Data functionality, which requires additional development.

2.1.5 Import.io.

Import.io[9] is am emerging data processing service. Comprehensive visualization and an opportunity to use the service without programming experience tend Import.io to become one of the most wide-spread and user-friendly software. The system offers users three methods of extraction arranged by growing complexity: an extractor, a crawler and a connector. The feature of automatic table extraction is also implemented but supports only CSV format.

3. CONCEPT

In order to achieve the automatic tables processing certain problems have to be solved:

1. HTML tables search and localization from a URL provided by the user;
2. Computing of appropriate heuristics;
3. Table genuineness check, in other words, check whether a table contains relevant data;
4. Table orientation check (horizontal or vertical);
5. Transformation of the table data to an RDF model.

The importance of correct tables recognition affects the performance of most of web–services. The growth of the data on the Web facilitates the data- and knowledge bases updates with such a frequency, that does not allow errors, inconsistency or ambiguity. With the help of our methodology we aim to address the challenges of automatic *knowledge extraction* and *knowledge replenishment*.

3.1 Knowledge retrieval

Knowledge extraction enables the creation of knowledge bases and ontologies using the content of HTML tables. It is also a major step towards five-star open data[10] making the knowledge linked with other datasources and accessible, in addition, in a machine-readable format. Thus, a correct table processing and an ontology generation is a crucial part of the entire workflow. Our framework implements learning algorithms which allow automatic distinguishing between *genuine* and *non-genuine* tables [17], as well as automatic ontology generation.

We call a table *genuine* when it contains consistent data (e.g. the data the user is looking for) and we call a table *non-genuine* when it contains any HTML page layout information or a rather useless content, e.g. a list of hyperlinks to other websites within one row or one column.

3.2 Knowledge acquisition

Knowledge replenishment raises important questions of data updating and deduplication. A distinctive feature of our approach is a fully automatic update from the datasource. The proposed system implements components of the powerful platform *Information Workbench* [7] which introduces the mechanism of *Data Providers*. Data Providers observe a datasource specified by a user and all its modifications according to a given schedule. Therefore, it enables the replenishment of the same knowledge graph with new entities and facts, which, in turn, facilitates data deduplication.

4. FORMAL DEFINITIONS

The foundation of the formal approach is based on ideas of Ermilov et al. [5]

Definition 1. A table $\mathcal{T} = (\mathcal{H}, \mathcal{N})$ is tuple consisting of a header $\mathcal{H}$ and data nodes $\mathcal{N}$, where:

- the header $\mathcal{H} = \{h_1, h_2, \ldots, h_n\}$ is an n-tuple of header elements h_i. We also assume that the set of headers might be optional, e.g. $\exists \mathcal{T} \equiv \mathcal{N}$. If the set of headers exists, it might be either a row or a column.

[5] Web: http://scrapy.org/

[6] Web: http://www.pricewiki.com/blog/

[7] Web: http://www.bitrake.com/

[8] Web: http://nutch.apache.org/

[9] Web: https://import.io/

[10] Web: http://5stardata.info/

Table 1: Horizontal orientation

H0	Header 1	Header 2	Header 3	Header 4
Obj 1				
Obj 2				
Obj 3				
Obj 4				

Table 2: Vertical orientation

H0	Obj 1	Obj 2	Obj 3	Obj 4
Header 1				
Header 2				
Header 3				
Header 4				

- the data nodes $\mathcal{N} = \begin{pmatrix} c_{1,1} & c_{1,2} & \cdots & c_{1,m} \\ c_{2,1} & c_{2,2} & \cdots & c_{2,m} \\ \vdots & \vdots & \ddots & \vdots \\ c_{n,1} & c_{n,2} & \cdots & c_{n,m} \end{pmatrix}$ are a *(n,m)* matrix consisting of n rows and m columns.

Definition 2. The *genuineness* of the table is a parameter which is computed via the function of grid nodes and headers, so that $G_T(\mathcal{N}, \mathcal{H}) = g_k \in \mathcal{G}, \mathcal{G} = \{true, flase\}$.

