Drag, Drop, Merge: A Tool for Streamlining Integration of Longitudinal Survey Instruments

Juseung Lee*
Pratik Pokharel*
juseungl@buffalo.edu
pratikpo@buffalo.edu
University at Buffalo
Buffalo, NY, USA

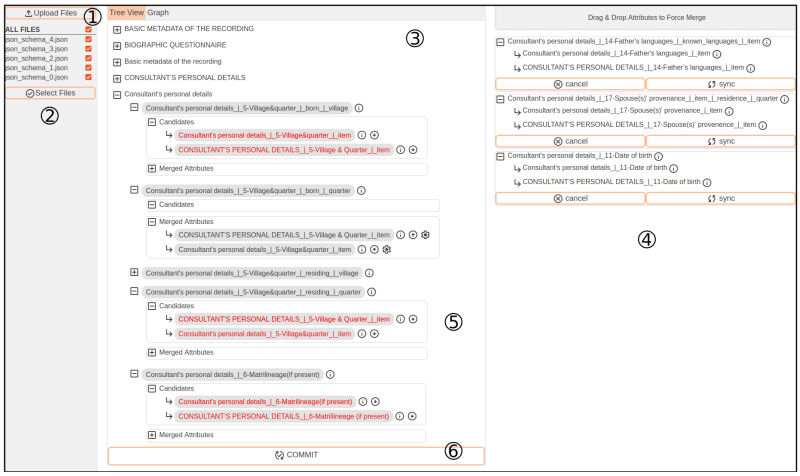
Marianthi Markatou
Andrew Talal
Raktim Mukhopadhyay
markatou@buffalo.edu
ahtalal@buffalo.edu
raktimmu@buffalo.edu
University at Buffalo
Buffalo, NY, USA

Jeff Good
Oliver Kennedy
jcgood@buffalo.edu
okennedy@buffalo.edu
University at Buffalo
Buffalo, NY, USA

① The user uploads a group of source datasets

② The user selects datasets for documenting attribute relations

③ The treeview is generated with potential merge candidates



④ The user can forcefully merge attributes by Drag and Drop

⑤ The user validates potential merge candidates

⑥ The user commits the changes to document attribute relations

Figure 1: DDM's User Interface

ABSTRACT

We explore data management for longitudinal study survey instruments: (i) Survey instrument evolution presents a unique data integration challenge; and (ii) Longitudinal study data frequently requires repeated, task-specific integration efforts. We present DDM (Drag, Drop, Merge), a user interface for documenting relationships among attributes of source schemas into a form that can streamline subsequent efforts to generate task-specific datasets. DDM employs a "human-in-the-loop" approach, allowing users to validate and refine semantic mappings. Through a simulation of user interactions with DDM, we demonstrate its viability as a way to reduce cognitive overhead for longitudinal study data curators.

*The first two authors contributed equally.

HILDA 24 , June 14, 2024, Santiago, AA, Chile

ACM ISBN 979-8-4007-0693-6/24/06
https://doi.org/10.1145/3665939.3665965

ACM Reference Format:
Juseung Lee, Pratik Pokharel, Marianthi Markatou, Andrew Talal, Raktim Mukhopadhyay, Jeff Good, and Oliver Kennedy. 2024. Drag, Drop, Merge: A Tool for Streamlining Integration of Longitudinal Survey Instruments. In *Workshop on Human-In-the-Loop Data Analytics (HILDA 24), June 14, 2024, Santiago, AA, Chile.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3665939.3665965

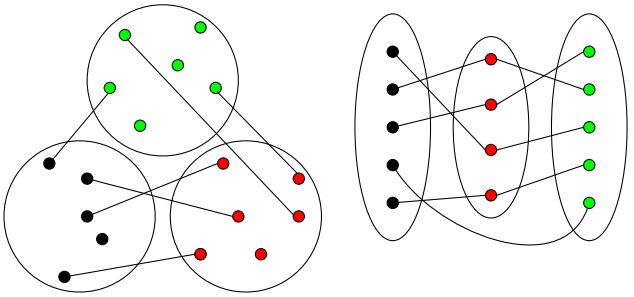
1 INTRODUCTION

In longitudinal studies[1], survey instruments (i.e., questionnaires) evolve over years, or even decades. Questions may be added, removed, or adjusted to account for changes in the social context in which the study is administered, or as the researcher's understanding of the study domain improves. For example, consider a longitudinal study of school students. In the midst of the study, researchers observe an increase in noise in answers to a question about the family's postal code. After examining the situation, the researchers are able to attribute the noise to changing social factors that result in fewer respondents knowing their postal code. In this situation, the researchers may opt to modify the question (e.g., to ask for the student's school district instead), reducing noise, albeit at the cost of creating heterogeneity in the data.

