Converting data organised for visual perception into machine-readable formats

A new intuitive tool for data preparation and structuring

Alexander Aue [1], Andrea Ackermann[1] and Norbert Röder [1]

Abstract: Spreadsheets are used to store an extraordinary amount of important data. The fact that spreadsheets are both easy to use and allow users a great deal of flexibility in how they store their data is a significant reason why they are so popular. Users often use a variety of layout techniques to make the data easy for humans to understand. But this layout also creates problems for traditional Extract-Transform-Load (ETL) tools. We propose a program that allows users to easily extract data from Excel files by selecting the cells containing the data and metadata thereby determining the data hierarchy. We have used this program to extract data of the Agricultural Structure Survey on land use and livestock in Germany, which does not follow a nationwide standard, leading to large differences in the structuring of the data between the federal states, making it a good benchmark.

Keywords: semi-structured data, ETL, no-code, Excel, spreadsheets, data harmonisation

1 Introduction

For regional or thematically comprehensive analyses, data must frequently be obtained from different data sources or providers, sometimes digitized, and structurally and contextually merged before analysis. Excel file formats are often used as the data format, as this software is widely available and has low entry barriers. One such use case in Germany is the data from the Agricultural Structure Survey on land use and livestock farming, regularly collected and provided by state agencies (e.g. [SB23]). These data are often not machine-readable because the design of the spreadsheets is optimized for complex structured data with multi-dimensional dependencies. This complexity is structured in a way that humans can understand the data when presented as two-dimensional tables. A variety of layout techniques are used to visually structure the spreadsheets. As the data structure does not follow any standard, leading to substantial differences in data structuring between federal states and sometimes over time, this

[1] Thünen-Institute of Rural Studies, Bundesallee 64, 38116 Braunschweig, alexander.aue@thuenen.de, https://orcid.org/0009-0001-8683-3630; andrea.ackermann@thuenen.de; norbert.roeder@thuenen.de, https://orcid.org/0000-0002-2491-2624

provision of data poses significant challenges for traditional Extract-Transform-Load (ETL) tools.

In Fig. 1, an example data subset of cattle farms at the state level in Germany is depicted. For the purposes of this demonstration, the original spreadsheet was modified to include only three rows per table, and the alignment of the state heading was adjusted to the left to simplify the visual complexity of the subsequent figures.

Row no.	Farms with ... to ... Calves and Young Cattle	Farms	Stock of Calves and Young Cattle		Stock of Cattle	Livestock Total	Agric. Utilised Area
		Quantity		LU			Hectares ha
		1	2	3	4	5	6
Germany							
1	1 - 9	35 237	161 349	48 405	608 150	759 225	1112 718
2	10 - 19	19 705	276 477	82 943	823 569	908 351	1025 698
3	20 - 49	26 140	829 799	248 940	2297 135	2453 837	2145 194
Berlin							
1	1 - 9	5	.	.	.	.	.
2	10 - 19	3	53	16	231	259	251
3	20 - 49	1	.	.	.	.	.

Fig. 1: Example data subset of the Agricultural Structure Survey on land use and livestock in Germany [SB23]

The table is designed for easy comparison of the count of farms and animals, as well as the associated units, such as quantity, livestock units, and land area in hectares. For statisticians for whom this overview is intended, the data organization would not present any difficulties in understanding the worksheet. However, typical data analysis software encounters challenges with this organization as it typically requires data to be in a relational form. R and Python scripts can bring the data into a structured format, but due to the many special cases and exceptions in the data structures, script development can be very complex.

One of the most widely used no-code ETL tools, Microsoft Power Query, is not suitable for these data. The visual formatting used in the spreadsheets such as layout elements include combinations of wide and long formats in the same table, nested table headers, as well as vertical thematic or regional subdivisions in the worksheet make them non-transformable by Power Query. Additionally, empty cells are used to centre headers vertically and/or horizontally, creating the visual impression of a merged column header, even though the cells remain technically separate.

The problem we aim to address with this paper is to enable ordinary spreadsheet users without programming knowledge to transform spreadsheets organized for visual perception into a relational form. For this purpose, we present a program that allows users to describe the hierarchical and relational structures within a spreadsheet. Using the

description of the data structures, the program generates a relational representation of the data from the spreadsheet as a CSV file.

2 Related Work

Our program's closest research counterparts are the Senbazuru project and the FlashRelate project, both of which share the goal of extracting relational data from spreadsheets [Ch13; Ba15].

Senbazuru aims to automatically infer the hierarchical structure within spreadsheets through the creation of a classifier that identifies data frames in the document, along with another classifier that infers the intended hierarchy using a predefined set of features [Ch13]. While Senbazuru offers automated inference of the hierarchical structure, it may occasionally produce errors. In such cases, the tool allows the user to rectify these errors in the hierarchy by manually dragging and dropping nodes to their correct parent locations. Unlike Senbazuru, our program does not automatically infer the hierarchical structure. Instead, our program lets the user decide if the next selection is a child or a sibling of another selection. To speed up the process, selections can be duplicated and moved both horizontally and vertically. Thus, a hierarchy representing a column can be duplicated and shifted over another column, and similarly, a hierarchy representing a table can be duplicated and shifted over another table.