Definition 3. The *orientation* of the table is a parameter which values are defined in the set $\mathcal{O} = \{horizontal, vertical\}$ if and only if $G_T(\mathcal{N}, \mathcal{H}) = true$. So that $\exists O_T \iff G_T \equiv true$. The orientation of the table is computed via a function of grid nodes and headers, so that $O_T(\mathcal{N}, \mathcal{H}) = o_k \in \mathcal{O}$. If the orientation is horizontal, then the headers are presented as a row, so that $O_T \equiv horizontal$. If the orientation is vertical, then the headers are presented as a column, so that $O_T \equiv vertical$.

Definition 4. A set of *heuristics* $\mathcal{V}$ is a set of quantitative functions of grid nodes $\mathcal{N}$ and headers $\mathcal{H}$ which is used by machine learning algorithms in order to define *genuineness* and *orientation* of the given table $\mathcal{T}$.

$$G_T(\mathcal{N}, \mathcal{H}) = \begin{cases} true, & V_{g1} \in [v_{g1_{min}}, v_{g1_{max}}], \ldots V_{gm} \\ false, & \text{otherwise.} \end{cases} \quad (1)$$

where $V_{g1}, ..., V_{gm} \in \mathcal{V}$, $[v_{g1_{min}}, v_{g1_{max}}]$ is the range of values of V_{g1} necessary for the *true* value in conjunction with $V_{g2}, ..., V_{gm}$ functions.

$$O_T(\mathcal{N}, \mathcal{H}) = \begin{cases} horizontal, & V_{h1} \in [v_{h1_{min}}, v_{h1_{max}}], ..., V_{hn}. \\ vertical, & V_{v1} \in [v_{v1_{min}}, v_{v1_{max}}], ..., V_{vl}. \end{cases} \quad (2)$$

where $V_{h1}, ..., V_{hn} \in \mathcal{V}$, $V_{v1}, ..., V_{vl} \in \mathcal{V}$, $[v_{h1_{min}}, v_{h1_{max}}]$ is the range of values of V_{h1} necessary for the *horizontal* value in conjunction with $V_{h2}, ..., V_{hn}$ functions, $[v_{v1_{min}}, v_{v1_{max}}]$ is the range of values of V_{v1} necessary for the *vertical* value in conjunction with $V_{v2}, ..., V_{vl}$ functions.

Thus, it becomes obvious, that the *heuristics* are used in order to solve the classification problem [11] $X = \mathbb{R}^n, Y = \{-1, +1\}$ where the data sample is $X^l = (x_i, y_i)_{i=1}^l$ and the goal is to find the parameters $w \in \mathbb{R}^n, w_0 \in \mathbb{R}$ so that:

$$a(x, w) = sign(\langle x, w \rangle - w_o). \quad (3)$$

We describe heuristics and machine learning algorithms in detail in Section 5. From description logics we define the terminological component $TBox_T$ as a set of concepts over the set of headers $\mathcal{H}$. We define the assertion component $ABox_T$ as a set of facts over the set of grid nodes $\mathcal{N}$.

Hypothesis 1 *Tables in unstructured formats contain semantics.*

$$\exists \mathcal{T} | \{\mathcal{H}_T \iff TBox_T, \mathcal{N}_T \iff ABox_T\} \quad (4)$$

In other words, there are tables with the relevant content, which could be efficiently extracted as knowledge.

The evaluation of the hypothesis is presented in the Section 7

4.1 Algorithm Description

Algorithm 1 The workflow of the system

```
procedure WORKFLOW
    $URI \leftarrow$ specified by the user
    tables localization
    $n \leftarrow$ found tables
while $n > 0$ :
    $n \leftarrow n - 1$.
    if genuine $= true$ then
        orientation check.
        RDF transformation.
    goto while.
```

The valid URL of the website where the data is situated is required from the user.

The next step is a process of search and localization of HTML tables on a specified website. One of the essential points in the process is to handle DOM < *table* > leaf nodes in order to avoid nested tables. Most of the systems described in 2.1 suggest a user with all the extracted tables, whether they are formatting tables or relevant tables. In contrast, our approach envisions full automation with the subsequent ontology generation.

The next step is a computation of heuristics for every extracted table. Using a training set and heuristics a machine learning algorithm classifies the object into a genuine or a non-genuine group. The input of the machine learning module is a *table trace* – a unique multi-dimensional vector of computed values of the heuristics of a particular table. Using a training set the described classifiers decide, whether the vector satisfies the genuineness class requirements or not. If the vector is decided to be genuine the vector then is explored by classifiers again in attempt to define the orientation of the table.

