Heuristic Algorithm for Automatic Extraction Relational Data from Spreadsheet Hierarchical Tables

Arwa Awad[1]
Faculty of Computer and Information Sciences
Ain Shams University, Cairo, Egypt

Ibrahim Moawad[3]
Faculty of Computer Science and Engineering
Galala University, New Galala City, Suez, Egypt

Rania Elgohary[2]
Faculty of Information Technology
Misr University for Science and Technology, Cairo, Egypt

Mohamed Roushdy[4]
Faculty of Computer and Information Technology
Future University in Egypt, Cairo, Egypt, Cairo, Egypt

***Abstract*—Spreadsheets are contained critical information on various topics and were most broadly utilized in numerous spaces. There are a huge amount of spreadsheet clients everywhere in the world. Spreadsheets provide considerable flexibility for data structure organization. As well as it gives their makers an enormous level of opportunity to encode their data as it is simple to utilize and easy to store the data in a table format. Because of this flexibility, tables with very complex and hierarchical data structures could be generated. Thusly, such complexity makes table processing and reusing this data is a difficult task. Therefore, the expansion in volume and complexity of these tables has prompted the necessity to preserve this data and reuse it. As a result, this paper implemented a novel algorithm-based heuristic technique and cell classification strategy to automate relational data extraction from spreadsheet hierarchical tables and without need any programming language experience. Finally, the paper does experiments on 2 different real public datasets. The percentage of average accuracy using the proposed approach on the two datasets is 95 % and 94.2% respectively.**

Keywords*—Spreadsheet table analysis; hierarchal table structure; cell classification; heuristic algorithm; relational data extraction

I. Introduction

A spreadsheet is an interactive application tool for organization charts, storage, and analysis of data. It contains data that is viewed in a tabular form. It consists of information orchestrated during a two-dimensional (2-D) cell. Normally, fragments during a table address the table name, and each line addresses the attributes or the values. The primary line or lines may contain segment attributes or variable names. Cells may contain numbers, text, dates, and other data types. Past this fundamental model, Spreadsheets are utilized by an enormous number of clients as a typical generally useful data management tool. Because of this flexibility, tables with very complex data structures could be generated. Thusly, such complexity makes automatic table processing and data extraction a difficult task. Therefore, the table pre-processing step is regularly needed in the data extraction pipeline [1].

Unfortunately, an enormous amount of spreadsheet tables are unstructured tabular data with simple or hierarchical and sophisticated table structures. The attributes or the header of simple tables [2] acts in only the first line or only in the left column (rule). The hierarchical table structures [3] act in hierarchical attributes at the top lines (highest rows) of spreadsheet tables or multi-dimensional levels of left attributes (left columns). As shown in "Fig. 1", the complex table structure [4] acts within the top and left attributes together. These attributes of complex table structures could also be hierarchical or simple (top and left) without hierarchy. They are intended to be explicitly understood by humans but not by a machine which makes it implicit to understand and process these hierarchal and unstructured data. Nowadays, there is an explosion of computation intelligence vision that requires the data to be understood by machine and extracted in a structured form like an electronic database. The automatic extraction of hierarchical table header structure is the initial step of converting unstructured data to structure data form.

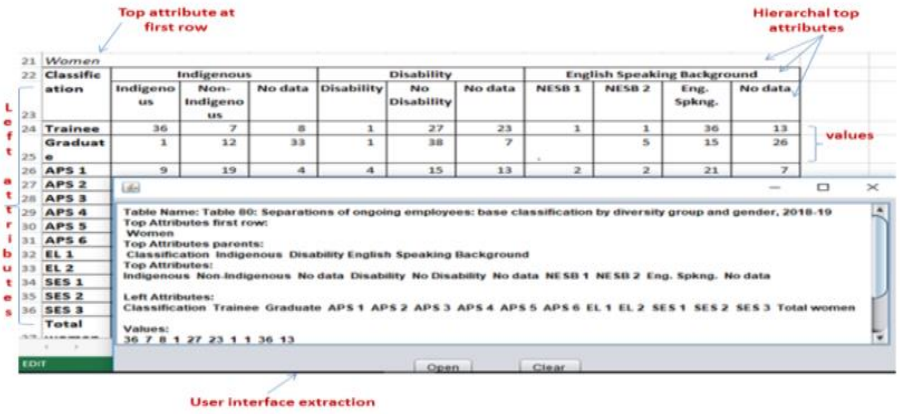
The data stored in spreadsheets is unstructured, inconsistent, has low quality, and lacks data integration operations. Spreadsheets do not observe a standard data model (definite structure or schema), thus it is difficult to integrate with other data sources. The spreadsheet tables contain a huge amount of high value that needs to convert to a relational form to reuse and integrate. The first logical step to make such data relational is to extract the relational table structure from a spreadsheet.

