UTOPIA: Automatic Pivot Table Assistant

Whanhee Cho
University of Utah
Salt Lake City, Utah, USA
whanhee@cs.utah.edu

Anna Fariha
University of Utah
Salt Lake City, Utah, USA
afariha@cs.utah.edu

ABSTRACT

Data summarization is required to comprehend large datasets, and aggregations are effective ways to summarize data. A *pivot table* is a mechanism to aggregate numerical attributes grouped by categorical attributes and spreadsheet pivot tables are particularly suitable for novices, where they can rearrange, group, and aggregate data using intuitive interfaces. However, real-world spreadsheet data is often disorganized, with attributes having *multiple values* and *synonymous variants*. For instance, in the IMDb data, multiple genres are stored as a comma-separated value ("Action, Comedy, Drama") and "Science Fiction" can be represented in various ways: "Sci-Fi", "scifi", "Technological Fiction", etc. Such data issues pose barriers for novices while constructing pivot tables, and result in noisy and incomprehensible summarization. Parsing multi-valued attributes forces users to resort to external tools (Power Query) or methods (Python), requiring additional expertise; and consolidating synonymous variants demand manual effort, which is a tedious task.

We introduce UTOPIA, an a**UTO**matic **PI**vot table **A**ssistant that extends the functionality of spreadsheet pivot tables, overcoming data issues such as multi-valued attributes and synonymous variants. UTOPIA helps construct pivot tables without requiring additional expertise by (1) automatically detecting multi-valued attributes and organizing the values, achieving *implicit data normalization*, and (2) leveraging SimCSE embeddings and K-Means clustering to consolidate synonymous variants, enabling *semantic aggregation*. We will demonstrate how UTOPIA enables effective pivot table construction, relieving technical novices from tedious data preprocessing while allowing them to remain in their familiar spreadsheet environment, without requiring external tools or additional expertise.

PVLDB Reference Format:
Whanhee Cho and Anna Fariha. UTOPIA: Automatic Pivot Table Assistant. PVLDB, 17(12): XXX-XXX, 2024.
doi:XX.XX/XXX.XX

1 INTRODUCTION

The recent growth of data volume and data utilization in machine learning demand effective mechanisms for data analysis and summarization. Aggregation is one of the most effective summarization methods, which involves many operations, such as "sum" over numerical attributes or "count" over categorical attributes. Mechanisms for creating aggregations range from SQL, which requires expert proficiency, to spreadsheet pivot tables, accessible by novices. SQL is suitable for experts, where (1) data must be clean, normalized, and stored in a relational format and (2) a query must be formed using aggregation operators (e.g., `SUM`, `AVG`, `CNT`). Alternatively, novices can utilize spreadsheet pivot table (Table 1(c)), which offers an intuitive and easy interface to group data by certain attributes (specified as "row") and apply aggregates to other attributes of interest (specified by "column").

Spreadsheets like Microsoft Excel[1] and Google Sheets[2] offer pivot table functionality for straightforward data analysis where users can use non-relational or "flat" data. However, spreadsheet pivot tables struggle with disorganized and messy data, preventing the users from obtaining the desired results directly. Disorganized data requires careful, manual preprocessing, demanding additional technical expertise, and time-consuming manual effort such as identifying and replacing all synonyms with a canonical value. Technical novices comprise approximately 11% of office workers [1], who lack programming expertise. While business intelligence models like Tableau[3] and Microsoft Power BI target such novices, they still require learning Power Query[4] or Regex to perform nontrivial data preprocessing, and offer no solution to automatically consolidate synonymous variants.

EXAMPLE 1. *Patel, a technical novice, is working with the IMDb[5] data about 1000 most popular movies (Table 1(a)). She wants to know which film genres yield significant gross. She uses Microsoft Excel pivot table, grouping by* `Genre` *and summing over* `Gross`*, and expects to obtain Table 1(c). To her disappointment, she gets Table 1(b), because* `Genre` *contains multiple values for some movies, such as "action, crime, drama", and Excel incorrectly assumes that this entire comma-separated list is the value for genre. To get her desired result, Patel must parse the multi-valued attribute* `Genre`*, which requires using an advanced Excel functionality or writing a Python script.*