The correct orientation determination is essential for correct transformation of the table data to semantic formats. Then it becomes possible to divide data and metadata of a table and construct an ontology.

If the table is decided to be a non-genuine then a user receives a message where it is stated that a particular table is not genuine according to the efficiency of a chosen machine learning method. However, the user is allowed to manually mark a table as a genuine which in turn modifies machine learning parameters.

5. MACHINE LEARNING METHODS

Machine learning algorithms play vital role in the system which workflow is based on appropriate heuristics which in turn are based on string similarity functions. The nature of the problem implies usage of mechanisms that analyze the content of the table and calculate a set of parameters. In our case the most suitable option is implementation of string similarity (or string distance) functions.

5.1 String Metrics

String metric is a metric that measures similarity or dissimilarity between two text strings for approximate string matching or comparison. In the paper three string distance functions are used and compared.

Levenshtein distance is calculated as a minimum number of edit operations (insertions, substitutions, deletions) required to transform one string into another. Characters matches are not counted.

Jaro-Winkler distance is a string similarity metric and improved version of the Jaro distance. It is widely used to search for duplicates in a text or a database. The numerical value of the distance lies between 0 and 1 which means two strings are more similar the closer the value to 1.

Being popular in the industry [20] **n-gram** is a sequence of n items gathered from a text or a speech. In the paper n-grams are substrings of a particular string with the length n.

5.2 Improvements

Due to the mechanism of tables extraction and analysis certain common practices are improved or omitted. Hence particular changes in string similarity functions have been implemented:

- The content type of the cell is more important than the content itself. Thus it is reasonable to equalize all numbers and count them as the same symbol. Nevertheless the order of magnitude of a number is still taken into account. For instance, the developed system recognizes 3.9 and 8.2 as the same symbols, but 223.1 and 46.5 would be different with short distance between these strings.

- Strings longer than three words have fixed similarity depending on a string distance function in spite of previously described priority reasons. Moreover, tables often contain fields like “description” or “details” that might contain a lot of text which eventually might make a mistake during the heuristics calculations. So that the appropriated changes have been implemented into algorithms.

5.3 Heuristics

Relying on the theory above it is now possible to construct a set of primary heuristics. The example table the heuristics mechanisms are explained with is:

Table 3: An example table

Name	City	Phone	e-mail
Ivanov I. I.	Berlin	1112233	ivanov@mail.de
Petrov P.P	Berlin	2223344	petrov@mail.de
Sidorov S. S.	Moscow	3334455	sidorov@ya.ru
Pupkin V.V.	Moscow	4445566	pupkinv@gmail.com

5.3.1 Maximum horizontal cell similarity.

The attribute indicates the maximum similarity of a particular pair of cells normalized to all possible pairs in the row found within the whole table under the assumption of horizontal orientation of a table. It means the first row of a table is not taken into account because of a header of a table (see Table 1). Having in mind the example table the strings “Ivanov I.I.” and “ivanov@mail.de” are more similar to each other than “Ivanov I.I.” and “1112233”.

The parameter is calculated under the certain rule:

$$maxSimHor = max_{i=2,n} \frac{\sum_{j_1=1}^{m} \sum_{j_2=1}^{m} dist(c_{i,j_1}, c_{i,j_2})}{m^2} \quad (5)$$

where i – a row, n – number of rows in a table, j – a column, m – number of columns in a table, *dist()* – a string similarity function, $c_{i,j}$ – a cell of a table.

5.3.2 Maximum vertical cell similarity.

The attribute indicates the maximum similarity of a particular pair of cells normalized to all possible pairs in the column found within the whole table under the assumption of vertical orientation of a table. It means the first column of a table is not taken into account because in most cases it contains a header (see Table 2). According to the example table the parameter calculated for this table would be rather high because it contains pairs of cells with the same content (for instance “Berlin” and “Berlin”).

