A Hybrid Technique for Annotating Book Tables

Asima Latif[1], Shah Khusro[1], Irfan Ullah[1], and Nasir Ahmad[2]
[1]Department of Computer Science, University of Peshawar, Pakistan
[2]Department of Computer Systems Engineering, University of Engineering and Technology Peshawar, Pakistan

Abstract: *Table extraction is usually complemented with the table annotation to find the hidden semantics in a particular piece of document or a book. These hidden semantics are determined by identifying a type for each column, finding the relationships between the columns, if any, and the entities in each cell. Though used for the small documents and web-pages, these approaches have not been extended to the table extraction and annotation in the book tables. This paper focuses on detecting, locating and annotating entities in book tables. More specifically it contributes algorithms for identifying and locating the tables in books and annotating the table entities by using the online knowledge source DBpedia Spotlight. The missing entities from the DBpedia Spotlight are then annotated using Google Snippets. It was found that the combined results give higher accuracy and superior performance over the use of DBpedia alone. The approach is a complementary one to the existing table annotation approaches as it enables us to discover and annotate entities that are not present in the catalogue. We have tested our scheme on Computer Science books and got promising results in terms of accuracy and performance*.

Keywords: *DBpedia spotlight, google snippets, table extraction, table annotation, table semantics, knowledge base.*

Received February8, 2015; accepted August 31, 2015

1. Introduction

A table is an important information entity that summarizes a given concept, phenomena or situation and augments reader understanding. However, usually lost in the ocean of text, it needs to be identified and retrieved [26]. In books and other technical documents tables are typically taken as a way of presenting large amount of information in a compact form and differ significantly in structure, flexibility, notation, representation and use [30]. For fulfilling the user needs, precise and accurate searching and ranking of the book tables is immensely required to convert this data into a structured and standard format so that the relevant information could be retrieved on the basis of these tables.

Tables are the effective means of arranging data in the form of rows and columns thus summarizing large amount of text into a small space. In the context of books the tables play very important role. Its compact view saves much time in understanding the underlying concepts. One can easily sort out the relations between the data by considering table's column headings and row values without digging into the large text. The arrangement of its columns and rows contains the hidden semantics. These hidden semantics can be captured by assigning a type to each column and establishing relations between the columns, if any. For searching, exploring and ranking book tables according to the user needs, there is a need of mechanisms and systems that can convert this data into a standard structured format. To the best of authors' knowledge, there has been no single effort in literature on annotating Portable Document Format (PDF) book table entities by using Google snippets and database lookups. The proposed solution devises a mechanism of enriching tables with additional annotations using online knowledge sources and search engine snippets to facilitate the searching and ranking of tables in a book. For prototypical implementation, the pdf2table tool was selected for the extraction of tables from books because of its good performance as compare to other PDF extraction tools [19]. The pdf2table is a good tool for structure identification but it is not very suitable for use with the large book documents. In this work, an algorithm was developed to enable pdf2table to identity tables in large book documents and extract all tables from them. The extracted tables are then passed to another algorithm which generates a type for each column. Finally, the relationships between the columns of the table are established using another algorithm. Rest of the paper is organized as follows. Section 2 describes the current state-of-the-art in table identification, extraction and annotation process. Section 3 presents the proposed solution and algorithms for table annotation in the domain of PDF books. Section 4 presents the experimental results obtained after implementing these algorithms. Finally, section 5 concludes the findings of this work and provides some recommendations for the future work.