FlashRelate utilizes both positive and negative output examples to generate a program in a domain-specific language called Flare [Ba15]. This approach enables FlashRelate to handle various extraction tasks from diverse spreadsheets, as long as it is possible to identify cells using regular expressions and recognize the spatial arrangement of the cells. In contrast to FlashRelate, our program necessitates the user to manually choose each data cell containing either data or metadata. It does not automatically identify similar cells based on user-provided selections. Instead, our program allows users to select not only individual cells but also sets of cells. Portions of the spreadsheet that represent relational data rather than hierarchical data can be chosen by first selecting the domain of the relation as a set of cells. Typically, the range values of the relation share the same row as the domain values. In a subsequent step, the ranges of the relation can be selected by duplicating the cell set of the domain and moving it horizontally to the desired cells containing the range values.

TableSense is another approach that employs Convolutional Neural Networks to identify tables in spreadsheets [Do19]. In their test with the WebSheet400 dataset, out of 400 random spreadsheets containing 795 tables, TableSense successfully detected 91.3% of the tables (recall), and of the tables detected by TableSense, 86.5% were indeed genuine tables (precision). However, while TableSense excels at detecting table boundaries, it does not identify data hierarchy or extract data from the tables it identifies. Our approach currently does not use artificial intelligence. Our project's primary focus is to empower

users to manually select all the desired data and describe its hierarchy, enabling our program to automatically extract this data into a relational format.

3 Methodology

We used the software development kit Flutter to develop a program with a user interface that allows the selection of the cells containing the metadata and data inside the spreadsheet. We refer again to the example from Fig. 1 to describe the necessary steps required for data extraction from spreadsheets. To minimize user interaction, it is advisable to begin by selecting the most abstract meta-information and then progressively narrow down to increasingly specific meta-information until the actual facts are chosen. In Fig. 2, cells are presented as they appear within our program's data selection view after all the selections are made. Additionally, an additional layer with black arrows illustrates the optimal sequence of selections for this example, designed to minimize user interaction. These arrows are numbered from 1 to 7, representing the specific steps of user interaction.

Please note that the program deconstructs merged cells. Cells from which the merged cell was created are displayed as individual cells (e.g. “Quantity” and “LU”). This will be useful for the following selections.

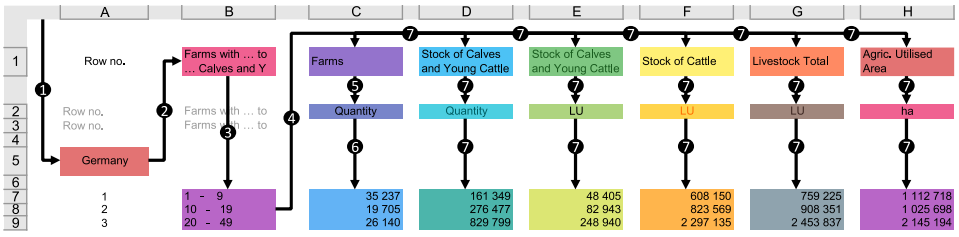

Fig. 2: The data selection view, once all selections are completed, along with an additional layer featuring black arrows indicating the sequence of selections with minimal user interaction

The most abstract meta-information, which is the meta-information shared by most cells, is the country or state in this example (Arrow 1). After that, the header of the domain of the relation is selected (Arrow 2) and then the domain itself will be picked (Arrow 3). While Selections 1 and 2 each selected individual cells, Selection 3 introduces the concept of choosing a set of cells.

Moving on to the first data column, the process involves selecting the first column header (Arrow 4), followed by choosing the second column header (Arrow 5). Now, the first range of the relation is selected (Arrow 6).

If the selection of data were to continue in the same manner, this would involve an additional 15 selections just for the first table, offering no significant improvement over manually creating the relation by simply copying and pasting the cells. However, the

subsequent selections can be automated, given that both the two column headers and the numeric values are located in the same row, each one cell to the right of the previous one. The program's 'Duplicate and Move' function allows these cells to be selected in a single step. To do this, Selection 4 will be chosen, and the duplicate is moved one cell to the right. A slider can be adjusted to determine how many duplicates are to be generated. Each duplicate is moved the same number of cells in the same direction as the previous one. For this example, the slider is set to create 5 duplicates (see Fig. 3). This results in all the selections depicted as arrows labelled 7 in Fig. 2. With this, the first table is captured.