[1]Although we use longitudinal studies as a motivating example, similar challenges arise in other settings like dataset archival, and meta-studies designed around creating publishable datasets.

Chronological Schema Evolution. Integration of heterogeneous data (i.e., entity resolution) is a well studied problem [2–5, 7, 12, 18]. A central problem in prior work on data integration is the massive number of integration options available: Each attribute in every dataset, could potentially align with any other attribute from any other dataset. This overwhelming number of options poses both a computational challenge as the runtime of naive algorithms grows quadratically with the number of datasets; and a cognitive challenge, as any errors in predicted aligning attributes require the user to manually search through a large number of potential matches.

By contrast, longitudinal data lacks this constraint: schema integration in longitudinal studies can consider each schema chronologically, as illustrated in Figure 2. Survey instruments forming a chronological chain (i.e., survey instruments in adjacent years are very similar, even if different) admit multiple opportunities for improved algorithms and integration interfaces with lower cognitive overheads. For example, the questions from adjacent years in Figure 3 are similar (i.e., Form 1 and 2, and 2 and 3), so merging adjacent schemas requires less cognitive overhead.



(a) Classical Ssetting **(b) Longitudinal Setting**

Figure 2: Classical Schema Integration vs. Longitudinal Schema Integration

With longitudinal schemas, if Form 1 and 2 are already merged then by transitivity, Form 3 only needs to be matched against 2. So, integrating similar questions between adjacent forms and building up on that iteratively would require 'small', 'easy' merges and is less cognitively demanding, as opposed to attempting all at once.

Hierarchical Form Structure. An additional feature of survey instruments over other schemas in general is that questions in survey forms are categorized hierarchically. For instance, survey forms 1 and 2 in Figure 3 have a category 'Family' and questions under it are (a) and (b). Grouping these questions under the same category and ordering them in a logical sequence emphasizes their relatedness. Similarly, Form 3 in Figure 3 has a category 'Family Details' and asks (a). The semantic similarity of these categories is high, which can be exploited by clustering them together in the user interface. This enables users to search for matching questions within the cluster, reducing the amount of interaction required.

Integration Reusability. In addition to opportunities for complexity reduction, data collected through longitudinal studies often serves as the basis for multiple distinct research efforts, each of which has its own requirements of the integrated dataset. Continuing our example, contrast one research effort trying to understand an educational phenomenon with respect to county-level policies, against another effort exploring the impact of town-level policies. Both postal codes and school districts uniquely identify counties, so the former study would be satisfied with a published dataset that identifies records at this granularity. The latter study, on the other hand, would not be satisfied by county-level location data, and instead requires a more nuanced data integration process — for example one that discards records identified by zip code. Researchers must frequently make such trade-offs between precision, number of records, and data completeness. However, the optimal point on the trade-off curve varies by the needs of each research effort, often requiring researchers to duplicate data integration work.

Form 1
1. Degree of Schooling
2. Family
   (a) Do you have children?
   (b) How many? What are their ages?
3. Consultant's personal details
   (a) Village & Quarter

Form 2
1. What are the schools that you attended?
2. Family
   (a) Do you have children?
   (b) How many? What are their ages?
3. Consultant's personal details
   (a) Village & Quarter

Form 3
1. Schooling (list all the schools attended, with location and remarks on schoolmates provenance and languages)
2. Family Details
   (a) List out the names and ages of your children
3. Biographic Questionnaire
   (a) Current Residence? Village, quarter, compound

Figure 3: Survey questions from three years, listed chronologically by year [11]

In this paper, we propose DDM (Drag, Drop, Merge), a data integration tool aimed at reducing per-use data integration effort in analytics over longitudinal study data. As input, DDM takes a set of schemas: one for each individual revision of a survey instrument, where each attribute of the schema corresponds to the answer to one specific question. DDM aims to simultaneously assist the researchers conducting the longitudinal study in data publication; as well as researchers attempting to leverage the published data, as they integrate it for the needs of their individual research efforts. The key insight behind DDM is to split the data integration process into two phases: (i) a curation phase in which study designers document relationships between the attributes of the source datasets; and (ii) a integration phase in which researchers leverage this documentation to explore the skyline of quality trade-offs, and generate a dataset that meets their needs. Although our primary focus for this paper is the first, curation phase, we first explore the needs of the second, integration phase.