According the literature, there are three main types of approaches that are responsible for extracting relational data from the spreadsheets:

1) Schema-based approach: It is a traditional schema mapping system used to extract or convert from spreadsheets to relational data by specifying the source and target attributes mapping.

2) Rule-based approach: It requires explicit user-provided conversion rules.

3) Visualization-based approach: It provides the users with an interactive visualization interface to extract or manage the underlying data that exist in the spreadsheets.



Women	Indigenous			Disability			English Speaking Background			
Classification	Indigenous	Non-Indigenous	No data	Disability	No Disability	No data	NESB 1	NESB 2	Eng. Spkng.	No data
Trainee	36	7	8	1	27	23	1	1	36	13
Graduate	1	12	33	1	38	7		5	15	26
APS 1	9	19	4	4	15	13	2	2	21	7
APS 2										
APS 3										
APS 4										
APS 5										
APS 6										
EL 1										
EL 2										
SES 1										
SES 2										
SES 3										
Total										

Fig. 1. A Running Example of Complex Table Structure with user Interface Extraction.

However, there are three common draw-backs of these three main approaches:

a) Challenges to handle hierarchal spreadsheet tables.

b) The extraction process cannot be accomplished automatically.

c) Most of the approaches require the users to learn a new language or pre-defined operators to describe the extraction rules.

Therefore, the paper used an automatic approach based-heuristic algorithms and cell classification strategy to accurately extract relational data from hierarchal and complex spreadsheet tables.

The importance of this research lies in the future vision of the relational data extracted from a spreadsheet includes computer agents searching for information, making decisions, and taking actions on behalf of human users. Thus, it is needed to automate extracts for collected complex table header structure in the spreadsheet (semi-structured format) to improve spreadsheet quality as it is the precondition for investigating and utilizing information and for ensuring the estimation of the information.

The main contribution of this research is developing variant techniques and algorithms that allow discovering tables in spreadsheets to infer their layout and their relational data. However, most of the existing approaches and techniques cannot achieve the extraction process automatically for hierarchal table structure extraction. It often required some user efforts to accomplish this process. As a result, this paper developed an automatic approach that is accompanied by some heuristic rules and cell classification features.

As extension of our previous work [5, 6], a new method was developed to extract multiple tables from the Excel sheets. Also, extended to extracts implicit and relational data from complex and hierarchy structure tables not only from simple table structure based on:

Proposed an algorithm based on heuristic rules and classification cell features for selecting complex and hierarchal section header lines and data values. This methodology provides a way to extract a more and more organized structure data from spreadsheets.

This paper is organized as follows. Section 2 discusses the related work as well as the relational data extraction. Whereas Section 3 introduces the proposed approaches with the process extraction of relational data from spreadsheet tables, Section 4 presents the proposed algorithms. Section 5 presents the results and analysis of the experiment for the accuracy of complex table structure. Finally, the paper is concluded in Section 6.

II. Related Work

Spreadsheet table discovery is the assignment of identifying all tables on a given sheet and finding their reaches. Table detection header is the stage of detection mechanism that achieves state-of-the-art results in computer vision [7, 8] based on the convolutional neural networks approach. On the other hand, [9] presented a multi-task learning method to extract a semantic structure of a table. [10] Includes heuristics on the structure and textual content of a table that is designed for three model table types. [11] Combines various heuristic-based algorithms that classify spreadsheet cells into four functional groups (roles; headings or data values). [12] Used rules-based language for table analysis and interpretation. [13] Assumes that all content in one cell is either a label or an entry. [14] Proposed a classification approach and training the data to discover only the layout of tables in spreadsheets above cell level structure. [15] Presented a predictive synthesis algorithm to helpfully automating the data wrangling process. [16] Used the heuristic approach for table header to correct and achieve matching between the physical and visual presentation of the table structure. However, authors in [17] used programming by examples for hierarchal data conversion to a relational table. The authors of [18] applied the cell features algorithm to classify and identify the table spatial in the spreadsheet. The authors of [19] generate a flash relate algorithm and the user must use the input-output examples for the relational data required. In addition, the authors in [20] used a heuristic method to correct a physical structure to the table header by merging the empty cell with the neighboring ones that are not empty. Most of these algorithms require user efforts for relational data extraction or conversion.