However, Patel's struggle continues even after parsing. Some movies have only one genre, whereas some have up to 10 different genres. She must decide how to store such variable-length values against a single attribute `Genre`*. She can derive 10 different attributes labeled by* `Genre1`*,* `Genre2`*, ...,* `Genre10`*, but this will result into two issues. First, many cells will be empty as not all movies have 10 genres, resulting in poor data representation. Second, she will still struggle to obtain her desired pivot table (Table 1(c)). Generating one pivot table for each of the split 10 attributes will result in 10 different pivot tables, which is not what Patel wants. While Power Query may offer a solution, Patel hesitates to leave the spreadsheet environment, which she is very comfortable with.*

Proceedings of the VLDB Endowment, Vol. 17, No. 12 ISSN 2150-8097.
doi:XX.XX/XXX.XX

[1]Microsoft Excel www.microsoft.com/en-us/microsoft-365/excel
[2]Google Sheets: www.google.com/sheets/about/
[3]Tableau: www.tableau.com/
[4]Microsoft Power Query powerquery.microsoft.com/en-us/
[5]IMDb: www.kaggle.com/datasets/PromptCloudHQ/imdb-data

Title	Genre	Gross
Joker	drama	28 M
2001: A Space Odyssey	action, crime, drama	535 M
Queen	action, sci-fi	171 M
The Prestige	biography, drama	97 M
The Departed	action, sci-fi	293 M
The Usual Suspects	drama	37 M
Back to the Future	action, adventure	323 M

(a)

Genre	Sum of Gross
action, crime, drama	535 M
action, sci-fi	464 M
action, adventure	323 M
biography, drama	97 M
drama	65 M

(b)

Genre	Sum of Gross
action	1322 M
drama	697 M
crime	535 M
sci-fi	464 M
adventure	323 M
biography	97 M

(c)

Table 1: (a) A sample from the IMDb dataset. `Title` indicates title of a movie, `Genre` indicates the movie's genres, separated by commas, and `Gross` indicates the corresponding profit. (b) Since `Genre` contains multiple values, if the values are not parsed before pivot table construction, such an ill-formed pivot table is produced. (c) The desired pivot table.

Title	Genre	Gross
The Shawshank Redemption	prison drama	28 M
The Dark Knight	superhero action, crime, epic drama	535 M
The Matrix	action, epic sci-fi	171 M
Schindler's List	biography	630 M
Inception	team action, space opera sci-fi	93 M
Fight Club	drama	37 M
Star Wars	epic action, space adventure	323 M

(a)

Genre	Sum of Gross
biography	630 M
epic drama	535 M
crime	535 M
superhero action	535 M
epic action	323 M
space adventure	323 M
action	171 M
...	...

(b)

Genre	Sum of Gross
action	1029 M
biography	630 M
drama	600 M
crime	535 M
adventure	323 M
sci-fi	264 M

(c)

Table 2: (a) An example dataset from IMDb. (b) Interpreting this pivot table is challenging due to the presence of synonymous variants in `Genre`, such as misspellings and sub-genres. (c) The expected pivot table.

EXAMPLE 2. *Patel somehow parsed the multi-valued attribute `Genre` using an external tool (Microsoft Power Query) and is now working on a different movie dataset (Table 2(a)). She proceeds to generate a pivot table over this new dataset and obtains Table 2(b). Patel struggles to interpret the results as she expects "action" to be the top-gross genre, but the result indicates "biography" to have a larger gross than "action". With a closer look, she realizes that the data has different representations for the same genres: "action" has synonymous variants such as "superhero action", "team action", and "epic action". Patel wants to generate Table 2(c), but for that, she must replace all variants of similar semantics with a canonical value, which demands domain expertise and is a tedious process.*

Patel's examples show that generating a pivot table over disorganized data requires tedious data preprocessing. While some tools can parse multi-valued attributes, they often force users to leave the spreadsheet environment and use external tools requiring specialized skills. For instance, parsing multiple values in Microsoft Excel requires Power Query, Tableau needs regex, and user forums suggest using spreadsheet functions or Python. Even a simple parsing problem like Example 1 requires six steps in Python: loading data, splitting text, iteratively creating new attributes based on splitting results, concatenating results to the original data, removing the original attribute to avoid redundancy, and saving new data back to the CSV format. In contrast, relational databases expect normalization such as "genre" relation containing a single genre in each row, requiring understanding of normalization and restructuring flat data format to relational format, which is not novice-friendly.

Handling synonymous variants requires domain knowledge and entails tedious work as users must explore and modify variants manually. For instance, in Example 2, if each of the 20 genres has 25 variants, one will have to manually issue 500 find-replace operations! Moreover, this will fail for typos, case mismatches, and formatting issues. Synonymous variants mandate a thorough exploration of the entire data to determine which values have equivalent semantic meanings. Another critical issue is that such an operation is *irreversible*, the user will lose the information about variants once they consolidate synonymous variants explicitly.