Target Audience. DDM is developed through an interdisciplinary collaboration involving experts from linguistics, medicine, and statistics. The target group for DDM is domain experts conducting longitudinal studies in their respective fields. These experts are

already familiar with the process of data integration, which they currently perform manually. The design considerations for DDM evolved based on the feedback from preliminary usability tests conducted with these collaborators, the prospective users of DDM.

1.1 Generating Task-Specific Datasets

The primary objective of researchers at this phase is to trade off between the quantity of data that they plan to analyze, and the completeness/precision of that data. As the researcher incorporates more years of data from the longitudinal survey, the number of questions missing from one or more revisions of the instrument grows. In some cases (e.g., counties in our running example), some attributes may be available at a reduced precision; or only in approximation for some source datasets. In other cases, questions may only have answers that are comparable in specific situations. For example, consider a question that asks respondents to list dialects that they speak, and a revision in which a specific dialect is introduced as an example. Such a revision may have a negligible impact on some analyses (e.g., if the new example dialect has few speakers), but could have a drastic effect on others (e.g., a study involving the dialect itself).

The goal of the curation phase, then, is to document the relationships between attributes, giving researchers using the published data (i) a starting point for their integration task, while simultaneously (ii) retaining the provenance of those relationships to allow researchers to refine the integrated dataset, based on study-specific criteria. In classical approaches to data integration, the objective is usually a single, clean, redundancy-free target schema. By contrast DDM's curation step adopts a flat, graph-based schema model. The focus of the model, and accordingly of the curation step, is on documenting relationships between attributes, and not on creating a single global schema. We discuss the model further in Section 3.

1.2 Curating Longitudinal Study Data

DDM's schema curation interface focuses on helping study designers in identifying relationships between attributes (questions) from distinct survey instruments and their revisions, specifically: (i) Pairs of questions from different revisions where answers to one or both may be derived from the other, and (ii) New, "least-upper-bound" attributes that may be derived from answers to related questions where no direct interchange is possible. Although this is fundamentally similar to a classical data integration process, DDM's use of a flat, relationship-based model, rather than one based on creating a single target schema, substantially increases the number of attributes that the user must sift through. Like other approaches to integration, DDM heuristically generates an approximate schema matching as a starting point. However, when this approximation is incorrect (e.g., if it misses a pair of equivalent questions), the survey designer must sift through a substantially larger number of attributes to manually declare the match. Worse, with a large number of attributes, it becomes easier for the designer to miss an unlisted equivalence.

Our key contribution here is to note that the order in which schemas are integrated can significantly affect the cognitive overhead placed on DDM curators. Specifically, we observe that asking users to perform many isolated 1-1 schema matchings (e.g., pairwise matching between schema revisions), can actually result in *less* work than asking them to match all schemas at once. Even though the latter approach involves less work overall, having more attributes under consideration at the same time creates a higher cognitive overhead for the curator when the approximate initial matching is incorrect.

Outline. The rest of the paper is organized as follows. Section 2 discusses related work, including table union search, schema evolution, and user interfaces for data integration. In Section 3, we introduce the DDM system and its schema model, and outline key features in the design of its interface. Section 4 reports the results of a simulation study, demonstrating how DDM can streamline the cost of integration. Section 5 briefly discusses our longer term vision for the DDM system, and we conclude in Section 6.

2 RELATED WORK

Schema Evolution. PRISM workbench [3] addresses schema evolution by using theories of mapping composition and invertibility. Its concise Schema Modification Operators (SMOs) language enables efficient representation of schema changes, facilitating predictability and automating verification processes for information preservation and query support. PRISM adopts a similar evolutionary model of schema evolution, but focuses on providing query compatibility with specific schema revisions, rendering certain attribute combinations inaccessible. Adaptive Schema Databases [19] explores query support for ambiguously defined schemas, but assumes that a space of possible schemas has already been defined. DDM could be used to create such a space.

Table Enrichment. As discussed in [14], table enrichment entails identifying unionable (resp., joinable) tables for a provided query table: Unionable tables contribute new tuples, while joinable tables add new attributes. Although DDM adopts similar mechanisms, table enrichment focuses on discovery, rather than integration.

UIs for Data Cleaning and Alignment. In Clio [7], users can view, insert, and delete correspondences between schemas. Additionally, users can assign transformation functions to these correspondences and inspect and modify the resulting logical mappings. However, the system has scalability issues when schemas are large because it can take exponential time in the worst case as it explores exponential path variable assignments. Muse [1], built on top of Clio, uses a series of simple data examples to discern between alternative mapping specifications and deduces the desired mapping semantics based on the actions of the designer. Wrangler [10] is an interactive data transformation tool that uses a spreadsheet-style interface allowing direct manipulation of visualized data with automatic deduction of pertinent transforms. Unlike conventional tools focusing on pairwise settings, SMART [16] addresses scenarios involving a large number of schemas through a pay-as-you-go approach that leverages integrity constraints. GestureDB [9], a database architecture system for keyboardless database interaction, includes a high-performance unionability index to allow low-latency gestural specification of unions.