2. Related Work

A lot of work on the extraction of tables out of web pages and other documents has been reported in literature [7, 8, 17, 20] however, because of the diversity in the table layouts, no work is perfect enough to extract all sorts of tables. Some existing PDF extraction tools includes freely available pdf2Hyper Text Markup Language (HTML), pdf2HTML by VeryPDF, Adobe Online Conversion Tool, TableSeer search engine, pdf2table and commercial products such as PDF-Analyzer by [31] and PDF Analyzer by Amyuni Technologies [4]. One exception is the work of Fang *et al.* [9] who evaluated their table extraction algorithm on an e-book dataset and a scientific document dataset. Comprehensive surveys about table detection approaches are provided by [7, 39]. Several approaches exist in the field of table recognition and extraction like predefined layout approach [24], heuristics-based approaches [14, 38], statistical approaches and hybrid of both the heuristics and statistical approaches [18]. The first relative research carried out on PDF for tables is by [38]. TableSeer is only search engine present so far for detecting, indexing and searching for tables in scientific PDF documents [23]. In PDF-Table Extraction and Recognition (TREX) an ontological approach [28] is based on segments in lines. Another approach is rule-based ontological approach to measure smallest distance between text chunks and to minimize wrong cluster creation [24]. Xonto is also an ontology-based system for semantic information extraction from PDF documents [27]. Border lines and rules for spotting text regions are considered by [9, 14]. A wrapper-based approach for table detection in PDF documents is presented in [14]. Use of both visual separators and irregular tables is considered in [9]. The evaluation of table detection algorithms is a big problem due to the lack of standard data sets and the ground truths [10]. Some evaluation methods are proposed by [6, 16, 32] however, all these methods cannot highlight detailed error descriptions with improvement hints. So each algorithm has its own limitations, and no single algorithm can provide ideal performance considering all evaluation metrics [10].

Semantic extraction from table is the main focus of tables-related research in general, however only very little work has been done on the table extraction and annotation in books. Table annotation is the identification of a correct type for each column and the relationships between columns is carried out in [13, 15, 21, 25, 34, 35, 36, 37]. In literature, tables are annotated either by pre-compiled catalogue of entities, types or relationships [21, 35, 36], Linked Open Data (LOD) datasets [25], or ontologies [15]. The problems of new entities annotation that do not exist in the reference catalogue is addressed by [2, 12, 30].To find out table associated main concepts using single 'entity column' values and rest of the column headers is done by Wang *et al*. [36] using Probase [37]. Using Yet Another Great Ontology (YAGO), the approach presented in [33] extracts multiple labels for each column in table based on maximum likelihood hypothesis. An automated graphical modeling interpretation of a table that focuses column headers, row values and relationships between columns is presented in [21]. However, none of the mentioned methods and proposed techniques provides domain independent interpretation of tables. All of these annotate tables using existing knowledge bases [21, 26, 36] ontologies [7] or content that is automatically extracted from web pages [22]. Thus only known entities annotation can be done, and the annotation of new entities is largely unexplored. An algorithm that claims to annotate such new entities not defined in data sources have been reported in [26], using text classifiers over snippets returned by search engines [29].

Sometimes entities are context-dependent and can be annotated with surrounding text [5, 11, 29] but this is not the case that always happens. Sometimes annotation of entities needs enrichment of semantic information from external knowledge sources [5, 11]. The approach presented in [5] uses WordNet for such annotation and Wang*et al*. [36] uses a document corpus for annotation of entities. The fattest open knowledge sources are YAGO [39], DBpedia [24] and Freebase [1]. Guo *et al.* [12] extract tables whose structure resembles to the RDF knowledge base and new tuples from tables are identified and stored in knowledge base. The approach of [32] is to recognize Google Fusion Tables (GFT) tables from specific field, by using ontology extract information from tables and fill the ontology with the extracted information. The work presented in [30] also resembles withIntegrate Tabular data and RDF Entities by schema Mapping (ITEM) tool. It identifies new entities by applying a trained text classifier on snippets returned by search engines (Microsoft Bing API). Another approach is labeling table columns using Wikipedia categories based on column content [3]. YAGO knowledge source is used for labeling cells and relation between columns with the help of probabilistic graphical model-based framework is presented in [3]. The approach in [32] proposes an algorithm that uses catalogue but is able to identify entities not recorded in the catalogue. Their work is close ITEM tool [12] for enriching knowledge bases. Tables can also be annotated semantically with surrounding text using domain ontology [2]. A comprehensive review of the methods and tools about PDF table detection, extraction and annotation can be found in [19].