Through reviewing previous work, the research has covered this point from different angles, including automation like the proposed work [21], which does not need any user intervention. However, they automatically infer some

spreadsheet structure, but they cannot process hierarchical spreadsheets. Semi-automation [22,23] who needs user intervention, and sometimes needs to understand the user to a specific programming language, and what needs to be programming language [24,25]. The current study is consistent with past investigations in its principle subject and the general objective, yet it differs from it in the application strategy.

III. Research Method

As depicted in "Fig. 2", the initial phase for the extraction process of research approach is to detect the tables found in the Excel sheet. The spreadsheet may contain only one table or multiple tables. After that, the algorithm starts to detect the tables' names if found. Then the algorithm begins to distinguish between tables' headers and their values. Finally, the proposed algorithm extracts tables found in the spreadsheet with their names and their attributes besides their values. The detailed process of each phase can be described in the next sub-section as the following steps:

A. Table Detection and Extraction Phases

Spreadsheet table detection and discovery is the first step needed to complete the process of changing over information put away in spreadsheets into organized information in a database that can be questioned and handled to applied decision making and knowledge discovery rules. Table detection is the errand of recognizing all tables on a given sheet and finding their separate reaches as shown in "Fig. 3".

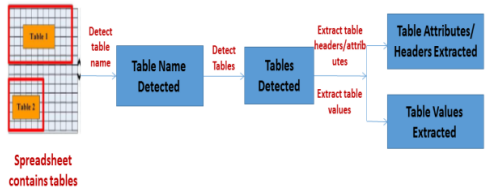


Fig. 2. The Research Methodology.

```
1- Iterate upon all sheets
      for (int i = 0; i < workbook.getNumberOfSheets(); i++) {XSSFSheet
sheet = workbook.getSheetAt(i);
2- Detect rows [max-written rows number].
sheet.getPhysicalNumberOfRows()
3-Read number of merge region and detect it's address.
CellRangeAddress adress = new CellRangeAddress(first column ,last column
,first row , last row);
 adress =  sheet.getMergedRegion(region number);
4-Iterate on all rows
Iterator<Row> rowIterator = sheet.iterator();
            while (rowIterator.hasNext())
            Row row = rowIterator.next();
5-Iterate on every cell in all rows.
Iterator<Cell> cell iterator = row.cellIterator();
while (cellIterator.hasNext())
Cell cell = cellIterator.next();
6-Read cell value and detect which it value or attribute.
7-Get last row number at every sheet int lastRow =
sheet.getLastRowNum();
```

Fig. 3. Table Detection Process.

After detecting the table in the spreadsheet, the algorithm starts to extract table name, attributes, and values based on the heuristic rules and cell features as shown in "Fig. 4", "Fig. 5", and "Fig. 6".

1. If this cell Attribute cell
2. Only one cell at this row
3. Next row is empty blankRow == null
4. No table name yet TablNm == false
5. Cell value contains substring "Table" CelValue.contains("Table")

Fig. 4. Extract Table Name.

1. If this cell Attribute cell
2. Extract table name before TablNm == true
3. There are other attribute cells at the same row
4. Check if this merged region and get its address (parent attribute)
5. sheet.getNumMergedRegions() \\ calculate NUMBER of merged region
6. CellRangeAddress address = new CellRangeAddress(firstRow,endow,firstColumn,endColumn) \\ return address of merged region
7. There is an empty cell in the same row
8. If the next row has an attribute cell, this is a hierarchal attribute
9. Every cell in the next row has the same address of merged cell be a child of parent Attribute

Fig. 5. Extract Top Attributes.

1. If this cell Attribute cell
2. First cell at row columnIndex == 0
3. There are not any attribute cells at the same row phNumOfCell == 1
4. If the rest of the row blank is parent left Attribute
5. If there are other data at the same row value left attribute

Fig. 6. Extract Left Attributes.

B. Extract more than One Table from One Datasheet

In the case of the existence of more than one table in the same datasheet, there are two possible causes for the positioning of them. Case (A), if these tables were placed side by side separated by one or more empty columns. Case (B), if these tables were placed over each other separated by one or more empty rows.