3 SYSTEM OVERVIEW

3.1 Overview

DDM's system architecture is shown in Figure 4. In DDM, a list of source schemas is uploaded to the workspace, and a subset (or

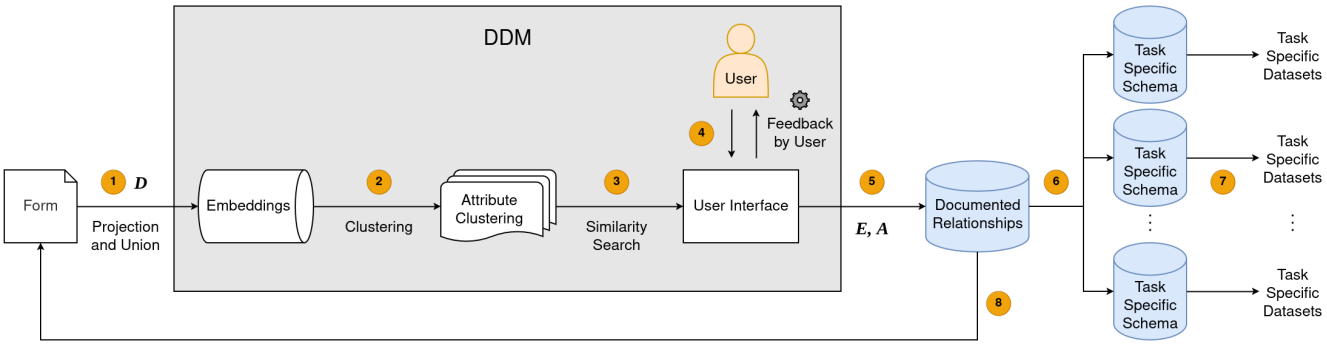


Figure 4: System Architecture

all) of it can be selected to document relationships among source attributes in ⑤. These documented relationships can then be transformed into study-specific schemas ⑥, and corresponding datasets ⑦. Without the documented relationships, researchers with new requirements must revisit the entire integration process for each new study; DDM kick-starts this integration process through a schema relationship graph (defined formally below).

We would like to avoid the need for schema curators to define this graph manually. Instead, DDM streamlines the process by proposing a schema relationship graph for review and manual correction. As shown in ④, the user validates the candidates, applies corrections as needed, and potentially marks specific relationships as requiring a mapping function. As part of this process, DDM uses the graph to warns users about attributes that have not been reviewed or matched. Once the user is satisfied, they commit their changes to a database of documented relationships.

In order for DDM to propose a preliminary graph to the curator, we needed a metric that could quantify the similarity between two attributes. Many such measures exist, relating categorical and prose data [14, 15], numerical data [20], or semantics [8]; but frequently rely on the data behind the schemas. We adopted a simpler semantic measure based on the question contents [13], leveraging vector embeddings of attribute names (i.e., question plaintext and their containing context). DDM identifies related questions from different instrument revisions by performing a top-k similarity search using Faiss[6], a library tailored for efficient similarity search and clustering of dense vectors in ③. Two attributes (questions) are considered to be related if their cosine distance exceeds a threshold, α. A higher value of α decreases false positives in the candidates list, but increases the number of false negatives.

We assume that the schemas are provided by the survey designer. Our initial approach is modeled after a collection of medical intake forms, used in a recent study on opioid use disorder[13]. The structure of the forms (i.e., the schema) had already been digitized into a hierarchical Json schema mirroring each individual instrument's structures and questions. Each section of the form and its questions are represented as nested objects, aligned with their hierarchy in the forms. Questions are encoded according to their type (e.g., free text, number, multiple choice) with appropriate data fields, including enums for multiple choice questions. This structure facilitates efficient data entry and supports automated validation and management of evolving schemas. To make the attributes easier to work with, we first project the hierarchical source schema into a flat collection of attributes. These projected attributes from multiple source forms are identified by the question text and elements from the hierarchy of section headers. The set of attributes (denoted $\mathcal{D}$ in ①) is the set of all projected attributes. $\mathcal{D}$ is transformed into vector embeddings using a pre-trained model [17] and clustered in ② based on the sentence embeddings of their root question category. Clustering the questions has two advantages: (i) it preserves the order of the question asked in original source schema up to some extent, and (ii) it minimizes the user effort to find an attribute from similar root question categories.

