

Table extraction, analysis, and interpretation: the current state of the TabbyDOC project

Alexey Shigarov¹, Nikita Dorodnykh¹, Andrey Mikhailov¹, Viacheslav Paramonov¹ and Alexander Yurin¹

¹*Matrosov Institute for System Dynamics and Control Theory, Siberian Branch of the Russian Academy of Sciences, 134 Lermontov St, Irkutsk, 664033, Russian Federation*

Abstract

The freely available tabular data represented in various digital formats, such as print-oriented documents, spreadsheets, and web pages, are a valuable source to populate knowledge graphs. However, difficulties that inevitably arise with the extraction and integration of the tabular data often hinder their intensive use in practice. TABBYDOC project aims at elaborating a theoretical basis and developing open software for data extraction from arbitrary tables. Previously, it was devoted to the following issues: (i) table extraction tables from print-oriented documents, (ii) data transformation from spreadsheet tables to relational and linked data. This paper summarizes the project's results that are intended for the following tasks: (i) automation of fine-tuning artificial neural networks for table detection in document images, (ii) a synthesis of programs for spreadsheet data transformation driven by user-defined rules of table analysis and interpretation, and (iii) generating RDF-triples from entities extracted from relational tables.

Keywords

table understanding, table extraction, table analysis, table interpretation, spreadsheet data extraction, data integration

1. Introduction

A large volume of tabular data represented in various digital formats, such as print-oriented documents (e.g. PDF), spreadsheets (e.g. Excel), flat file databases (e.g. CSV), and web-pages (e.g. HTML), is freely available in the Web. Such data can be a valuable source for populating knowledge graphs [1, 2]. However, difficulties that inevitably arise with extraction and integration of the tabular data often hinder their intensive use in practice. Typically, they are not accompanied by explicit semantics necessary for the machine interpretation of their content, as conceived by their author. Since such data are unstructured or semi-structured, first they should be transformed to a structured representation with a formal model.

The general-purpose tools for document converting, text mining, or web-scraping typically do not take into account the relational nature of tabular data, whereas table-specific tools enable a much more effective implementation of tabular data processing [3, 4]. They shorten the

ITAMS 2021: Information Technologies: Algorithms, Models, Systems, September 14, 2021, Irkutsk, Russia


✉ shigarov@icc.ru (A. Shigarov); tualatin32@mail.ru (N. Dorodnykh); mikhailov@icc.ru (A. Mikhailov); slv@icc.ru (V. Paramonov); iskander@icc.ru (A. Yurin)

🌐 <http://td.icc.ru> (A. Shigarov)

🆔 0000-0001-5572-5349 (A. Shigarov); 0000-0001-7794-4462 (N. Dorodnykh); 0000-0003-4057-4511 (A. Mikhailov); 0000-0002-4662-3612 (V. Paramonov); 0000-0001-9089-5730 (A. Yurin)



© 2021 Copyright for this paper by its authors. Use permitted under Creative Commons License Attribution 4.0 International (CC BY 4.0).

 CEUR Workshop Proceedings (CEUR-WS.org)

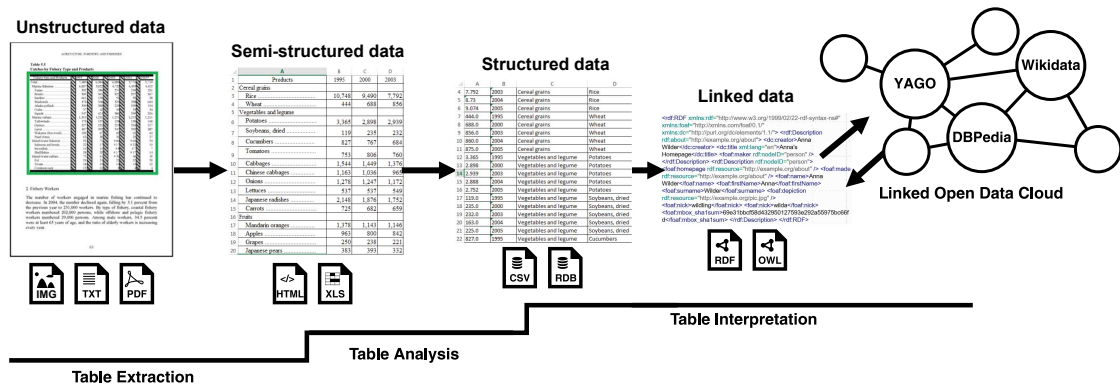


Figure 1: Goals of the TabbyDOC project: table extraction, table analysis, and table interpretation.

software development time hiding inessential details and focusing on table specifics. This is especially important in cases where it is necessary to develop software for the mass processing of tabular data in a short time and with a lack of resources.