Concluding state-of-the-art, relatively little work has been done on the semantics extraction from tables in general and almost no work has been done on extracting, interpreting and semantically annotating book tables using surrounding text and online knowledge bases for finding/ranking related tables in other books. The algorithm proposed in this work

[1] http://www.freebase.com

presents an effective technique for table annotation in books.

3. The Proposed Approach

The proposed approach is depicted in Figure 1. The steps involved are explained below. Firstly, it detects and extracts tables from the book and then searches all the entities of the table on DBpedia for finding the best annotation of values, relationship between table values including cells, rows, and table object. Finally the annotation of the missing entities is performed using snippets from the Google. Using Google snippets is not new rather it has been used by researchers for different purposes e.g., in [1] these snippets were used in finding the semantic resemblance or similarity among words and sentences. As book tables are small and usually have no more than 50 rows, we have selected pdf2table tool for extracting tables from PDF documents because of its good performance for small and simple tables. However, the problem with this tool is that, it cannot deal with large PDF documents efficiently. To deal with this limitation, we designed an algorithm to search book pages that contain tables which are extracted and passed to pdf2table tool for table extraction. Thus the problem of pdf2table of handling only small PDF documents is resolved as the proposed algorithm passes only those pages that contain tables. DBpedia Spotlight[2]provides a web service for annotating textual data. We use this service for finding semantics of table from DBpedia. For testing purposes, books from Computer Science domain were gathered. The process with a user interface where users input a book to the system and annotation process gets started by reading PDF book using iText [3] PDF library followed by scanning the book pages to identify pages containing tables. A copy of such pages is obtained and passed on to pdf2table to get the output.xml file. The output.xml file provides an xml version of the extracted table. Table cell values are obtained though XML stripping using XPath[4] library. The search string has multiple options for searching, such as, searching a single cell value, complete row values, complete column values or even complete table depending on the choice of user. The search string is checked for format compatibility with DBpedia search string. Signs, symbols, line separators, extra spaces, dots, spellings etc. are checked. All the stop words are removed from search string and encoding scheme is changed to UTF-8 for making search string compatible with DBpedia search. Each chunk of the search string is sent to DBpedia Spotlight service and the response labels for each chunk are stored in a file, however chunk with no response is considered as the missing entity which is dealt with Google snippets. The white spaces and stop words are removed from the obtained snippets and Jwnl[5] library is used for snippet text classification and stemming. For eliminating incorrect annotation we compare each DBpedia result with Google snippets based on the concept that a column contains homogeneous data types. Results are obtained by considering some threshold which is obtained by dividing number of occurrences of words by the total length of terms containing document. We find most common terms as annotated terms for the search query. We obtain the DBpedia labels for each cell and compare all labels for same column to get column annotation. Same process is considered for row annotation. At last we compare all the cells labels of complete table to get the table annotation. The annotated results are then showed to the user.

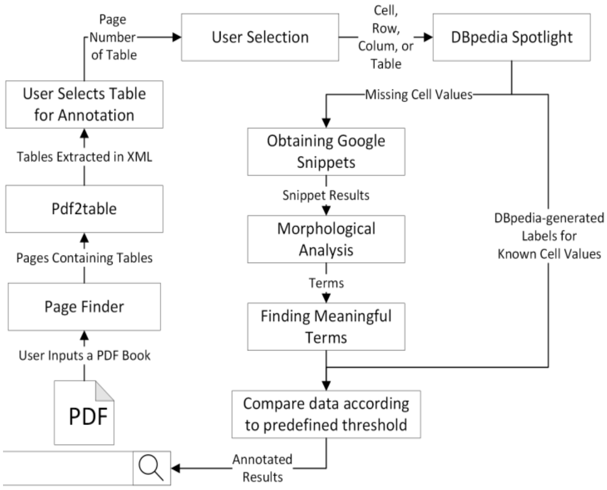

Figure 1. Table annotation process.