The entire process is iterative and the information obtained from human interaction is used in future iterations to reduce required calculations for similarity search and suggest preciser candidates.

3.2 Model

Definition 3.1. (Dictionary). A schema set $\mathcal{S} = \{S_1, \ldots, S_N\}$ is a set of disjoint source schemas, where $S_i = \{a_{i1}, \ldots, a_{ij}\}$ is a set of attributes in the schema S_i. We typically assume that each pair S_i and S_j (where $i \neq j$) is fully disjoint. An attribute a_{ij} indicates j^{th} attribute of schema S_i. A dictionary $\mathcal{D}$ is the union of all of the attributes in source schemas: $\mathcal{D} = \bigcup_{i=1}^{N} S_i$.

Definition 3.2 (Correspondence). A *lossy correspondence* $\langle a_j, a_k, f \rangle$ defines a relationship between a pair of attributes a_j, a_k from two distinct source schemas $a_j \in S_j, a_k \in S_k$ through a function $f_{j,k}$ that maps values of attribute a_j into values of attribute a_k. If $f_{j,k}$ is invertible, we call it a *lossless correspondence* $\langle a_j, a_k, f, f^{-1} \rangle$.

Definition 3.3 (Relationship Set). A *relationship set* is a mixed graph $G : (\mathcal{D}, E, A)$, where E is a set of undirected edges representing *lossless correspondences*, and A is a set of directed edges representing *lossy correspondences*.

We emphasize the choice to explicitly distinguish lossless and lossy edges as undirected and directed edges, respectively. This treatment simplifies the handling of bidirectional transformations and allows for efficient processing of inverse relationships. A correspondence is either (i) lossless, or (ii) lossy. If $f(j) = k$ and f^{-1} exists such that $f^{-1}(k) = j$, then the correspondence is lossless and is denoted as a tuple $\langle a_j, a_k, f, f^{-1} \rangle$. For instance, temperature unit conversion from Celsius to Fahrenheit is a lossless correspondence. On the other hand, the correspondence is lossy if $f(j) = k$ and f^{-1} does not exist, denoted as tuple $\langle a_j, a_k, f \rangle$. An example of

a lossy correspondence is the mapping from school district and zip code to county from the introduction.

Definition 3.4 (Global Attribute). A *global attribute* is a fully connected component C in G: $C = \{v_1, v_2, \ldots, v_n\} \subseteq V$, where $\forall v_i, v_j \in C, i \neq j, \exists e_{ij} \in E \cup \exists a_{ij} \in A$.

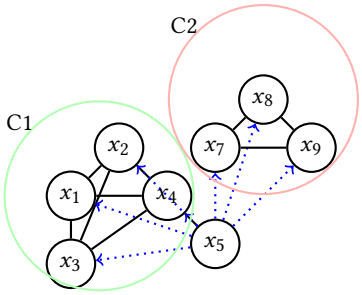


Figure 5: Global attributes as fully connected components. Node x_5 has lossy correspondence to both C_1 and C_2 and belongs to both of them

If we can find a correspondence for a vertex v with any one of the vertices v_i in the fully connected component C_k, then v can be assigned to C_k. Similarly, a vertex v can have a lossy correspondence to multiple fully connected components C_k and can be assigned to all of them. For every vertex $v, v' \in V$, if v has a lossy correspondence to the fully connected component C_k and v' has a lossless correspondence to v, then we remove v from C_k and form a new fully connected component C'_k by including v and v'.

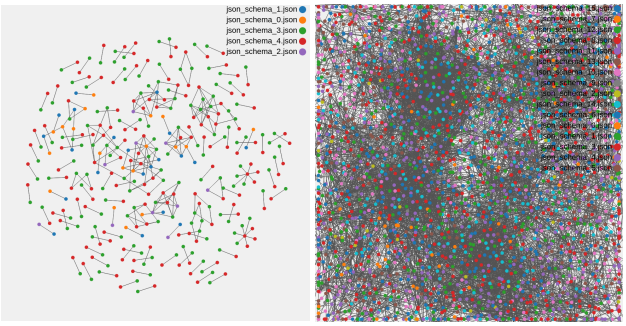
3.3 Interface Design Considerations

Our choice of the mixed graph model for DDM was backed by its flexibility in including various types of mappings and transformations between different data elements. This modular approach facilitates the incremental development of integration workflows. In addition, the features of connected components in a graph can be leveraged to reduce computation time and document more refined mappings.

The primary goal of DDM's user interface is to produce the relationship set G for a set of source schemas $\mathcal{S}$. As a completion measure, DDM requires every attribute in $\mathcal{D}$ to be associated with a global attribute; If the attribute is a singleton in the relationship set, we require the user to explicitly confirm this fact, to avoid accidentally omitted attributes.