TABBYDOC project aims at elaborating a theoretical basis and developing open software for data extraction from tables (Fig. 1). It covers the following tasks of the *table understanding*: (i) *table extraction* (i.e. detection of table bounding boxes and recognition of their cells in print-oriented documents), (ii) *table analysis* (i.e. extraction of interrelated functional data items from recognized tables), and (iii) *table interpretation* (i.e. mapping extracted data items to an external vocabulary).

The rest of the paper summarizes TABBYDOC’s results as follows: (Section 2) an automation of fine-tuning artificial neural networks for table detection in document images, (Section 3) cleansing table structure (erroneously split cells of header), (Section 4) a synthesis of programs for spreadsheet data transformation driven by user-defined rules of table analysis and interpretation, and (Section 5) generating RDF-triples from entities extracted from relational tables.

2. Table extraction

Schreiber et al. [5] first discovered that deep learning (DL) based “*object detection*” [6] in natural scene images can be successfully applied to the *table detection* in document images via the transfer learning. Their promising approach is based on fine-tuning pretrained artificial neural networks (ANN) and the well-known “Faster R-CNN” architecture [7]. However, this process includes many routine manipulations to prepare training data. We addressed the issue of shortening expert efforts by unifying existing collections of ground-truth data as well as by transforming and augmenting samples.

To simplify the development of ANN-models for table detection, we designed a workflow [8] (Fig. 2) that covers the following steps: (i) unifying samples (document images) from diverse annotated collections to “Pascal VOC”¹ format; (ii) blurring and augmenting image samples via

¹<http://host.robots.ox.ac.uk/pascal/voc>

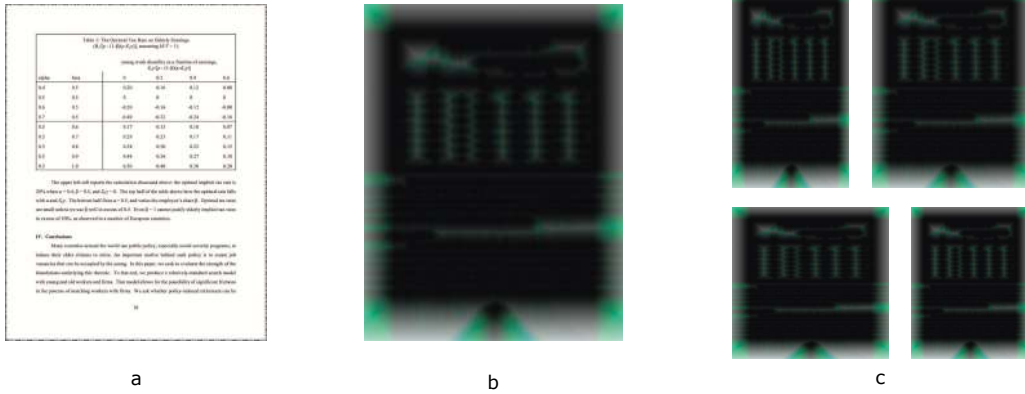


Figure 2: Transformation and augmentation of samples: an origin page – (a), after image blurring – (b), after affine transformation (c).

affine transformation (Fig. 2); (iii) converting samples from “Pascal VOC” to TFRecord² format; (iv) training ANN-model by using TensorFlow³, the open source platform for deep learning; (v) evaluating the target ANN-model on a competitive dataset.

The workflow was automated by DL4TD⁴ [8], a set of Python scripts. This allowed us to prepare about 19K annotated samples without the augmentation that were collected from five freely available datasets, namely, UNLV⁵, Marmot⁶, “ICDAR2017 POD” [9], SciTSR⁷ [10], and “ICDAR2019 cTDaR”⁸ [11]. We also converted them in the “Pascal VOC” format. The proposed solution enabled reduction of routine manipulations, i.e. expert efforts for trying various training options.

The automation was used to choose training options. As a result, more than 50 ANN models with admissible quality were created. We selected the ANN model with the best accuracy among all that we trained. It was incorporated into TABBYPDF⁹ [12, 13], our tool for table extraction from untagged PDF documents. This solution was evaluated with the “ICDAR 2013 Table Competition”¹⁰ methodology [14] and dataset [15]. The precision reached only 0.8651 while recall was 0.9795.

It should be noted that false positives among predictions made by the selected ANN-model sufficiently degraded the precision. We proposed to verify predictions to reduce the false positives by probing the text arrangement inside the bounding box of a candidate table. The idea consists in the following: (i) to segment text of a table candidate into blocks each of which represents a paragraph or cell, (ii) to compose a graph from the text blocks, and (iii) to analyze the graph and to probe features indicating that the candidate is probably a table or not.