3.1. Algorithm for Finding Column/Row Labels

Algorithm 1 in proposed scheme is named as "FindLabels". This algorithm associates best labels with column. In Algorithm 1, "S" is table column containing entities. "s" represents single cell value of column. "N" top results gathered from DBpedia. "C" is combined DBpedia and Google snippets result set.

Algorithm 1: FindLabels (S, s, N, C)

1. *Let S be the set of entities in a table column.*
2. *For each s in S, a query is sent to DBpedia KB to get a ranked list of top N instances along with their types or class labels.*
3. *For each s, if DBpedia Knowledge Base returns no results search top 3 Google Snippets*
   Remove all stop words from snippet.

   Apply morphological analyzers on Google Snippet results
4. *Generate a set of class labels, combine snippets and DBpedia generated class labels. Let C be this set*
5. *Using TF/IDF assign each resulting term a weight.*
6. *Compare with threshold.*

[2] https://github.com/dbpedia-spotlight/dbpedia-spotlight/wiki
[3] http://itextpdf.com/
[4] http://www.w3.org/TR/xpath20/

[5] http://sourceforge.net/projects/jwordnet/

7. *Consider the best matched results as annotations for table entities.*

3.2. Relation Identification Between Columns

Algorithm 2 in the proposed scheme is named as "FindRelations" which finds the relations between columns. Let "Ci" is a string from column I and "Cj" is corresponding string from column J. "k" is the order pair of stringsfrom column I and J, and K is the set of k. "CR" is the set of DBpedia and/or Google Snippets extracted labels for all k in K. "r" is single label in set CR. "Score(r)" is the numberof occurrences of rin CR.

Algorithm 2: FindRelations (C_i, $C_{j,}$ k, CR, r, Score(r))

```
1. Let $C_i$ and $C_j$ be corresponding strings from columns I and J.
2. Let r (a label extracted from DBpedia and/or Google Snippets) from set CR is a candidate relation between $C_i$ and $C_{j.}$
3. for all r in CR do
     score =0;
         for k = 1 to lengthOf(CR) do
             if$CR_k$=r then
                 score= score+ 1
             end if
         end for
                 score(r)= score
     end for
4. Choose the relation r from CR that gets the highest score
```

Table 1. pdf2table Generated XML table from an e-Book.

	Import	Export
	XLS	**XLS3, XLS4, XLS5**
	WKS, WK1, WK3, WK4	WK1, WK3
	DIF, XDIF	DIF
	CSV	CSV
	SYLK	SYLK
17	text	text

4. Results and Evaluation

Table 1 presents a table from the book entitled "Teach Yourself Linux in 24 Hours by Share Reactor" which is extracted by pdf2table tool in xml format. The table headers, rows and column values are used for querying the Knowledge Base (KB). The results obtained from the knowledge base for each value of column are processed to predict a class label for every column. As shown in Table 2 the terms DIF, XDIF are missing in the knowledge base therefore Google snippets are obtained for entity annotation. Word occurrences are counted in snippets and checked them in lookup results to find out most suitable results. For example the processed result for first column values in Table 1 predominantly contains the terms like office, spread sheet, file formats etc. There are some ambiguous results too like radiobiology, agent etc. For handling these ambiguous results, semantic analysis is performed on such results to find out similarity between results. The final semantically annotated results are obtained as column labels. For example, the results obtained for the first column after passing to semantic analyzer and comparing with threshold are "Microsoft office, data serialization formats, spreadsheet file formats, and full-size vehicles" etc. as shown in Table 2. The threshold value (word occurrence/total document terms) obtained for the first column was 25%, therefore, the obtained results show that these are Microsoft Office spreadsheet file formats. Approach is also used for other columns; rows and even complete table could be passed to the system for annotation.

Table 2. Results from DBpedia and google snippets.