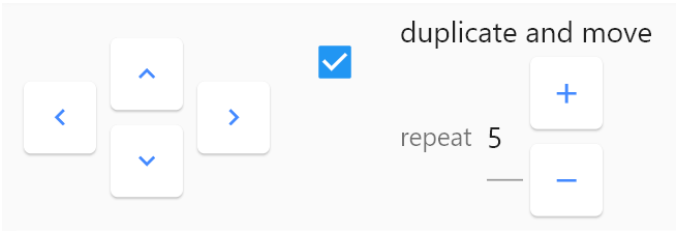

Fig. 3: The user interface to move selections and to duplicate and repeat the step multiple times

The same procedure used to capture the previous 5 columns can be applied to capture the tables arranged below one another. For that, Selection 1 is chosen, then duplicated and moved down by 6 cells. However, the header cells (B1 to H2) are an exception to this, because they do not reappear in the tables below. The program offers a 'freeze' option for this purpose. The 'frozen' cells are duplicated but remain in their original positions during the move. Fig. 4 illustrates how the 'Duplicate and Move' function was employed to capture the lower table by moving all but the 'frozen' selections down by 6 cells (all arrows labelled with the number 8).

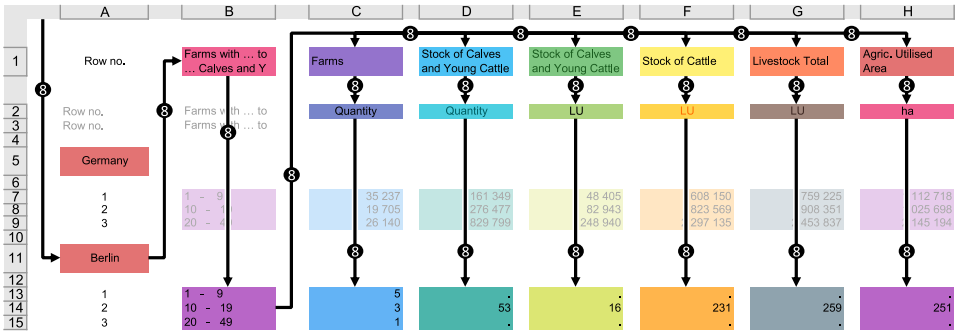

Fig. 4: Expansion of the example depicted in Fig. 2, showcasing the final step of selecting the tables arranged one below the other

After the selections are made, the data can be exported as a CSV file as depicted in Fig. 5.

Germany	Farms with ... to ... Calves and Young Cattle	1 - 9	Farms	Quantity	35237
Germany	Farms with ... to ... Calves and Young Cattle	10 - 19	Farms	Quantity	19705
Germany	Farms with ... to ... Calves and Young Cattle	20 - 49	Farms	Quantity	26140
Germany	Farms with ... to ... Calves and Young Cattle	1 - 9	Stock of Calves and Young Cattle	Quantity	161349
Germany	Farms with ... to ... Calves and Young Cattle	10 - 19	Stock of Calves and Young Cattle	Quantity	276477
Germany	Farms with ... to ... Calves and Young Cattle	20 - 49	Stock of Calves and Young Cattle	Quantity	829799

Fig. 5: The first rows of the exported CSV file

4 Results

Our student assistants used the program to extract data from more than 500 Excel files. The time taken for each file was determined from a sample of 331 processed Excel files, comprising 3,093 worksheets, with outliers caused by pauses removed. On average, the student assistants needed 15 minutes per file. The range falls between 4 minutes (25% quantile), 7 minutes (median), and 18 minutes (75% quantile) of processing time per file. This results in an average processing time per worksheet of 95 seconds.

5 Conclusion and Outlook

We have successfully developed a prototype No-Code ETL tool. Using data from the Agricultural Structure Survey on land use and livestock farming, we showed that our program can be used by users with no programming experience to extract data from Excel files with diverse formats. In the future, we aim to explore how AI can potentially reduce user interaction by providing suggestions for additional selections. Additionally, we plan to integrate data cleansing capabilities into the user interface.

Bibliography

[Ba15] Barowy, D. W. et al.: FlashRelate: extracting relational data from semi-structured spreadsheets using examples. ACM SIGPLAN Notices 6/50, pp. 218–228, 2015.

[Ch13] Chen, Z. et al.: Senbazuru. Proceedings of the VLDB Endowment 12/6, pp. 1202–1205, 2013.

[Do19] Dong, H. et al.: Tablesense: Spreadsheet table detection with convolutional neural networks: Proceedings of the AAAI conference on artificial intelligence, pp. 69–76, 2019.

[SB23] Statistisches Bundesamt: Viehhaltung der Betriebe - Fachserie 3 Reihe 2.1.3 - 2020 (Letzte Ausgabe - berichtsweise eingestellt). https://www.destatis.de/DE/Themen/Branchen-Unternehmen/Landwirtschaft-Forstwirtschaft-Fischerei/Tiere-Tierische-Erzeugung/Publikationen/Downloads-Tiere-und-tierische-Erzeugung/viehhaltung-2030213209005.xlsx, accessed 30 Oct 2023.