Therefore, an ideal pivot table functionality should gracefully handle disorganized data and have three key features: (1) the ability to extract values from multi-valued attributes, (2) the capability to perform *semantic aggregation*, by grouping synonymous variants *implicitly*, without altering the data, and (3) allowing the user to stay in their preferred spreadsheet environment.

We introduce UTOPIA—an a**UTO**matic **PI**vot table **A**ssistant—to extend the functionality of spreadsheet pivot tables, overcoming data issues like multi-valued attributes and synonymous variants. UTOPIA automatically parses and organizes values for multi-valued attributes, providing *implicit* data normalization. Additionally, UTOPIA enables *semantic aggregation* using sentence embeddings and K-Means clustering to consolidate synonymous variants. Finally, UTOPIA produces a *dynamic* and *interactive* pivot table with options for expanding and collapsing data values to display synonymous variants.

Related work. Microsoft Excel FlashFill and Google Sheets SmartFill can split multi-valued attributes but require manual input of examples with uniform delimiters and knowledge of the resulting fragments, demanding significant user effort. Unlike UTOPIA, these tools cannot handle synonymous variants. Auto-Tables [4] keeps only the first value of a multi-valued attribute, addressing only the structural issue of the data while neglecting its content. Automatic data analysis tools [3, 5, 9] focus on data summarization, while UTOPIA reduces the data preprocessing barrier for technical novices, specifically for generating pivot tables in a spreadsheet environment.

In our demonstration, participants will observe how UTOPIA effectively generates the desired pivot table, bypassing issues such as multi-valued attributes and synonymous variants, without requiring additional effort from the user. We provide an overview of UTOPIA in Section 2 and a detailed walkthrough of our demonstration scenario, based on Example 2, in Section 3.

2 SYSTEM OVERVIEW

UTOPIA consists of three key components: (1) Multi-Valued Attribute Handler, which automatically identifies and parses multiple values within an attribute, (2) Synonymous Variants Handler, which identifies and semantically aggregates variants, and (3) Data Organizer, which generates an interactive pivot table.

Multi-Valued Attribute Handler. This component is responsible for detecting and parsing multi-valued attributes, thereby achieving data normalization. UTOPIA can identify multiple values, even if explicit delimiters are absent. For instance, for the value "superhero actioncrimeepic drama", UTOPIA recognizes the presence of multiple values and parses them by transforming them into a set: {"superhero action", "crime", "epic drama"}. To accomplish this, UTOPIA leverages a data extraction algorithm [7] that enables the extraction of multiple values even in the absence of explicit delimiters.

Synonymous Variants Handler. Next, UTOPIA searches for synonymous variants within each relevant attribute by computing the ratio r between the number of unique and total values. If r exceeds threshold $\tau_{lb} \in [0, 1]$, UTOPIA determines that synonymous variants *may* exist. E.g., values like "Sci-Fi", "Science Fiction", and "Cyberpunk" lead to a large r due to many unique values. However, a very high r indicates that unique values are natural for an attribute (people's first names). Therefore, when r does not exceed the threshold $\tau_{ub} \in [0, 1]$, while exceeding τ_{lb}, UTOPIA assumes presence of synonymous variants. While users can tune the parameters, we found the values $\tau_{lb} = 0.4$ and $\tau_{ub} = 0.8$ to work well in practice.

UTOPIA employs word embeddings to compute the similarity between values in the vector space. For instance, in Table 2, "action" and "superhero action" share the same semantic meaning, resulting in a small distance between them in the embedding space. Since traditional word embeddings [6] struggle to comprehend long phrases such as "space opera sci-fi", UTOPIA opts for state-of-the-art pre-trained sentence embedding model, SimCSE [2], which obtains superior performance over Sentence-BERT [8].

Next, UTOPIA consolidates semantically similar values using K-Means clustering. It suggests the value of k based on the best *silhouette score*, a metric for measuring cluster quality. However, k is a customizable: for fine-grained grouping, k should be set to a large value, and for a more generalized view, k should be smaller. Note that traditional relational databases do not offer such flexibility due to strict schema requirements.

Data Organizer. Finally, UTOPIA displays a pivot table with parsed data and semantically aggregated variants. UTOPIA represents parsed values as row or column labels in the pivot table. UTOPIA displays a representative value (e.g., "action" is chosen as representative for "action", "superhero action", etc.) in the presence of synonymous variants. The representative value is the one with the closest embedding to the average of the embeddings over all variants.