IV. Horizontal Synchronization

In this case, if the algorithm found an empty column before the maximum number of the filled data columns, that means there is another table in the same rows. The algorithm will implement a new matrix to store the new table.

V. Vertical Synchronization

In this case, if the algorithm found an empty row before the maximum number of the filled data rows, that mean there is another table below. The algorithm will finalize and send the current table then starts to explore the new one.

VI. Algorithms Used

A. Header Inference Algorithm based on Heuristic Rules and Cell Classification Features

1) The spreadsheet table has its exceptional trait of being a structure of vertically on a level plane extended segments, which are table name, attributes, and values. The paper displays a calculation for table identification in spreadsheets. The calculation utilizes three kinds of cells as its premise: table name cell, attribute cell, and value cell. As shown in "Fig. 7", the cell classification identification proof was based on its heuristic features and the number of cells, one cell or more than one cell.

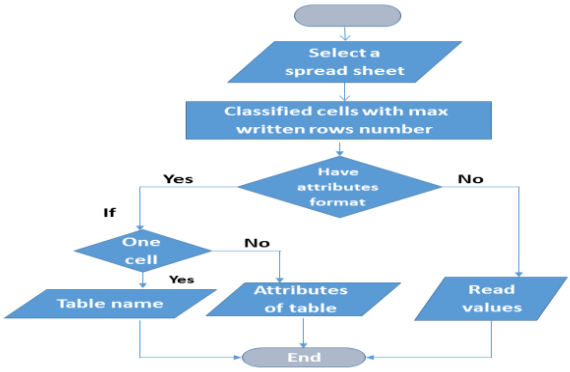


Fig. 7. Cell Classification Flow Chart.

Once the user opened the selected spreadsheet, the algorithm begins to detect all tables found in the spreadsheet using the method of cell classification based on detect max written rows number. This method detects the first row that contains data and the last row that contains data which makes a table frame. After that, the algorithm will checks the cell formats for example if it is bold, italic, justified, capitalized, or colored to detect if this cell has an attribute's format or not. The attribute's format detects based on the cell classification numbers as well as one cell and has attributes format, it will be table name. However, if there is more than one cell and contains the format of the attributes, it will be attributes or headers of the detected table. However, table attributes detection is divided into two phases, top and left attributes. Top attributes are the first row that is not empty and contains data with attributes format and the left attributes are in the first left column within attributes format. The Top and Left attributes may be extended in a multi-level of rows or columns respectively which makes it a hierarchal attribute. Otherwise, the algorithm detects and extracts table values based on a cell type if it is the numeric type which means the cell contains numbers or the string cell is not an attribute cell.

Heuristic rules for complex and hierarchal table structures are found in the top first row and first left column. Also, Heuristic rules for hierarchal table attributes extended for multi-level rows or columns. Excel relational data extraction for table structure can be summarized as shown in Algorithm 1.

Physical layout features (the cell has right border, the cell has bottom border, the row has bottom border, if there is a table, the table has outside border, the table has a bottom double border, the table has thick bottom border, the table has top and double bottom border, the row has a top and double border, the cell has all borders, the column has a right border if there are merged cells).

Algorithm 1. Excel relational data extraction

Input: Exc
el workbook with multiple sheets A = {S_1,S_2,…,S_n} and contains complex and hierarchal tables structure
Output: Separate tables each include TBL: Table name as a variable TblNm and Attributes as an array list Att ={att_1, att_2,…..,att_n}, values as Val[$_{m,n}$] = {val_{m1n1},val_{m1n2},…….$val_{m,n}$}

```
Begin A
  For every sheet in A Do
    Begin new TBL                 // row 3
      For every filled row Do
      For every filled column Do
          For every cell Do
              If the cell has attributes format and no other data in
this row or column    // one cell
              TblNm<- cell data
              Else if a cell has attributes format and there is other
data in this row or column      // more than one cell contains the same
format  // top or left attribute
              Att <- cell data
              Else if the cell in the next line have Attributes format
// next line column or row
              Att <- cell data    // Hierarchal attributes
              If a cell does not have attributes format or it is a numeric
value
              Val[m,n] <- cell data
              Else if an empty row or an empty column go to row 3
              End if
              End if
          End for
      End for
    End for
End for
```

Semantic content features for cells (bold, italic, underline, different font or content style, different background cells color (fill color or cell format), alignment (top, middle, bottom, left, center, right), attribute contains keyword such as: "Total"," Name", "ID", "Average, "Avg", "Date", Year…etc., attribute letters are all capitalized, attribute contains a colon, attributes font size is bigger than values font size, the attribute has blank cells in the middle(blank row or column inside the detected frame), child's row index is greater than parents, parent row index is smaller than children, in left attributes the child's indentation is greater than parents, attribute cell is long width, if there are the same features on multiple rows or column until finding deferent features, etc.