We considered two design options for the user interface: (i) allowing users to manually explore the top-k similar candidates list generated by Faiss, for each attribute in $\mathcal{D}$, and (ii) suggesting the mergeable candidates (based on the Faiss metric) for all attributes in $\mathcal{D}$ and making them visible as a list in the UI. The former approach allows more control to the user, in terms of exploring mergeable candidates. However, based on the preliminary usability study with our area-expert collaborators, we observed that it required far too many user interactions to generate (E, A), because the user had to click each attribute in $\mathcal{D}$ and choose from their top-k list; we ultimately adopted the latter approach. We wanted to preserve the order of questions asked in the source schemas, so the attributes in $\mathcal{D}$ are displayed in tree-view format as shown in Figure 1. Here, each node is a cluster that represents root category of a question. Each attribute appears with a list of potential merge candidates. To allow users to distinguish the strength of similarity between an attribute in $\mathcal{D}$ and the merge candidates, the merge candidates are ordered in descending order of the similarity scores, appearing with a color gradient as shown in Figure 1.

Since the user begins with a proposed relationship set, there are three fundamental interactions that the user needs to perform in DDM: (i) confirming True Positives (correctly predicted merge candidates), (ii) rejecting False Positives (incorrectly predicted merge candidates), and (iii) repairing False Negatives (missed merge candidates). Confirming true positives can be done by dragging a candidate attribute from candidates branch to merge branch of the corresponding global attribute in the tree-view. Untouched attributes are rejected as false positives. We note that a key challenge in resolving false negatives is the need to find the mapped attribute among the list of available mappable attributes. To mitigate the scrolling distance that the user needs to drag a value, as shown in Figure 1, DDM provides a separate temporary merge area where attributes can be dragged from tree-view of $\mathcal{D}$, dropped in the pane, and merged.



(a) Lesser number of source attributes **(b) Large number of source attributes**

Figure 6: Graph Visualization of Relationships Among Source Attributes

A visualization tab, powered by D3 was created to allow users to visualize the status of the identified relationships in the current workspace and quickly locate global attributes (i.e., connected components in the graph). The initial approach allowed the user to see all relationships of attributes in $\mathcal{D}$, but it did not provide a clear indication of which attributes were candidates for merging when there were many source attributes. To provide users with a clear visualization, the latter approach only displays attributes with their merge candidates in the visualization tab. As shown in Figure 6, the graph was useful as long as the number of attributes was small in the workspace since it fills the graph with a thick cloud of nodes and edges that hinders visualization as the number of attributes grows. It also explains why one should not want to integrate many things at once.

A few fundamental considerations were made in terms of the user interface to improve the user experience: (i) use of a fixed-width panel to display candidate attributes instead of a single line panel with a horizontal scroll bar, (ii) auto-collapse of attributes with candidates available (to reduce the need for vertical scrolling), and (iii) use of a brighter color gradient (e.g. red) for the appearance of candidates list.

4 EXPERIMENTS

Experiments were performed to assess the impact of the following factors on building mappings: (i) the order of source datasets subjected during merging, and (ii) the number of attributes attempted to merge at once (i.e. size of $\mathcal{D}$).

Dataset. We used a collection of medical intake forms (schemas only) used in a recent Social Determinants of Health (SDOH) study on opioid use disorder[13]. The subset of the form layouts we used involved three distinct instruments, including medical intake forms from two sites (i.e., clinics) A and B, with form revisions denoted a_1 and a_2, and b_1, respectively.

Overview. We were interested in the relationship between the effort required to merge schemas and the order in which integration was performed. To understand this relationship, we simulated merging between the source schemas belonging to the same sites as well as different sites was simulated in different orders. Recall that three classes of user input are required: (i) Confirming a predicted attribute relationship (TP), (ii) Rejecting a predicted attribute relationship (FP), and (iii) Adding a non-predicted attribute relationship (FN). Additionally, recall that adding a non-predicted attribute relationship requires exploring the full list of available attributes. We simulate this by measuring how far two mergeable candidates (missed by similarity search) are in the tree-view, where two attributes appearing one after another in the treeview have a distance of 1. This metric is summed over all the False Negative pairs. We set the word embedding threshold α to 0.7.
We considered three different orders:
Experiment 1 $[(a_1+a_2)+b_1]$: First, the schemas belonging to site A were merged, followed by a schema belonging to site B.
Experiment 2 $[(a_1+b_1)+a_2]$: First, one schema from each of sites A and B were merged, then the remaining schema of site A.
Experiment 3 $[a_1+a_2+b_1]$: All three source schemas were merged simultaneously.