Using the same designations the parameter is calculated:

$$maxSimVert = max_{j=2,m} \frac{\sum_{i_1=1}^{n} \sum_{i_2=1}^{n} dist(c_{i_1,j}, c_{i_2,j})}{n^2} \quad (6)$$

It is obvious that the greater the maximum horizontal similarity the greater a chance that a table has vertical orientation. Indeed, if the distance between values in a row is rather low it might mean that those values are instances of a particular attribute. The hypothesis is also applicable to the maximum vertical similarity which indicates possible horizontal orientation.

5.3.3 Average horizontal cell similarity.

The parameter indicates average similarity of the content of rows within the table under the assumption of horizontal orientation of a table. Again, the first row is not taken into account. The parameter is calculated under the certain rule:

$$avgSimHor = \frac{1}{n}\sum_{i=2}^{n}\frac{\sum_{j_1=1}^{m}\sum_{j_2=1}^{m} dist(c_{i,j_1}, c_{i,j_2})}{m^2} \quad (7)$$

where i – a row, n – number of rows in a table, j – a column, m – number of columns in a table, *dist()* – a string similarity function, $c[i,j]$ – a cell of a table.

The main difference between maximum and average parameters is connected with size of a table. Average parameters give reliable results during the analysis of large tables whereas maximum parameters are applicable in case of small tables.

5.3.4 Average vertical cell similarity.

The parameter indicates average similarity of the content of columns within the table under the assumption of vertical orientation of a table. The first column is not taken into account.

$$avgSimVert = \frac{1}{m}\sum_{j=2}^{m}\frac{\sum_{i_1=1}^{n}\sum_{i_2=1}^{n} dist(c_{i_1,j}, c_{i_2,j})}{n^2} \quad (8)$$

5.4 Machine Learning Algorithms

With the formalization established we are now ready to build classifiers which use apparatus of machine learning. Four machine learning algorithms are considered in the paper:

Naive Bayes classifier is a simple and popular machine learning algorithm. It is based on Bayes' theorem with naive assumptions regarding independence between parameters presented in a training set. However, this ideal configuration rarely occurs in real datasets so that the result always has a statistical error [11].

A decision tree is a predictive model that is based on tree–like graphs or binary trees [13]. Branches represent a conjunction of features and a leaf represents a particular class. Going down the tree we eventually end up with a leaf (a class) with its own unique configuration of the features and values.

k-nearest neighbours is a simple classifier based on a distance between objects [16]. If an object might be represented in Euclidean space then there is a number of functions that could measure a distance between these objects. If the majority of neighbours of the object belongs to one class than the object would be classified into the same class.

Support Vector Machine is a non–probabilistic binary linear classifier that tries to divide instances of classes presented in a training set by a gap as wide as possible. In other words, SVM builds separating surfaces between categories, which might be linear or non-linear [19].

6. IMPLEMENTATION

The proposed approach was implemented in Java as a plugin for the Information Workbench[11] platform developed by fluidOps. The platform provides numerous helpful APIs responsible for the user interaction, RDF data maintenance, ontology generation, knowledge bases adapters and smart data analysis. The machine learning algorithms are supplied by *WEKA*[12] – a comprehensive data mining Java framework developed by the University of Waikato. The *SimMetrics*[13] library by UK Sheffield University provided string similarity functions. The plugin is available on a public repository[14] both as a deployable artifact for the Information Workbench and in source codes.

7. EVALUATION

The main goal of the evaluation is to assess the performance of the proposed methodology in comparison with the existing solutions. By the end of the section we decide whether the hypothesis made in Section 4 is demonstrated or not. The evaluation consists of two subgoals – the evaluation of machine learning algorithms with string similarity functions and the evaluation of the system as a whole.

7.1 Algorithms Evaluation

A training set of 400 tables taken from the corpus[15] as a result of [2] was prepared to test suggested heuristics and machine learning methods. In addition, the efficiency of algorithms modifications was estimated during the tests. Results are presented on the Table 4 and Fig. 2,3. In case of the genuineness check *Precision*, *Recall* and *F–Measure* were computed.