VII. Results and Discussion

The evaluations of the accuracy of the approach are used and illustrate it by utilizing the gov.au.data dataset [26]. The paper downloaded two different public datasets with Excel (.xlsx) Formats. The datasets contain multiple sheets (61 and 82 sheets, respectively) and every sheet may contain only one table or multiple tables as illustrated in Table I. The evaluation supported relational data detected and extraction from the spreadsheet tables.

TABLE I. EXPERIMENTAL RESULTS

Datasets	Number of sheets	Number of randomly selected sheets, tables	Number of successfully identified relational data	Percentage of successfully extracted relational data %
1- Australian Public Service Statistical Bulletin - December 31, 2016	61 sheets	20 sheets with 20 tables	Table name: 20 Top attributes: 19 Left attributes: 18	Table name: 100% Top attributes: 95% Left attributes: 90%
2- APS Employment Data 30 June 2019 release	82 sheet	20 sheets with 29 tables	Table name: 29 Top attributes: 27 Left attributes: 26	Table name: 100% Top attributes: 93.1% Left attributes: 89.6%

The experimental results showed that the percentage of accuracy to the first dataset of table name detection and extraction is 100%, top attributes are 95%, and left attributes are 90%. However, the second dataset results accuracy is 100% for the table name detected and extraction, 93.1% of top attributes, and 89.6% of left attributes. In summary, It is important to need to understand the nature of the Excel sheet that will be working on because each excel sheet has a different structure from the other one, and therefore the difficulty of extracting data from it comes from here. Therefore, it needs to know the structure of the Excel sheet that will be work on, and if there is any structure other than this, the extraction algorithm will be stopped working, or it will not work correctly to extract the data from it. Thus, to extract a large corrected number of relational tables successfully from spreadsheets, you have to identify a variety and of heuristic tables' cells features and rules.

The algorithms are written in JAVA as a programing language, connected with SQL server as an analysis tool, and Windows 7 as an operating system. To test it, an executable jar file is created. The hardware specifications for the system are, Intel® Core™ 2 Duo CPU with 4 GB of RAM. The version of Microsoft Excel installed on the machine is 2013. However, the proposed tool works only with new version of Excel ('.xlsx' format).

VIII. CONCLUSION AND FUTURE WORK

This paper presents a detailed algorithmic data processing method for automatically extracting relational data in complex and hierarchal table structures from spreadsheets. The automatic approach is accompanied by some heuristic rules, features, and cell classification categories. The experimental results showed that the accuracy of relational data detected and extracted from the spreadsheet tables for the tested two datasets are 95% and 94.2% respectively. In the future work, the proposed method can be extended to integrate the extracted data into an existing relational database.

IX. CONFLICTS OF INTEREST

The paper declares that it has no conflict of interest and no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

ACKNOWLEDGMENT

We would like to thank the "Future University in Egypt" for its continuous support and efforts to improve the scientific research results for publication.

REFERENCES

[1] N. Mohamad, N. Ahmad, and S. Sulaiman. "Data pre-processing: a case study in predicting student's retention in MOOC." Journal of Fundamental and Applied Sciences, 9.4S, (2017), pp.598-613.

[2] L. Irina, and A. Begler. "A method of semi-automated ontology population from multiple semi-structured data sources." Journal of Information Science, (2020): 0165551520950243 , pp.1-14.

[3] K. Elvis, et al. "A machine learning approach for layout inference in spreadsheets." IC3K 2016: Proceedings of the 8th International Joint Conference on Knowledge Discovery, Knowledge Engineering and Knowledge Management: volume 1: KDIR. SciTePress, 2016, pp.77-88.

[4] C. Zhe, and M. Cafarella. "Automatic web spreadsheet data extraction." Proceedings of the 3rd International Workshop on Semantic Search over the Web. 2013, p.1-8.