Column Terms	Known/Unknown Entities	Obtained Snippets/ DBpedia Results
DIF XDIF	Google Snippets result for missing entity	class can use convert data dif file format use share spreadsheets applications star office microsoft office etc class data description sdif type tag osc namespace function xvectxdiffxnit secantx1x0nmaxfuntoll secant iterations argxdif difference two successive value argfx nov 15 2013 size database import database different server import tmpnsmdbxdifvarnetscreenguisvrxdbinit now
CSV	DBpedia results for known entities	Units of radiation dose Radiobiology Christian Social People's Party organization political party agent Comma-separated values Spreadsheet file formats Data serialization formats mean of transportation Rear-wheel-drive vehicles Full-size vehicles
SYLK		Person American hip hop musicians SYmbolicLinK (SYLK) Spreadsheet file formats Microsoft Office
TEXT		Literature ASCII Acronyms ASCII Short Message Service Text messaging

4.1. Evaluation of Algorithm

We develop a prototype to test the validity of our algorithm. We run the prototypical version for 20 Computer Science books. Computer science books are selected because domain independent interpretation of book tables generates ambiguous results. So, the algorithm is currently handling only domain specific data set. In future we are planning to extend the approach for more domains. It is noteworthy that the algorithm works fine with domain specific datasets, although it can be extended to other domains. Domain independent interpretation of book tables generates ambiguous results. This is one of the reasons; we selected a particular domain which is Computer Science in our case. Similarly, number of books and number of book tables can be increased. We inputted each book separately and carefully recorded our observations. Pdf2table tool can identify table structure quite well but it often inaccurately merges rows and produces extra columns. It cannot support multiple column documents. Therefore, we enhanced the performance of tool for making it able to extract tables out of book documents. All extracted tables are containing no more than 50 rows. Most of the tables in our book dataset contain numerical data. This numerical data annotation is not considered by our algorithm because of ambiguous results from DBpedia and Google snippets. The application performs well with tables having more than two columns.

We examined that our algorithm outperforms existing table annotation techniques by considering missing entities too in annotation processes. Our dataset table contains no more than 50 rows normally because book tables are usually small. For processing a row the proposed algorithm takes 0.5 seconds therefore this algorithm is suitable for dealing with book tables. The run-time of the algorithm is dependent upon by the potential time required to connect to the search engine and DBpedia Spotlight service. The accuracy of the algorithm is also dependent upon pdf2table tool generated table results and DBpedia generated interpretations.

For obtaining accurate evaluation we first annotate each table by one person then compare it against our algorithm for finding correct/missed/incorrect (C/M/I) concepts and relationships. From book dataset 30 tables had 150 columns out of which 100 columns were numerical, which are not considered in this research because DBpedia and search engine snippets are poor in such data annotation producing ambiguous results. Remaining 50 columns contain 250 instances by DBpedia correctly classified instances C were 127, incorrectly classified instances I are 35 and missed M were 88. After gathering and classifying Google snippets for 88 missing entities for the same 30 tables we get 50 correctly classified instances 30 incorrect classification. No result/very short snippet/with no meaningful information from snippet/non-English result is considered as missing informationM, therefore we get M for 8 instances. The ambiguous results from Google snippets are ignored by comparing threshold value for obtained terms. Incorrect results generated during process are rejected because of no homogeneity with other obtained annotated result. By increasing number of tables to 300 we had 500 non-numeric columns which had 2500 instances. Correctly classified instances C by DBpedia were 1800, incorrectly classified instances I were 300 and missed M were 400. Results obtained from Google Snippets for missed entities contain 295 correct classifications and 90 ambiguous results and for 15 instances Google generated no snippets. After this we compute precision P using Equation (1), recall R using Equation (2) and f-measure F using Equation (3). These equations are as follows:

$$P = \frac{C}{(C+I)} \tag{1}$$

$$P = \frac{C}{(C+I)} \tag{2}$$

$$P = \frac{C}{(C+I)} \tag{3}$$

Table 3. Proposed algorithm evaluation table.