Fig. 2 represents the overall fracture of correctly classified genuine and non-genuine tables w.r.t. used machine learning algorithms and string similarity functions. The machine learning algorithm based on kNN in conjunction with Levenshtein distance or n-grams demonstrated the highest efficiency during the genuineness check. A slight increase in efficiency in spite of modifications is observed mostly for kNN. It also could be noted that overall results of classification are generally lower in comparison with orientation classification task. This may indicate a lack of information about the table structure caused by a small amount of heuristics. Development and implementation of more sophisticated numerical parameters is suggested in order to improve the performance of classification. Hence, the way towards improving overall F–Measure is connected with raising Recall of the approach.

Fig. 3 indicates the high efficiency of the orientation check task. Most of the used machine learning methods except Naive Bayes demonstrated close to 100% results. A relatively low result of Naive Bayes regardless of the chosen string similarity function might be explained by a number of assumptions which the method is established on. On the one hand the algorithm has the advantage of simplicity and on the other hand it might overlook important details which affect the classification process because of such simplicity. During the orientation check only genuine tables are considered and assessed. Therefore, the eventual result is Precision.

[11] Web: http://www.fluidops.com/en/portfolio/information_workbench/

[12] Web: http://www.cs.waikato.ac.nz/ml/weka/

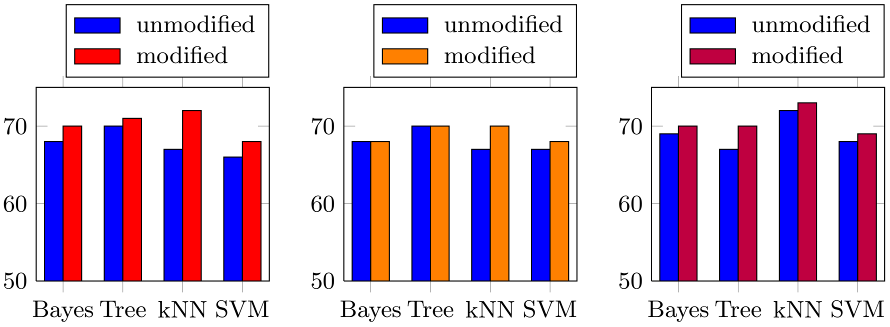
[13] Web: http://sourceforge.net/projects/simmetrics/

[14] Web: https://github.com/migalkin/Tables_Provider

[15] WDC – Web Tables. Web: http://webdatacommons.org/webtables/

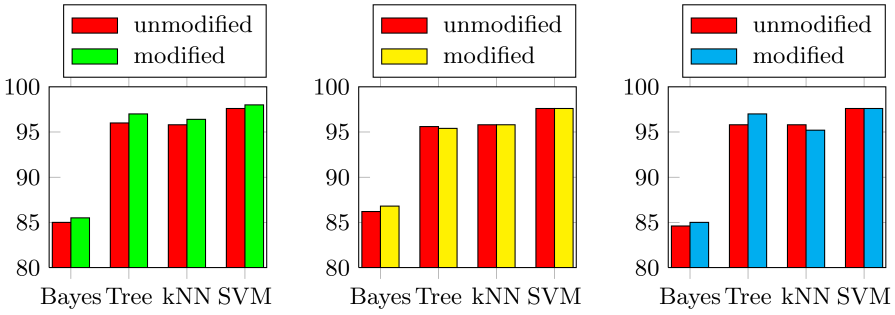
Table 4: Evaluation of the genuineness check

Method			Precision	Recall	F-Measure
Naive Bayes	Levenshtein	unmodified	0.925	0.62	0.745
		modified	0.93	0.64	0.76
	Jaro-Winkler	unmodified	0.939	0.613	0.742
		modified	0.939	0.617	0.744
	n-grams	unmodified	0.931	0.633	0.754
		modified	0.937	0.643	0.763
Decision Tree	Levenshtein	unmodified	0.928	0.65	0.765
		modified	0.942	0.653	0.76
	Jaro-Winkler	unmodified	0.945	0.637	0.761
		modified	0.946	0.64	0.763
	n-grams	unmodified	0.933	0.603	0.733
		modified	0.945	0.637	0.76
kNN	Levenshtein	unmodified	0.904	0.623	0.74
		modified	0.943	0.667	0.78
	Jaro-Winkler	unmodified	0.928	0.607	0.734
		modified	0.941	0.64	0.762
	n-grams	unmodified	0.948	0.663	0.78
		modified	0.949	0.677	0.79
SVM	Levenshtein	unmodified	0.922	0.597	0.725
		modified	0.93	0.62	0.744
	Jaro-Winkler	unmodified	0.924	0.61	0.735
		modified	0.926	0.623	0.745
	n-grams	unmodified	0.922	0.627	0.746
		modified	0.927	0.637	0.755