4.1 Experimental Results

The comparison of the number of validations required and cognitive overhead for the three different approaches of experiments is shown in Table 1. The union of the TP, FP, and FN scores across all the steps for all the experiments were found to be the same. Meaning that the order of merging has no bearing on the merge candidates generated altogether. This is due to the fact that the final dictionary $\mathcal{D}$ would eventually be the same for each of the approaches and the semantic similarity search would generate the same results overall. However, it significantly affected the effort to find False Negative pairs. In experiment 3, we found that merging all schemas at once would result in a higher overhead than merging two schemas in a single step and building up on that in further iterations. This is because having more attributes under consideration at the same time creates a higher cognitive overhead for the curator when there are False Negatives to identify manually.

Experiment	Step	TP	FP	FN	∑ Distance to FN
1	1	16	3	7	356
	2	2	1	1	8
2	1	2	1	1	4
	2	16	3	7	476
3	1	18	4	8	562

Table 1: Comparison of cognitive overheads for merging schemas from same sites first vs. merging schemas from a different site first

5 FUTURE WORK

DDM is a proof of concept, but substantial work remains to be done to identify pain points. We plan to conduct comprehensive expert and user studies to confirm our hypothesis that incremental integration can produce a more reliable relationship set. If this hypothesis is supported, we will explore the design of a recommendation system that suggests an order of source datasets in which merging should be performed. For this work, we adopted a simpler semantic similarity measure. Unlike classical database schemas with attribute names identified by short keywords or codes, survey questionnaires are often succinct, clearly informative, and sometimes verbose. However, we understand this approach may not be entirely sufficient. We will further explore the possibilities of using ontologies, taxonomy, and most importantly profiling actual data values behind the schemas. These methodologies might further help identify similarities.

The dataset [13] used in Section 4 contains multiple years of medical intake forms from multiple sites and the results in Section 4.1 show that site has a significant impact on the cognitive overhead of merging schemas, so we plan to apply a hybrid strategy that combines our methods with more classical techniques.

In this paper, we focus exclusively on schema integration; Leveraging these mappings for subsequent analysis is left to future work. The key challenge here is helping users to identify which attribute relationships can be safely leveraged. We plan to explore strategies for measuring whether the form and/or question variant acts as a meaningful predictor of participant responses, and for helping users to visualize relationships between the quantity and quality of data extracted.

Similarly, as in to PRISM [3] and similar works, it may be impossible to define some schema mappings precisely, a problem that only becomes more acute in survey instruments. For example, minor changes in phrasing can substantially impact how respondents interact with it. End even if a question is unaltered, changing social context (e.g., the relative value of a dollar) may affect the comparability of answers from across instrument revisions. We will explore how such partial relationships can be captured, and safely incoroporated into the dataset construction phase.

6 CONCLUSION

This paper describes an initial implementation of DDM- an approach to simplifying the documentation of attribute relationships within a pool of source schemas. By employing a user-friendly interface and a human-in-the-loop approach, DDM facilitates the validation and refinement of semantic mappings, with a goal of reducing cognitive overload for data curators. DDM aims to aid in efficient management of longitudinal study data integration tasks, offering researchers the flexibility to generate task-specific datasets tailored to their unique requirements.

REFERENCES

[1] Bogdan Alexe, Laura Chiticariu, Renee J. Miller, and Wang-Chiew Tan. 2008. Muse: Mapping Understanding and deSign by Example. In *2008 IEEE 24th International Conference on Data Engineering*. 10–19. https://doi.org/10.1109/ICDE.2008.4497409

[2] Sudarshan S. Chawathe, Hector Garcia-Molina, Joachim Hammer, Kelly Ireland, Yannis Papakonstantinou, Jeffrey D. Ullman, and Jennifer Widom. 1994. The TSIMMIS Project: Integration of Heterogeneous Information Sources. In *Proceedings of the 10th Meeting of the Information Processing Society of Japan, Tokyo, Japan, October 1994*. 7–18.

[3] Carlo A. Curino, Hyun J. Moon, and Carlo Zaniolo. 2008. Graceful database schema evolution: the PRISM workbench. *Proc. VLDB Endow.* 1, 1 (aug 2008), 761–772. https://doi.org/10.14778/1453856.1453939

[4] Hong-Hai Do and Erhard Rahm. 2002. COMA: a system for flexible combination of schema matching approaches. In *Proceedings of the 28th International Conference on Very Large Data Bases (VLDB '02)*. VLDB Endowment, Hong Kong, China, 610–621.