ANNOTATION	NO OF TABLES	P	R	F
DBpedia Annotation	For 50 Tables of Book Dataset	0.78	0.59	0.67
DBpedia + Snippet Annotation		0.73	0.95	0.82
DBpedia Annotation	For 150 Tables of Book Dataset	0.85	0.81	0.82
DBpedia + Snippet Annotation		0.84	0.96	0.89

Table 3 shows that for only DBpedia annotated results precision, recall and f-measure was low, but after including Google Snippets in annotation process for handling missing entities we find significant change in recall and f-measure.

5. Conclusions and Future Work

We have contributed an algorithm that annotates entities in PDF book tables which a complimentary algorithm for existing table annotation approaches as the algorithm is able to discover and annotate entities that are not present in catalog. To the best of our knowledge this is the first such work that deals with extraction and annotation of tables in PDF books using online knowledge bases and search engine snippets. Although the accuracyof pdf2table has been enhanced by making it able to extract tables from large documents like books but still the limitations of pdf2table tools, DBpedia and Google Snippets' ambiguous results affect the performance of the proposed algorithm. The proposed technique is limited only to PDF books but could be extended to other formats. Similarly availability and accuracy of the proposed technique for table annotation and enrichment will be able to become complimentary technique for the previous techniques.

As future work, we intend to integrate our algorithm with annotation of book tables using book table surrounding textand well as table metadata like the approach in [11] and comparing book annotated results with our algorithm in order to handle ambiguous results produced by snippets. We also intend to extend the algorithm for annotation of numeric columns. Furthermore, for improving the scalability of our algorithm we intend to improve algorithm performance for several thousand types of entities. It has been found from experimental results that the algorithm is able to detect concepts and relations automatically with good accuracy. These results encourage us to integrate the algorithm in our future book search engine for book table search, annotation and ranking purposes. Its main objective is to facilitate discovery of similarity in book topics, headings, subheadings based on table comparisons.

References

[1] Adhikesavan K., "An Integrated Approach for Measuring Semantic Similarity between Words and Sentences using Web Search Engine," *The International Arab Journal of Information Technology*, vol. 12, no. 6, pp. 589-596, 2015.

[2] Alrashed S., "Finding Hidden Semantics of Text Tables," *in Proceedings ofInternational Workshop on Document Analysis Systems*, Heidelberg, pp. 449-461, 2006.

[3] Amin M., Bhattacharjee A., and Jamil H., "Wikipedia Driven Autonomous Label Assignment in Wrapper Induced Tables with Missing Column Names," *in Proceedings of the 2010 ACM Symposium on Applied Computing*, Switzerland, pp. 1656-1660, 2010.

[4] Amyuni T., Amyuni Technologies Inc., Montreal, Available: http://blog.amyuni.com/?p=1062, Last Visited, 2018.

[5] Cimiano P. and Volker J. "Towards Large-scale, Open-domain and Ontology-based Named Entity Classification," *in Proccedings ofInternational Conference on Recent Advances in Natural Language Processing*, Bulgaria, pp. 166-172, 2005.

[6] E-Silva A., "New Metrics for Evaluating Performance in Document Analysis Tasks_Application to the Table Case," *in Proccedings of 9th International Conference on Document Analysis and Recognition*, Parana,pp.481-485, 2007.

[7] E-Silva A., Jorge A., and Torgo L., "Design of an end-to-end Method to Extract Information from Tables," *International Journal of Document Analysis and Recognition*, vol.8, no. 2, pp. 144-171, 2006.

[8] Embley D., Lopresti D., and Nagy G., "Notes on Contemporary Table Recognition," *in Proccedings of Document Analysis Systems VII*, Berlin, pp. 164-175, 2006.

[9] Fang J., Gao L., Bai K., Qiu R., Tao X., and Tang Z., "A Table Detection Method for Multipage PDF Documents via Visual Seperators and Tabular Structures," *in Proceedings of International Conference on Document Analysis and Recognition*, pp. 779-783, 2011.

[10] Fang J., Tao X., Tang Z., Qiu R., and Liu Y., "Dataset, Ground-Truth and Performance Metrics for Table Detection Evaluation," *in Proceedings of 10th IAPR International Workshop on Document Analysis Systems*, Gold Cost, pp. 445-449, 2012.