[5] Awad, A., Roushdy, M. I., ElGohary, R. A. E., & Moawad, I. F. "Metadata extraction for low-quality semi-structured spreadsheets." Joint European-US Workshop on Applications of Invariance in Computer Vision. Springer, Cham, 2020, p. 448-457.

[6] Awad, A., Roushdy, M. I., ElGohary, R. A. E., & Moawad, I. F. An interactive tool for extracting low-quality spreadsheet tables and converting into relational database. International Journal of Intelligent Computing and Information Sciences, 21(1), (2021), 1-18.

[7] G. Ross, et al. "Rich feature hierarchies for accurate object detection and semantic segmentation." Proceedings of the IEEE conference on computer vision and pattern recognition. 2014, p. 580-587.

[8] D. Haoyu, et al. "Tablesense: Spreadsheet table detection with convolutional neural networks." Proceedings of the AAAI Conference on Artificial Intelligence. Vol. 33. No. 01. 2019, pp. 69-76.

[9] D. Haoyu, et al. "Semantic structure extraction for spreadsheet tables with a multi-task learning architecture." Workshop on Document Intelligence at NeurIPS 2019. 2019.

[10] P. Aleksander, et al. "Transforming arbitrary tables into logical form with TARTAR." Data & Knowledge Engineering 60.3 (2007): 567-595.

[11] RA. Robin, and M. Erwig. "UCheck: A spreadsheet type checker for end users." Journal of Visual Languages & Computing 18.1 (2007): 71-95.

[12] S. Alexey, and A. Mikhailov. "Rule-based spreadsheet data transformation from arbitrary to relational tables." Information Systems 71 (2017): 123-136.

[13] TT. Cui, and D. Embley. "Automatic hidden-web table interpretation, conceptualization, and semantic annotation." Data & Knowledge Engineering 68.7 (2009): 683-703.

[14] KK. Elvis, et al. "A machine learning approach for layout inference in spreadsheets." IC3K 2016: Proceedings of the 8th International Joint Conference on Knowledge Discovery, Knowledge Engineering and Knowledge Management: volume 1: KDIR. SciTePress, 2016, p. 77-88.

[15] VV. Gust, and L. Raedt, "Towards automated relational data wrangling." Proceedings of AutoML 2017@ ECML-PKDD: Automatic selection, configuration and composition of machine learning algorithms 1998 (2017): 18-26.

[16] PP. Viacheslav, A. Shigarov, and V. Vetrova. "Table header correction algorithm based on heuristics for improving spreadsheet data extraction." International Conference on Information and Software Technologies. Springer, Cham, 2020, p. 147-158.

[17] YY. Navid, X. Wang, and I. Dillig. "Automated migration of hierarchical data to relational tables using programming-by-example." Proceedings of the VLDB Endowment 11.5 (2018): 580-593.

[18] KK. Elvis, et al. "Table identification and reconstruction in spreadsheets." International Conference on Advanced Information Systems Engineering. Springer, Cham, 2017, p. 527-541.

[19] BB. Daniel, et al. "FlashRelate: Extracting relational data from semi-structured spreadsheets using examples." ACM SIGPLAN Notices 50.6 (2015): 218-228.

[20] PP. Viacheslav, et al. "Heuristic algorithm for recovering a physical structure of spreadsheet header." International Conference on Information Systems Architecture and Technology. Springer, Cham, 2019, p. 140-149.

[21] JJ. Cunha, J. Saraiva, and J. Visser. From spreadsheets to relational databases and back. In PEPM, 2009, p. 179-188.

[22] AA. Nikita, D. Turdakov, and N. Vassilieva, "Semi-automatic data extraction from tables." RCDL, 2013, p. 14-20.

[23] KK. Sean, et al. "Wrangler: Interactive visual specification of data transformation scripts." Proceedings of the SIGCHI Conference on Human Factors in Computing Systems. 2011, p. 3363-3372.

[24] GG. Sumit, W. Harris, and R. Singh, "Spreadsheet data manipulation using examples." Communications of the ACM 55.8 (2012): 97-105.

[25] LL. Vu, and S. Gulwani, "Flashextract: A framework for data extraction by examples." Proceedings of the 35th ACM SIGPLAN Conference on Programming Language Design and Implementation. 2014, p. 542-553.

[26] https://data.gov.au/data/dataset?organization=australianpublicservicecommission&res_format=excel+%28.xlsx%29&_res_format_limit=0.