To ensure data integrity, UTOPIA stores the parsed data in JSON format, which is ideal for storing multi-valued attributes and avoids repetitive parsing computation for subsequent operations. Moreover, UTOPIA is robust to data updates: instead of recomputing clusters for minor data changes, it assigns the new data to the most similar cluster. Note that our focus is on *usability* enhancement and our target platform is spreadsheets, which doesn't support big data, indicating no scalability challenge is involved.

Preliminary results: Using the IMDb dataset with 27 expected genres, UTOPIA achieved a cluster purity score of 0.86, indicating how much each cluster contains semantically similar values. We also tried ChatGPT 3.5 using the prompt "Group the following words into semantically related groups. Don't change or omit words. Create k groups." for different values of k. However, ChatGPT behaved undesirably when k is smaller than ideal. With $k = 10$, ChatGPT forms the following groups: {"Action and Adventure", "Crime and Documentary",...}. While ChatGPT succeeds in including all subgenres of "Action" and "Adventure" into "Action and Adventure", it incorrectly merges groups based on their lexicographic similarity: "Action" is alphabetically close to "Adventure" but not semantically. Ideally, "Action" should merge with "Thriller" or "Crime" over "Adventure". In contrast UTOPIA provides semantically meaningful groups with even when fewer clusters are requested, usually for enhanced interpretability.

3 DEMONSTRATION SCENARIO

We will demonstrate UTOPIA across diverse domains, including recipe data listing multiple ingredients[6], IMDb data with various sub-genres, and university survey across departments[7]. Below we provide a demonstration scenario over part of the IMDb dataset, which contains metadata for the top 1000 successful movies, with eight attributes (movie title, year, genre, etc.). We randomly introduced some misspellings and augmented this data with sub-genres. We will guide users through eleven steps (annotated in Figure 1) impersonating Patel, who is interested in generating a pivot table to find top-grossing movie genres for each year.

Steps Ⓐ & Ⓑ (Uploading and overviewing data) In step Ⓐ, the user uploads a (disorganized) data containing multi-valued attributes and synonymous variants. In this case, the IMDb dataset for the top 1000 successful movies. In step Ⓑ, the user previews the data. UTOPIA displays the first three rows. The user can horizontally or vertically scroll to see more data.

Steps Ⓒ & Ⓓ (Selecting attributes and choosing positions) The user selects the attributes they want to focus on for the pivot table generation in step Ⓒ. In our scenario, the user chooses `Year`, `Genre`, and `Gross`. They can assign a selected attribute to one of the pivot table positions, "Column", "Row", or "Value", by dragging