[11] Fleischman M. and Hovy E. "Fine Grained Classification of Named Entities," *in Proceedings of 19th International Conference on Computational Linguistics*, Stroudsburg, pp. 1-7, 2002.

[12] Guo X., Chen Y., Chen J., and Du X. "ITEM: Extract and Integrate Entities from Tabular Data to RDF Knowledge Base," *in Proceedings of 13th Asia-Pacific Web Conference on Web Technologies and Applications*, Heidelberg, pp. 400-411, 2011.

[13] Han L., Finin T., Parr C., Sachs J., and Joshi A.,"RDF123: From Spreadsheets to RDF," *in Proceedings of the 7th International Conference on The Semantic Web*, Karlsruhe, pp. 451-466, 2008.

[14] Hassan T. and Baumgartner R., "Table recognition and understanding from pdf files," *in Proceedings of 9th IEEE International Conference on Document Analysis and Recognition*,Parana, pp. 1143-1147, 2007.

[15] Hignette G., Buche P., Dibie-Barthélemy J., and Haemmerlé O.,"Fuzzy Annotation of Web Data Tables Driven by a Domain Ontology," *The Semantic Web: Research and Applications*, Crete, pp. 638-653, 2009.

[16] Hu J., Kashi R., Lopresti D., and Wilfong G., "Evaluating the Performance of Table Processing Algorithms," *International Journal on Document Analysis and Recognition*, vol. 4, no. 3, pp.140-153, 2002.

[17] Hurst M., "Towards a Theory of Tables," *International Journal of Document Analysis and Recognition*, vol. 8, no. 2-3, pp. 123-131, 2006.

[18] Jiang D. and Yang X., "Converting PDF to HTML Approach based on Text Detection," *in Proccedings of 2nd International Conference on Interaction Sciences: Information Technology, Culture and Human*,Seoul, pp. 982-985, 2009.

[19] Khusro S., Latif A., and Ullah I., "On Methods and Tools of Table Detection, Extraction and Annotation in PDF Documents," *Journal of Information Science*, vol. 41, no. 1, pp. 41-57, 2014.

[20] Kieninger T. and Dengel A., "A Paper-to-Html Table Converting System," *Document Analysis Systems*, vol. 98, pp. 1-10, 1998.

[21] Limaye G., Sarawagi S., and Chakrabarti S., "Annotating and Searching Web Tables Using Entities, Types and Relationships," *in Proccedings of VLDB Endowment*, vol. 3, no. 1, pp. 1338-1347, 2010.

[22] Liu Y., Tableseer: Automatic Table Extraction, Search, and Understanding, Pennsylvania State University, 2009.

[23] Liu Y., Bai K., Mitra P., and Giles C., "TableSeer:Automatic Table Metadata Extraction and Searching in Digital Libraries," *in Proccedings of 7th ACM/IEEE-CS joint conference on Digital libraries*, Vancouver,pp. 91-100, 2007.

[24] Mohemad R., Hamdan A., Othman Z., and Noor

N., “Automatic Document Structure Analysis of Structured PDF Files,” *International Journal of New Computer Architectures and their Applications*, vol. 1, no. 2, pp. 404-411, 2011.

[25] Mulwad V., “DC Proposal : Graphical Models and Probabilistic Reasoning for Generating Linked data from tables,” *in Proccedings of The Semantic Web-ISWC*, Heidelberg, pp. 317-324, 2011.

[26] Mulwad V., Finin T., and Joshi A., “Generating Linked Data by Inferring the Semantics of Tables,” *in Proccedings of The workshop Very Large Data Search VLDS*, pp. 17-22, 2011.

[27] Oro E. and Ruffolo M.. “Xonto: An Ontology-based System for Semantic Information Extraction from pdf Documents,” *in Proccedings of 20th IEEE International Conference on Tools with Artificial Intelligence*, Dayton, pp. 118-125, 2008.