(a) Levenshtein (b) Jaro-Winkler (c) n-grams

Figure 2: Genuineness check, correctly classified, %



(a) Levenshtein (b) Jaro-Winkler (c) n-grams

Figure 3: Orientation check, correctly classified, %

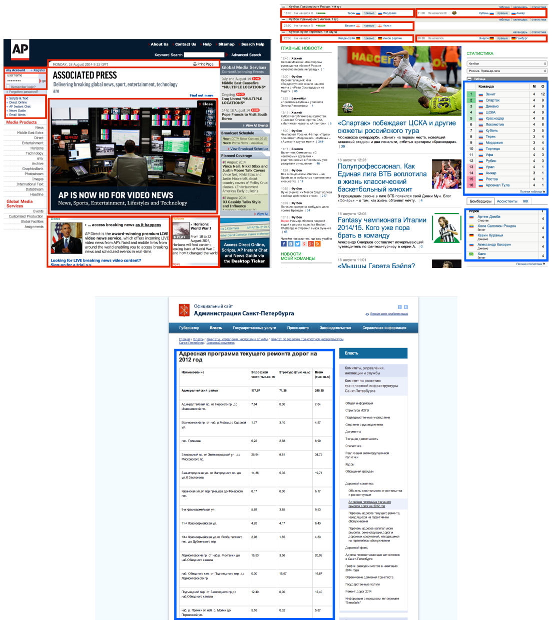

ASSOCIATED PRESS
AP IS NOW HD FOR VIDEO NEWS
Адресная программа текущего ремонта дорог на 2012 год

Figure 4: Table recognition results

Having analyzed the efficiency of machine learning methods with string metric mechanisms we decided to apply modified kNN in conjunction with Levenshtein distance during the genuineness check process and modified SVM in conjunction with Levenshtein distance during the orientation check process.

7.2 System Evaluation

The overall performance of the approach is defined as a product of the highest *F–Measure* of the genuineness check and the highest *Precision* of the orientation check, which results in *0.77* or *77%*. It is therefore indicating, that we are able to correctly extract knowledge at least from three of given four arbitrary tables.

On Fig. 4 the results of tables recognition are presented. All the tables that are marked in HTML code of web–pages as tables are coloured in red and blue. The tables were extracted from the websites of Associated Press[16], Sports.ru[17] and Saint Petersburg Government Portal[18]. According to the theory those tables might be divided in genuine and non-genuine (relevant or irrelevant) groups. It might be easily noted that the tables coloured in red use the tag for formatting reasons and do not contain appropriate table data. In contrast, the tables coloured in blue are relevant tables which data might be parsed and processed. ScraperWiki was able to extract all the red and blue tables. The user therefore should choose relevant tables for a further processing. As a counter to ScraperWiki the developed system was able to extract and process only blue genuine tables using appropriate heuristics and machine learning algorithms.

Taking into account the achieved results we consider the hypothesis suggested in Section 4 demonstrated. Indeed, unstructured data contains semantics. Hence, the next questions are raised. *How much semantics does unstructured data contain? Is there an opportunity to semantically integrate tables with other types of Web content?* Answering the questions will facilitate the shift from neglecting the tables towards close integration of all the Web content.

8. CONCLUSION AND FUTURE WORK

Automatic extraction and processing of unstructured data is a fast–evolving topic in science and industry. Suggested machine learning approach is highly effective in table structure analysis tasks and provides the tools for knowledge retrieval and acquisition.

To sum up, the system with the distinctive features was developed:

1. Automatic extraction of HTML tables from the sources specified by a user;
2. Implementation of string metrics and machine learning algorithms to analyze genuineness and structure of a table;
3. Automatic ontology generation and publishing of the extracted dataset;
4. The software takes advantages of Information Workbench API, enabling data visualization, sharing and linking

Future work concerns the question of ontology mapping. The datasets to be extracted might be linked with the already existing ones in the knowledge base dynamically during the main workflow, e.g. discovery of the same entities and relations in different datasets. It will facilitate the development of the envisioned Web of Data as well as wide implementation of Linked Open Data technologies.