[6] www.github.com/majumderb/recipe-personalization

[7] www.kaggle.com/datasets/sank3t/university-student-survey

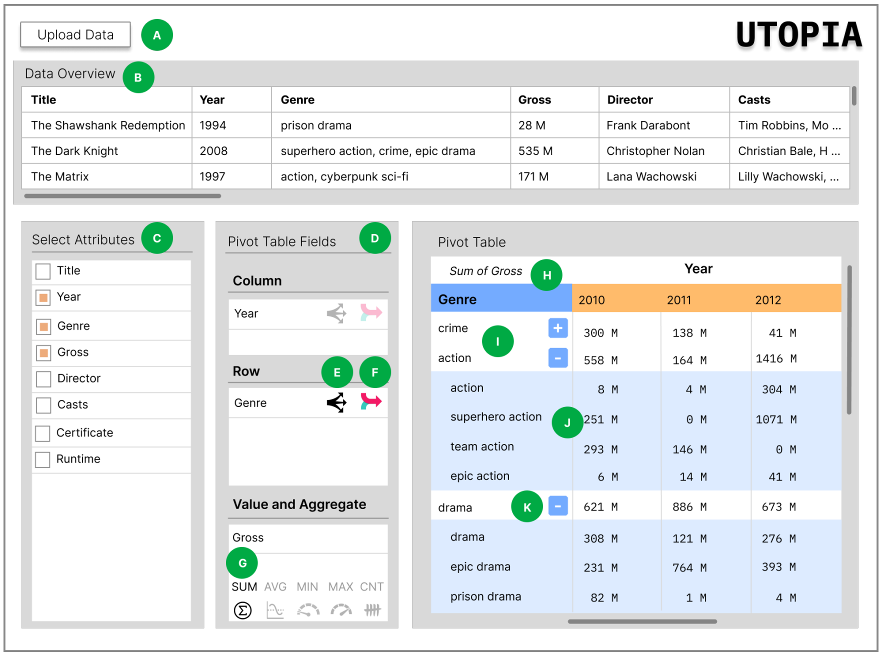

Figure 1: The UTOPIA demo: (A) upload data, (B) view data overview, (C) select attributes, (D) choose an attribute position for the pivot table, (E) select multi-valued attribute handler, (F) select synonymous variants handler, (G) choose aggregation method, (H) view aggregated attribute, (I) view the parsed data of multi-valued attributes, (J) view semantically aggregated values from synonymous variants, (K) extend or collapse items.

it to the desired position in (D). When an attribute moves to "Row" or "Column", its values become the row or column labels in the pivot table, respectively. If an attribute is selected for "Value", it will be aggregated. In our guided scenario, the user chooses `Year` for column, `Genre` for row, and `Gross` for value.

Steps (E) & (F) (Enabling multi-valued attribute and synonymous variants handler) Next to each row/column attribute, UTOPIA displays an icon for multi-valued attribute handler. This icon is gray (disabled) if the attribute does not contain multiple values. In our guided scenario, the icon for `Year` is disabled, while the icon for `Genre` is enabled (step (E)). Next to this icon, UTOPIA shows another icon for the synonymous variants handler (step (F)), which is gray if no synonymous variants are present. The user can turn on or off these functionalities. By right-clicking these icons, the user can specify system parameters τ_{lb}, τ_{ub}, and k (not shown in the figure).

Step (G) (Choosing an aggregation method) The user selects `SUM` as the aggregation method over `Gross`.

Steps (H)–(J) (Viewing the pivot table) UTOPIA now produces a pivot table in the bottom right panel (step (H)). All values for `Gross` for the `Genre` "action" are aggregated, showing a sum of `558 M` for `2010`. UTOPIA automatically parses the values within the multi-valued attribute `Genre` and organizes them as row labels (step (I)). The user can also view semantic aggregation of synonymous variants (step (J)). UTOPIA semantically aggregates these variants and shows the representative value on top.

Step (K) (Expanding or collapsing variants) The user can expand (collapse) the row labels to show (hide) the synonymous variants. E.g., expanding "action" reveals four synonymous variants.

After the guided demonstration, participants may use UTOPIA to explore their own datasets. The key takeaway is the convenience UTOPIA provides for creating pivot tables over disorganized data, without requiring any additional user skills. In summary, UTOPIA targets technical novices who seek to avoid technically challenging and time-consuming data preprocessing while generating pivot tables in a spreadsheet environment.

ACKNOWLEDGMENTS

This work was supported by the One Utah Data Science Hub Seed-Grant Award.

REFERENCES

[1] 2024. Excel Facts & Statistics: New Research 2024. `www.acuitytraining.co.uk/news-tips/new-excel-facts-statistics-2022/`.
[2] T Gao, X Yao, and D Chen. 2021. SimCSE: Simple Contrastive Learning of Sentence Embeddings. In *EMNLP*. 6894–6910.
[3] R J L John, D Bacon, J Chen, U Ramesh, J Li, D Das, R V Claus, A Kendall, and J M Patel. 2023. DataChat: An Intuitive and Collaborative Data Analytics Platform. In *SIGMOD*. 203–215.
[4] P Li, Y He, C Yan, Y Wang, and S Chaudhuri. 2023. Auto-Tables: Synthesizing Multi-Step Transformations to Relationalize Tables without Using Examples. *PVLDB* 16 (2023), 3391–3403.
[5] P Ma, R Ding, S Wang, SHan, and D Zhang. 2023. InsightPilot: An LLM-Empowered Automated Data Exploration System. In *EMNLP*. 346–352.
[6] T Mikolov, K Chen, G Corrado, and J Dean. 2013. Efficient Estimation of Word Representations in Vector Space. In *ICLR*.
[7] M Raza and S Gulwani. 2017. Automated Data Extraction Using Predictive Program Synthesis. In *AAAI*. 882–890.
[8] N Reimers and I Gurevych. 2019. Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks. In *EMNLP*. 3980–3990.
[9] A Wu, Y Wang, M Zhou, X He, H Zhang, H Qu, and D Zhang. 2022. MultiVision: Designing Analytical Dashboards with Deep Learning Based Recommendation. *EEE Trans Vis Comput Graph* 28, 1 (2022), 162–172.