²https://www.tensorflow.org/tutorials/load_data/tfrecord

³<https://www.tensorflow.org>

⁴<https://github.com/tabbydoc/dl4td>

⁵http://tc11.cvc.uab.es/datasets/DFKI-TGT-2010_1

⁶<https://www.icst.pku.edu.cn/cpd/sjzy>

⁷<https://github.com/Academic-Hammer/SciTSR>

⁸https://github.com/cndplab-founder/ICDAR2019_cTDaR

⁹<https://github.com/tabbydoc/tabbypdf2>

¹⁰<https://www.tamirhassan.com/html/competition.html>

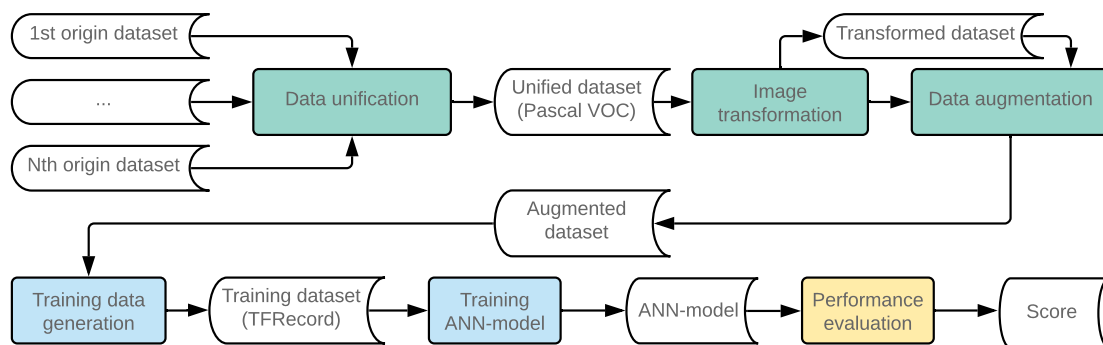


Figure 3: DL4TD workflow creates ANN-models for table detection in document images.

To implement the approach mentioned above, we adapted the T-RECS [16, 17] algorithms for “*clustering of word blocks*” in a plain-text document, by adding PDF-specific constraints. The main changes were the following: (i) composing text blocks by using the rendering order and formatting of text and graphics, (ii) calculating interline spacing. We also supplemented T-RECS algorithms with a new one for eliminating erroneously glued blocks due to subscript and superscript fonts. All the algorithms were implemented as part of TABBYPDF.

As a result, the adapted algorithms enabled representing table candidates as graphs of connected text blocks. We selected a set of 32 features for classifying predictions and trained a binary classifier based on the Random Forest classifier. The hyperparameters were tuned by using the Randomized Search. To train a model with a satisfactory accuracy, first we conducted a number of experiments with changing training datasets. As a result, we prepared the reference dataset as follows. Our ANN-model predicted some table candidates in PDF documents. Two experts manually assigned one of the two labels, either “*Table*” or “*Not table*”, to each of these table candidates. The graphs were generated for both positive and negative samples. The target classifier was trained on this dataset.

The verification allowed us to increase the precision up to 0.9703 on the competition part of “ICDAR 2013 Table Competition”, which is 10% higher than the initial measurement (0.8651). The approach showed that the graph-based table verification phase can significantly improve results obtained from the deep learning-based table prediction phase. More details can be found in our previous paper [18].

3. Table cleansing

The table analysis stage assumes that each physical (machine-readable) cell corresponds to one logical (human-readable) cell (Fig. 3). However, spreadsheets actually do not guarantee the fulfillment of this assumption, so it is often violated in real-world tables. We addressed this issue with HeadRecog, a rule-based algorithm for correcting the physical structure of cells in column headers by using visual borders [19, 20]. The algorithm matches physical cells with logical ones that are highlighted by visual borders (Fig. 3). The HEADRECOG was implemented

	A	B	C	D
1				
2	Items		Total	
3		1990	1995	2000

a

	A	B	C	D
1				
2	Items	Total		
3		1990	1995	2000

b

Figure 4: An example of the cell structure before (a) and after (b) the table cleansing.

as a part of TABBYXL¹¹, our software platform for rule-based transformation spreadsheet data from arbitrary to relational tables [21]. It was experimentally demonstrated that the correctness of the cell structure significantly affects the effectiveness of table analysis and interpretation by reducing a number of errors.

It should be highlighted that the existing methods for the cell structure recognition are mainly image-based. They focus on a lower-level representation of documents such as bitmap images. Unlike them, our solution deals with the high-level representation of spreadsheets. This help avoid data loss that inevitably accompanies converting to a lower-level format required for the use of the image-based methods.

We developed and tested the software tool (HEADRECOD) for correcting a physical structure of headers in spreadsheet tables according to their visual clues. This tool realized the algorithms which were previously made in the project. We used the assumption of K. Broman [22] about filling header cells of a table that states that empty cells are used for decoration only. If a cell is empty, but the corresponding column-decorator cannot be found, this cell should be merged with one of the non-empty neighboring cells. The decision if the merging is possible is based on the analysis results about cells mutual disposition, their styles and visual borders.