[5] AnHai Doan, Pedro Domingos, and Alon Y. Halevy. 2001. Reconciling schemas of disparate data sources: a machine-learning approach. In *Proceedings of the 2001 ACM SIGMOD International Conference on Management of Data* (Santa Barbara, California, USA) *(SIGMOD '01)*. Association for Computing Machinery, New York, NY, USA, 509–520. https://doi.org/10.1145/375663.375731

[6] Matthijs Douze, Alexandr Guzhva, Chengqi Deng, Jeff Johnson, Gergely Szilvasy, Pierre-Emmanuel Mazaré, Maria Lomeli, Lucas Hosseini, and Hervé Jégou. 2024. The Faiss library. (2024). arXiv:2401.08281 [cs.LG]

[7] Ronald Fagin, Laura M. Haas, Mauricio Hernández, Renée J. Miller, Lucian Popa, and Yannis Velegrakis. 2009. *Clio: Schema Mapping Creation and Data Exchange*. Springer Berlin Heidelberg, Berlin, Heidelberg, 198–236. https://doi.org/10.1007/978-3-642-02463-4_12

[8] Madelon Hulsebos, Kevin Hu, Michiel Bakker, Emanuel Zgraggen, Arvind Satyanarayan, Tim Kraska, Çagatay Demiralp, and César Hidalgo. 2019. Sherlock: A Deep Learning Approach to Semantic Data Type Detection. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. ACM.

[9] Lilong Jiang, Michael Mandel, and Arnab Nandi. 2013. GestureQuery: a multitouch database query interface. *Proc. VLDB Endow.* 6, 12 (aug 2013), 1342–1345. https://doi.org/10.14778/2536274.2536311

[10] Sean Kandel, Andreas Paepcke, Joseph Hellerstein, and Jeffrey Heer. 2011. Wrangler: interactive visual specification of data transformation scripts. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '11)*. Association for Computing Machinery, New York, NY, USA, 3363–3372. https://doi.org/10.1145/1978942.1979444

[11] KPAAM-CAM team. 2016–2021. Sociolinguistic interview guides. (2016–2021). Unpublished working documents of the KPAAM-CAM project.

[12] Jayant Madhavan, Philip A. Bernstein, and Erhard Rahm. 2001. Generic Schema Matching with Cupid. In *Proceedings of the 27th International Conference on Very Large Data Bases (VLDB '01)*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 49–58.

[13] Marianthi Markatou, Oliver Kennedy, Michael Brachmann, Raktim Mukhopadhyay, Arpan Dharia, and Andrew H. Talal. 2023. Social determinants of health derived from people with opioid use disorder: Improving data collection, integration and use with cross-domain collaboration and reproducible, data-centric, notebook-style workflows. *Frontiers in Medicine* 10 (2023).

[14] Fatemeh Nargesian. 2019. *Relational Data Enrichment by Discovery and Transformation*. Ph. D. Dissertation. University of Toronto. Copyright - Database copyright ProQuest LLC; ProQuest does not claim copyright in the individual underlying works; Last updated - 2023-06-21.

[15] Fatemeh Nargesian, Erkang Zhu, Ken Q. Pu, and Renée J. Miller. 2018. Table union search on open data. *Proc. VLDB Endow.* 11, 7 (2018), 813–825. https://doi.org/10.14778/3192965.3192973

[16] Quoc Viet Hung Nguyen, Thanh Tam Nguyen, Vinh Tuan Chau, Tri Kurniawan Wijaya, Zoltán Miklós, Karl Aberer, Avigdor Gal, and Matthias Weidlich. 2015. SMART: A tool for analyzing and reconciling schema matching networks. In *2015 IEEE 31st International Conference on Data Engineering*. 1488–1491. https://doi.org/10.1109/ICDE.2015.7113408

[17] Nils Reimers and Iryna Gurevych. 2019. Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics. http://arxiv.org/abs/1908.10084

[18] Diego Rodrigues and Altigran Silva. 2021. A study on machine learning techniques for the schema matching network problem. *Journal of the Brazilian Computer Society* 27 (12 2021). https://doi.org/10.1186/s13173-021-00119-5

[19] William Spoth, Bahareh Sadat Arab, Eric S. Chan, Dieter Gawlick, Adel Ghoneimy, Boris Glavic, Beda Hammerschmidt, Oliver Kennedy, Seokki Lee, Zhen Hua Liu, Xing Niu, and Ying Yang. 2017. Adaptive Schema Databases. In *CIDR*.

[20] William Spoth, Poonam Kumari, Oliver Kennedy, and Fatemeh Nargesian. 2020. Loki: Streamlining Integration and Enrichment. In *HILDA*.