[28] Oro E. and Ruffolo M., “PDF-TREX: An Approach for Recognizing and Extracting Tables from PDF Documents,” *in Proccedings of 10th International Conference on Document Analysis and Recognition*, Barcelona, pp. 906-910, 2009.

[29] Quercini G. and ReynaudC., “Des Données Tabulaires à RDF: l'extraction de Données de Google Fusion Tables,” *in Proccedings of Atelier Ontologies et Jeux de Données pour évaluer le Web Sémantique*, Paris, 2012.

[30] Quercini G. and Reynaud C., “Entity Discovery and Annotation in Tables,” *in Proccedings of 16th International Conference on Extending Database Technology*,Genoa, pp. 693-704, 2013.

[31] Schmoekel I., “PDF-Analyzer Pro 4.0. Software-Development and Distribution,” *Informer Technologies*, vol. 1, pp. 1-11, 2010.

[32] Shahab A., Shafait F., Kieninger T., and Dengel A., “An Open Approach Towards the Benchmarking of Table Structure Recognition Systems,” *in Proccedings of 9th IAPR International Workshop on Document Analysis Systems*,Massachusetts, pp. 113-120, 2010.

[33] Suchanek F., Kasneci G., and Weikum G.,“Yago: A Core of Semantic Knowledge,” *in Proccedings of 16th International World Wide Web Conference*, Alberta, pp. 697-706, 2007.

[34] Van Assem M., Rijgersberg H., Wigham M., and Top J., “Converting and Annotating Quantitative Data Tables,” *in Proccedings of 9th International Semantic Web Conference on The Semantic Web*, Heidelberg, pp. 16-31, 2010.

[35] Venetis P., Halevy A., Madhavan J., Paşca M., Shen W., Wu F., Miao G., and Wu C., “Recovering Semantics of Tables on the Web,” *VLDB Endowment*, vol. 4, no. 9, pp. 528-538, 2011.

[36] Wang J., Wang H., Wang Z., and Zhu K., “Understanding Tables on the Web,” *Conceptual Modeling*, Heidelberg, pp. 141-155, 2012.

[37] Wu W., Li H., Wang H., and Zhu K., “Towards a Probabilistic Taxonomy of Many Concepts,” Technical Report, Microsoft Research Asia, 2011.

[38] Yildiz B., Kaiser K., and Miksch S., “pdf2table: A Method to Extract Table Information from PDF Files,” *in Proceedings of the 2nd Indian International Conference on Artificial Intelligence*, Pune, pp. 1773-1785, 2005.

[39] Zanibbi R., Blostein D., and Cordy J., “A Survey of Table Recognition,” *Document Analysis and Recognition*, vol. 7, no. 1, pp. 1-16, 2004.

Asima Latif obtained her BS and MS degrees in Computer Science from the Department of Computer Science, University of Peshawar, Pakistan. Her research interests include information retrieval, information extraction, information semantics and search engines.

Shah Khusro received his Ph.D. degree from Vienna University of Technology, Vienna, Austria. He is currently working as Professor at the Department of Computer Science, University of Peshawar, Pakistan. His research interests include Web Semantics, Web Engineering, Information Retrieval, Web based Systems, Ambient Assisted Living, and Mobile Technology for People with Special Needs.

Irfan Ullah received his MS degree in Web Engineering from the Department of Computer Science, University of Peshawar, Pakistan in 2014. He is now pursuing his PhD from the same institute. His research interests include Web Semantics, Linked Open Data, Information Retrieval, Web Engineering and Digital Libraries. He is also working as Assistant Professor at Shaheed Benazir Bhutto University, Sharingal, Pakistan.

Nasir Ahmad received his PhD degree from Loughborough University, UK. Currently he is working as Assistant Professor at the Department of Computer Systems Engineering, University of Engineering and Technology Peshawar, Pakistan. His Research interests include Speech and Video Processing and Digital Signal Processing, Pattern Recognition, and Machine Learning.