The efficiency of the implemented solution (HEADRECOG) was demonstrated on real-life tables from the open access corpora SAUS (“The 2010 Statistical Abstract of the United States”). We prepared SAUS200 [23], the subset of 200 tables randomly selected by G. Nagy and published by Z. Chen and M. Cafarella¹². It should be noted that we have made minor improvements to these tables. We have deleted hidden columns that contained service (markup) data. To automatically test the implemented software, a set of ground-truth tables corresponding to the subset of SAUS200 was also prepared. Each of the ground-truth tables has a correct structure of header cells.

To evaluate the performance of the automatic cells correction, a utility was implemented. It allows one to compare the header part of two tables and calculate the difference in the cells structure and their content. The total number of cells in the original SAUS200 tables header is 8028 compared to 3768 in the ground-truth tables headers. The use of HEADRECOG brings the number of cells to 3795 items. Comparing the cells adjusted automatically and manually, we obtained a complete match for 93.2% of the cases.

¹¹<https://github.com/tabbydoc/tabbyxl>

¹²<http://dbgroup.eecs.umich.edu/project/sheets/datasets.html>

4. Table analysis

Other advancements in TABBYXL [21, 24, 25] concerned the development of CRL, our domain-specific language (DSL) for specifying production rules for analysis and interpretation of tables [26, 27, 28]. CRL enables determination of queries (conditions) and operations (actions) that are necessary to develop programs for the spreadsheet data transformation from an arbitrary to a canonical form (Fig. [?]). CRL-rules map a physical structure of cells (properties of layout, formatting, and content) to a logical structure (interrelated functional data items such as entries, labels, and categories). In comparison with general-purpose rule languages (e.g. DROOLS¹³, JESS¹⁴, or RULEML¹⁵), the advanced version of the language enables expression of rulesets without any instructions for management of the working memory (such as updates of modified facts, or blocks on the rule re-activation). This provides syntactically simplified declaration of the right-hand side of CRL-rules. End-users can focus more on the logic of the table analysis and interpretation than on the logic of the rule management and execution.

We developed an interpreter of CRL-rules. It provides translation of CRL-rulesets (declarative programs) to Java source code (imperative programs). The generated source code is ready for compilation and building of executable programs for the domain-specific spreadsheet data extraction and transformation. Many of the existing solutions with similar goals use predefined table models embedded into their internal algorithms. Such systems usually support only a few widespread layout types of tables. Unlike them, our software platform defines a general-purpose table model that does not restrict layout types. It allows expressing user-defined layout, style, and text features of arbitrary tables in external CRL-rules. In comparison to our competitors, we support not only widespread layout types of arbitrary tables, but also some specific types. The empirical results show that our software platform can be successfully used in development of programs for the spreadsheet data extraction and transformation.

Additionally, we developed a generator of MAVEN-projects to build executable applications for transformation of spreadsheet tables to a canonical form. The generated applications have a basic functionality of spreadsheet data canonicalization and can be applied without additional programmer efforts. This can be useful in diverting software development towards rule-driven data extraction and transformation from spreadsheet tables. The implemented tools were integrated into the final release of TABBYXL¹⁶. Its source code was published in open access, including the accompanying wiki-documentation¹⁷, software demos in formats of Docker-container¹⁸, and CodeOcean-capsule¹⁹.

An illustrative example of using the updated TABBYXL when extracting data from real-life statistical tables was prepared. The example includes: the original SAUS200 tables, CRL-rules for table analysis and interpretation that enables generation of an executable Java application for converting these tables to a canonical form, ground-truth data for automatic performance