9. ACKNOWLEDGMENTS

This work was partially financially supported by Government of Russian Federation, Grant 074-U01.

10. REFERENCES

[1] M. A. Babyak. What you see may not be what you get: a brief, nontechnical introduction to overfitting in regression-type models. *Psychosomatic medicine*, 66(3):411–421, 2004.

[2] M. Cafarella, E. Wu, A. Halevy, Y. Zhang, and D. Wang. Webtables: exploring the power of tables on the web. In *VLDB 2008*, 2008.

[3] E. Crestan and P. Pantel. Web-scale table census and classification. In ACM, editor, *WSDM 2011 Proceedings of the fourth ACM international conference on Web search and data mining*, pages 545–554, 2011.

[4] D. Embley, C. Tao, and S. Liddle. Automating the extraction of data from html tables with unknown structure. *Data and Knowledge Engineering - Special issue: ER*, 54:3–28, 2005.

[5] I. Ermilov, S. Auer, and C. Stadler. User-driven semantic mapping of tabular data. In *Proceedings of*

[16] Associated Press. Web: http://www.aptn.com/

[17] Sports.ru. Web: http://www.sports.ru/

[18] Roads repair dataset | Official Website of Government of Saint Petersburg. Web: http://gov.spb.ru/gov/otrasl/tr_infr_kom/tekobjekt/tek_rem/

the 9th International Conference on Semantic Systems, pages 105–112. ACM, 2013.

[6] W. Gatterbauer, P. Bohunsky, M. Herzog, B. Krüpl, and B. Pollak. Towards domain- independent information extraction from web tables. In *Proceedings of the 16th WWW.*, pages 71–80, 2007.

[7] P. Haase, M. Schmidt, and A. Schwarte. The information workbench as a self-service platform for linked data applications. In *COLD*. Citeseer, 2011.

[8] J. Hu, R. Kashi, D. Lopresti, and G. Wilfong. Evaluating the performance of table processing algorithms. *International Journal on Document Analysis and Recognition*, 4:140–153, 2002.

[9] M. Hurst. A constraint-based approach to table structure derivation. In *Proceedings of International Conference on Document Analysis and Recognition (ICDAR'03)*, pages 911–915, 2003.

[10] M. Kolchin and F. Kozlov. A template-based information extraction from web sites with unstable markup. In *Semantic Web Evaluation Challenge*, volume 475, pages 89–94, 2014.

[11] T. Mitchell. *Machine Learning*. McGraw-Hill Science/Engineering/Math, 1997.

[12] D. Mouromtsev, V. Vlasov, O. Parkhimovich, M. Galkin, and V. Knyazev. Development of the st. petersburg's linked open data site using information workbench. In *14th FRUCT Proceedings*, pages 77–82, 2013.

[13] L. Rokach and O. Maimon. *Data mining with decision trees: theory and applications*. World Scientific Pub Co Inc, 2008.

[14] A. Silva, A. Jorge, and L. Torgo. Design of an end-to-end method to extract information from tables. *International Journal of Document Analysis and Recognition (IJDAR)*, 8:144–171, 2006.

[15] Y. Tijerino, D. Embley, D. Lonsdale, Y. Ding, and G. Nagy. Towards ontology generation from tables. *World Wide Web*, 8:261–285, 2005.

[16] J. Tou and R. Gonzalez. *Pattern Recognition Principles*. University of Southern California, 1974.

[17] E. Ukkonen. Approximate string matching with q-grams and maximal matches. *Theoretical Computer Science*, pages 191–211, 1992.

[18] P. M. V. Crescenzi, G. Mecca. Roadrunner: Towards automatic data extraction from large websites. In *Proceedings of the 27th International Conference on Very Large Data Bases (VLDB'01)*, pages 109–118, 2001.

[19] K. Vorontsov. Machine learning methods. *Course at http://www. machinelearning. ru*, 2009.

[20] J. H. Y. Wang. A machine learning based approach for table detection on the web. In *Proceedings of the 11th WWW*, pages 242–250, 2002.