¹³<https://www.drools.org>

¹⁴<https://jess.sandia.gov>

¹⁵<http://ruleml.org>

¹⁶<https://github.com/tabbydoc/tabbyxl/releases/tag/v1.1.1>

¹⁷<https://github.com/tabbydoc/tabbyxl/wiki>

¹⁸<https://hub.docker.com/r/tabbydoc/tabbyxl>

¹⁹<https://codeocean.com/capsule/5326436/tree/v1>

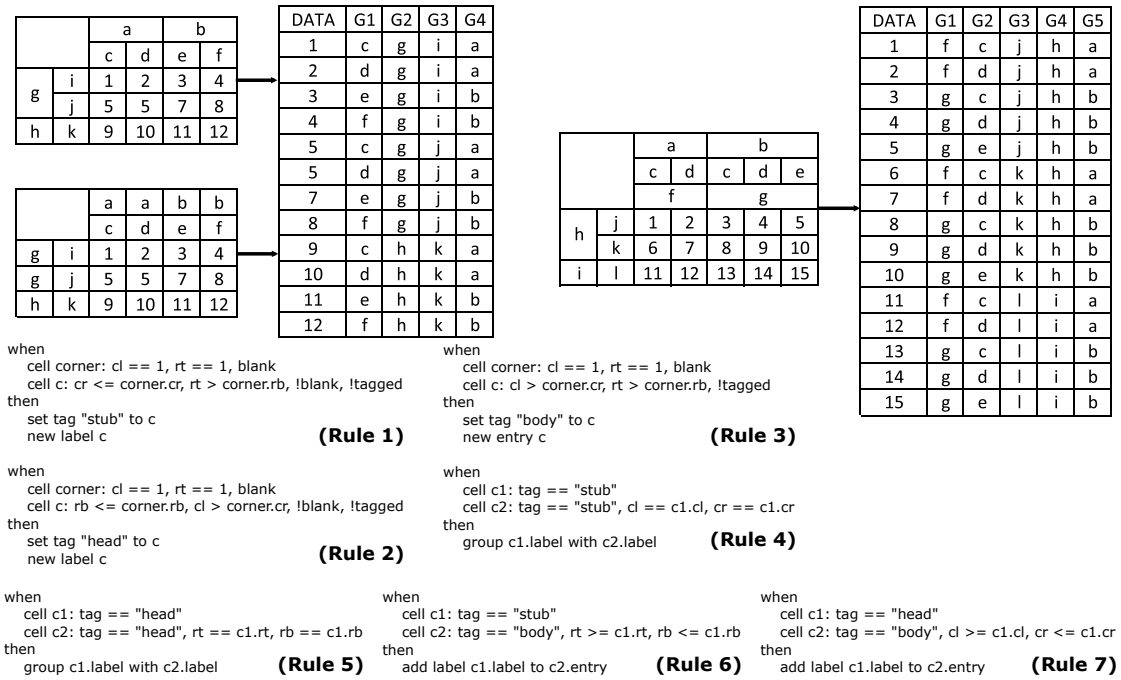


Figure 5: An example of CRL-rules for transforming tables (on the left) to a canonical form (on the right).

evaluation (of the generated Java application), as well as a step-by-step description of the workflow. The accompanying material was published as an open-access archive²⁰.

The performance evaluation was conducted for three cases of preprocessing of the SAUS200 dataset, namely: (i) original tables; (ii) tables corrected automatically using the HEADRECOG algorithms proposed by us; (iii) tables manually corrected by three experts. For the original data, the following indicators were achieved: the F_1 -score of data extraction (occurrences and labels) reached 84%, the F_1 -score of relationship extraction (such as “entry-label” and “label-label”) reached 72.4%. An automatic structure correction improved the F_1 -score of data extraction to 85.5% and the F_1 -score of relationship extraction to 82.2%. With the expert correction, they reached 96.3% and 93.7%, respectively. All materials and steps to reproduce this experiment are available at the dataset [23].

5. Table interpretation

We developed TABBYLD²¹ [29], a tool for the semantic interpretation of tabular data by using a cross-domain knowledge graph, namely, DBPEDIA²². It implements the following functions: (i) *Data cleansing* transforms origin tabular data to a canonical form which is suitable to be used in

²⁰<https://doi.org/10.6084/m9.figshare.14371055.v2>

²¹<https://github.com/tabbydoc/tabbyld>

²²<https://www.dbpedia.org>

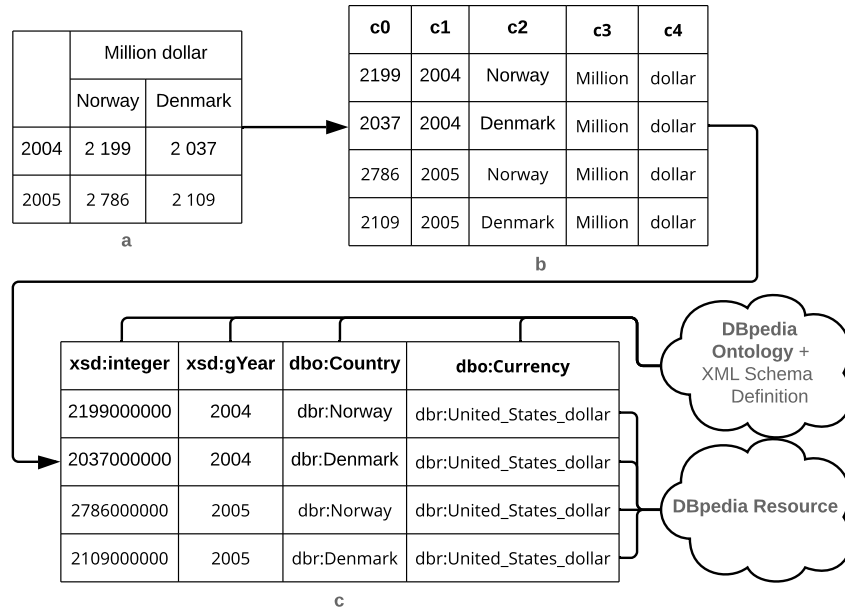


Figure 6: Table representation in the interpretation process: origin (a), canonicalized (b), annotated (c).

SPARQL²³ queries to look up the candidate KG-entities (Fig. 5 a, b). (ii) *Column type classification* assigns one of two types, either *literal* (containing numbers, dates, etc.) or *categorical* (containing entity mentions), to each column. (iii) *Entity linking* matches cell values with KG-entities of DBPEDIA. (iv) *Table annotation* enriches canonicalized tables with links to DBPEDIA’s resources (Fig. 5 c). (v) *Linked data generation* represents annotated data as RDF²⁴-triplets.

One of the important issues studied in our work was the named entity disambiguation, when two or more candidate KG-entities that were looked up from a knowledge graph (KG) match the same mention (surface form) from a table. To resolve such conflicts, we use semantic similarity by an edit distance, headings similarity, as well as consistency and context of the candidate KG-entities. In comparison with others, we employ an additional metric estimated by relationships between the candidate KG-entities and a recognized KG-class. The final decision is made by an aggregation of the metrics based on linear convolution.

The proposed solution was evaluated on the “T2Dv2 Gold Standard” dataset²⁵ that includes 779 reference tables from the “Web Data Commons” corpus. We used a subset of randomly selected 237 tables and 150 negative samples from T2Dv2. The precision for entity linking was about 73% and the recall reached about 50%. We also evaluated it with Troy200²⁶, a dataset of 200 statistical tables. In this case, the accuracy for entity linking was 64%. Note that a statistical table is cross-tabulation containing at least 2 variables (2 subjects in terms of [30]). To the best of our knowledge, the existing proposals are limited to one-subject tables (i.e. relations in

²³<https://www.w3.org/TR/rdf-sparql-query>

²⁴<https://www.w3.org/TR/rdf-concepts>

²⁵<http://webdatacommons.org/webtables/goldstandardV2.html>

²⁶http://tc11.cvc.uab.es/datasets/Troy_200_1

3NF) [31]. This work [32] pioneered an attempt to semantically interpret such tables.

The use of the proposed solution in domain-specific ontology engineering was demonstrated by an example from the industrial safety inspection (ISI) [33, 34, 35]. We developed an extension for PKBD²⁷ [36], the knowledge base management system. It allows for construction of an ontology in the OWL²⁸ format from RDF-triples, fragments of interrelated entities extracted from tables. A dataset²⁹ of 161 tables extracted from ISI reports was used as a source. As a result, a target ISI ontology including 25 KG-entities, 196 KG-properties, and 21 KG-relationships in the terminological level was automatically generated.

6. Conclusions

TABBYDOC project contributed in the theoretical basis and software tools for table extraction, cleansing, analysis, and interpretation. The results rely on contemporary techniques of the deep-learning, rule-based and generative programming, linked open data, and table understanding. The obtained results discovered new opportunities for intellectualization of the software engineering in data extraction from unstructured and semi-structured sources. Particularly, we proposed to develop methods and tools for the synthesis of tabular data transformation software based on table analysis and interpretation rules. We expect that it can expand the theoretical knowledge in the integration of heterogeneous tabular data. The developed software can be used in data science and business intelligence.

Further work implies automatic understanding of web-tables tagged with HTML markup. To extract data from spreadsheet tables we used the end-user programming as the main approach for development of the user-defined rules. This allowed us to support specific tricks of table layout, formatting, and content. However, to scale such solutions is too challenging when there are ambiguous tricks applied within source tables, whereas it is important that a solution intended for the Web should be easily scaled. This is possible for predefined types of web-tables. The latter is needed to classify them and select type-specific algorithms of analysis and interpretation. Thus, the approach we used heretofore is suitable for spreadsheet sources, but not for the Web. Further work will aim to fill this gap by developing a scalable solution for web-tables.

Acknowledgments

This work was supported by the Russian Science Foundation (Grant No. 18-71-10001).

References

- [1] J. L. Martinez-Rodriguez, A. Hogan, I. Lopez-Arevalo, Information extraction meets the semantic web: a survey, *Semantic Web* 11 (2020) 255–335. doi:10.3233/SW-180333.

²⁷<http://knowledge-core.ru>

²⁸<https://www.w3.org/OWL>

²⁹<https://data.mendeley.com/datasets/8zdymg4y96/1>

- [2] A. Hogan, E. Blomqvist, M. Cochez, C. d'Amato, G. de Melo, C. Gutierrez, J. E. L. Gayo, S. Kirrane, S. Neumaier, A. Polleres, R. Navigli, A.-C. N. Ngomo, S. M. Rashid, A. Rula, L. Schmelzeisen, J. Sequeda, S. Staab, A. Zimmermann, Knowledge graphs, 2021. [arXiv:2003.02320](https://arxiv.org/abs/2003.02320).
- [3] N. Milosevic, C. Gregson, R. Hernandez, G. Nenadic, A framework for information extraction from tables in biomedical literature, *Int. J. on Document Analysis and Recognition* 22 (2019) 55–78. doi:10.1007/s10032-019-00317-0.
- [4] J. C. Roldán, P. Jiménez, P. Szekely, R. Corchuelo, Tomate: A heuristic-based approach to extract data from html tables, *Information Sciences* (2021). doi:10.1016/j.ins.2021.04.087.
- [5] S. Schreiber, S. Agne, I. Wolf, A. Dengel, S. Ahmed, DeepDeSRT: deep learning for detection and structure recognition of tables in document images, in: *Proc. 14th IAPR Int. Conf. on Document Analysis and Recognition*, volume 1, 2017, pp. 1162–1167. doi:10.1109/ICDAR.2017.192.
- [6] Z.-Q. Zhao, P. Zheng, S.-T. Xu, X. Wu, Object detection with deep learning: a review, *IEEE Transactions on Neural Networks and Learning Systems* 30 (2019) 3212–3232. doi:10.1109/TNNLS.2018.2876865.
- [7] S. Ren, K. He, R. Girshick, J. Sun, Faster r-cnn: Towards real-time object detection with region proposal networks, in: *Proc. 28th Int. Conf. Neural Information Processing Systems - Volume 1*, MIT Press, Cambridge, MA, USA, 2015, pp. 91–99.
- [8] I. Cherepanov, A. Mikhailov, A. Shigarov, V. Paramonov, On automated workflow for fine-tuning deep neural network models for table detection in document images, in: *2020 43rd International Convention on Information, Communication and Electronic Technology (MIPRO)*, 2020, pp. 1130–1133. doi:10.23919/MIPRO48935.2020.9245241.
- [9] L. Gao, X. Yi, Z. Jiang, L. Hao, Z. Tang, Icdar2017 competition on page object detection, in: *2017 14th IAPR Int. Conf. Document Analysis and Recognition (ICDAR)*, volume 01, 2017, pp. 1417–1422. doi:10.1109/ICDAR.2017.231.
- [10] Z. Chi, H. Huang, H.-D. Xu, H. Yu, W. Yin, X.-L. Mao, Complicated table structure recognition, 2019. [arXiv:1908.04729](https://arxiv.org/abs/1908.04729).
- [11] L. Gao, Y. Huang, H. Déjean, J. Meunier, Q. Yan, Y. Fang, F. Kleber, E. Lang, ICDAR 2019 competition on table detection and recognition (cTDaR), in: *Proc. 15th Int. Conf. Document Analysis and Recognition*, 2019, pp. 1510–1515. doi:10.1109/ICDAR.2019.00243.
- [12] A. Shigarov, A. Mikhailov, A. Altaev, Configurable table structure recognition in untagged PDF documents, in: *Proc. ACM S. on Document Engineering*, 2016, pp. 119–122. doi:10.1145/2960811.2967152.
- [13] A. Shigarov, A. Altaev, A. Mikhailov, V. Paramonov, E. Cherkashin, TabbyPDF: web-based system for PDF table extraction, in: *Information and Software Technologies*, volume 920 CCIS, 2018, pp. 257–269. doi:10.1007/978-3-319-99972-2_20.
- [14] M. Göbel, T. Hassan, E. Oro, G. Orsi, A methodology for evaluating algorithms for table understanding in PDF documents, in: *Proc. ACM S. on Document Engineering*, 2012, pp. 45–48. doi:10.1145/2361354.2361365.
- [15] M. Gobel, T. Hassan, E. Oro, G. Orsi, ICDAR 2013 table competition, in: *Proc. 12th Int. Conf. on Document Analysis and Recognition*, 2013, pp. 1449–1453. doi:10.1109/ICDAR.2013.292.

- [16] T. G. Kieninger, Table structure recognition based on robust block segmentation, in: Document Recognition V, 1998, pp. 22–32. doi:10.1117/12.304642.
- [17] T. Kieninger, A. Dengel, The t-recs table recognition and analysis system, in: Document Analysis Systems: Theory and Practice, volume 1655 LNCS, 1999, pp. 255–270. doi:10.1007/3-540-48172-9_21.
- [18] A. Mikhailov, A. Shigarov, E. Rozhkov, I. Cherepanov, On graph-based verification for pdf table detection, in: 2020 Ivannikov ISPRAS Open Conference (ISPRAS), 2020, pp. 91–95. doi:10.1109/ISPRAS51486.2020.00020.
- [19] V. Paramonov, A. Shigarov, V. Vetrova, A. Mikhailov, Heuristic algorithm for recovering a physical structure of spreadsheet header, in: Information Systems Architecture and Technology: Proc. 40th Anniversary Int. Conf. on Information Systems Architecture and Technology – ISAT 2019, volume 1050 AISC, 2020, pp. 140–149. doi:10.1007/978-3-030-30440-9_14.
- [20] V. Paramonov, A. Shigarov, V. Vetrova, Table header correction algorithm based on heuristics for improving spreadsheet data extraction, in: Information and Software Technologies, volume 1283 CCIS, 2020, pp. 147–158. doi:10.1007/978-3-030-59506-7_13.
- [21] A. Shigarov, V. Khristyuk, A. Mikhailov, TabbyXL: software platform for rule-based spreadsheet data extraction and transformation, SoftwareX 10 (2019) 100270. doi:10.1016/j.softx.2019.100270.
- [22] K. W. Broman, K. H. Woo, Data organization in spreadsheets, The American Statistician 72 (2018) 2–10. doi:10.1080/00031305.2017.1375989.
- [23] A. Shigarov, V. Paramonov, V. Khristyuk, Spreadsheet data extraction from real-world tables of saus (the 2010 statistical abstract of the united states): case study, 2021. doi:10.6084/m9.figshare.14371055.v2.
- [24] A. Shigarov, V. Khristyuk, A. Mikhailov, V. Paramonov, Tabbyxl: Rule-based spreadsheet data extraction and transformation, in: Information and Software Technologies, volume 1078 CCIS, 2019, pp. 59–75. doi:10.1007/978-3-030-30275-7_6.
- [25] A. Shigarov, V. Khristyuk, A. Mikhailov, V. Paramonov, Software development for rule-based spreadsheet data extraction and transformation, in: Proc. 42nd Int. Conv. on Information and Communication Technology, Electronics and Microelectronics, 2019, pp. 1132–1137. doi:10.23919/mipro.2019.8756829.
- [26] A. Shigarov, Rule-based table analysis and interpretation, in: Information and Software Technologies, volume 538 CCIS, 2015, pp. 175–186. doi:10.1007/978-3-319-24770-0_16.
- [27] A. Shigarov, V. Paramonov, P. Belykh, A. Bondarev, Rule-based canonicalization of arbitrary tables in spreadsheets, in: Information and Software Technologies, volume 639 CCIS, 2016, pp. 78–91. doi:10.1007/978-3-319-46254-7_7.
- [28] A. Shigarov, A. Mikhailov, Rule-based spreadsheet data transformation from arbitrary to relational tables, Inform. Syst. 71 (2017) 123–136. doi:10.1016/j.is.2017.08.004.
- [29] N. Dorodnykh, A. Yurin, Tabbyld: a tool for semantic interpretation of spreadsheets data, in: Modelling and Development of Intelligent Systems, volume 1341 CCIS, 2021, pp. 315–333. doi:10.1007/978-3-030-68527-0_20.
- [30] K. Braunschweig, M. Thiele, W. Lehner, From web tables to concepts: a semantic normalization approach, in: Conceptual Modeling, volume 9381 LNCS, 2015, pp. 247–260.

doi:10.1007/978-3-319-25264-3_18.

- [31] S. Zhang, K. Balog, Web table extraction, retrieval, and augmentation: a survey, *ACM Trans. Intell. Syst. Technol.* 11 (2020). doi:10.1145/3372117.
- [32] N. Dorodnykh, A. Yurin, Towards a universal approach for semantic interpretation of spreadsheets data, in: *Proc. 24th S. on International Database Engineering and Applications*, 2020. doi:10.1145/3410566.3410609.
- [33] N. Dorodnykh, A. Yurin, Towards ontology engineering based on transformation of conceptual models and spreadsheet data: a case study, in: *Intelligent Systems Applications in Software Engineering*, volume 1046 AISC, 2019, pp. 233–247. doi:10.1007/978-3-030-30329-7_22.
- [34] A. Y. Yurin, N. O. Dorodnykh, Experimental evaluation of a spreadsheets transformation in the context of domain model engineering, in: *2020 Ural Symposium on Biomedical Engineering, Radioelectronics and Information Technology (USBREIT)*, 2020, pp. 0388–0391. doi:10.1109/USBREIT48449.2020.9117674.
- [35] N. O. Dorodnykh, A. Yurin, A. Shigarov, Conceptual model engineering for industrial safety inspection based on spreadsheet data analysis, in: *Modelling and Development of Intelligent Systems*, volume 1126 CCIS, 2020, pp. 51–65. doi:10.1007/978-3-030-39237-6_4.
- [36] A. Yurin, N. Dorodnykh, Personal knowledge base designer: software for expert systems prototyping, *SoftwareX* 11 (2020) 100411. doi:10.1016/j.softx.2020.100411.