Received 11 September 2023, accepted 1 October 2023, date of publication 11 October 2023, date of current version 18 October 2023.

Digital Object Identifier 10.1109/ACCESS.2023.3323846

RESEARCH ARTICLE

Restoration of Data Structures Using Machine Learning Techniques

BRANISLAVA CVIJETIĆ[1,2] AND ZAHARIJE RADIVOJEVIĆ[2]
[1]Agency for Statistics of Bosnia and Herzegovina, 71000 Sarajevo, Bosnia and Herzegovina
[2]University of Belgrade School of Electrical Engineering, 11020 Belgrade, Serbia

Corresponding author: Branislava Cvijetić (branislava.cvijetic@gmail.com)

This work was supported in part by the Ministry of Science, Technological Development and Innovation of the Republic of Serbia under Contract 451-03-47/2023-01/200103; and in part by the Science Fund of the Republic of Serbia under Contract AVANTES.

ABSTRACT Tabular data is the most common format used to represent real-world information. Almost all programs created for storing or processing data, such as relational database systems, spreadsheets, and statistical analysis software can import or export tabular data. These programs are not sufficiently robust to automatically solve the problems of importing messy delimited files or files that contain data from multiple tables. Additional messy datasets contain data delimited by multiple delimiters without the names of the table columns, and parts of the table rows have substituted or deleted columns. This paper proposes the STCExtract algorithm for reconstructing table structures and data in which the input file can be arranged. The STCExtract algorithm is designed to be domain-independent and modular according to machine learning algorithms and other parameters. The algorithm was developed as a two-phase process, in which the original data tables were recognized in the first phase and the columns of the original data tables in the second phase. The STCExtract algorithm was evaluated through expensive experiments using multiple real datasets. Multiple messy datasets were generated for the four experiments. Three experiments were conducted to determine the optimal parameters for the STCExtract algorithm. A fourth experiment was conducted to evaluate the proposed algorithm. The results show that the STCExtract algorithm correctly arranged the structure of the tables with an accuracy of 94.4% to 100%. The accuracy of the STCExtract algorithm in the second phase (when the data were allocated to columns) ranged from 59.7% to 90.2%.

INDEX TERMS Information extraction, messy datasets, machine learning algorithms, schema discovery.

I. INTRODUCTION

With the advancement of information and communication technologies, the amount of stored and electronically accessible data has significantly increased in recent years. Electronic data can be found in various forms, including unstructured data stored in textual documents, semi-structured XML data, and structured data in relational databases. Various interfaces, such as web browsers, structured query languages, and desktop applications, can be used to access data.

Accordingly, the data can be divided into three categories: unstructured, structured, and semi-structured. Unstructured data do not necessarily have a defined format or follow any rules regarding the data content. Unstructured data included text, video, audio, and photos. In contrast, structured data are organized into semantic entities. Similar entities are grouped in the form of relations and entities in the same group have the same attribute descriptions. The descriptions of all entities in a group comprise a schema. Attributes usually have a predefined format and length and usually follow a predefined order. An example of structured data is the relational data stored in relational databases. The third type of data were semi-structured. Semi-structured data included Email, Hypertext Markup Language (HTML), Electronic data interchange (EDI), and JavaScript Object Notation (JSON). Although they contain structural information, they are not entirely structured. In contrast to structured data, entities of the same group can have different attributes in

The associate editor coordinating the review of this manuscript and approving it for publication was Hongwei Du.

semi-structured data, and the order of the attributes is not particularly important. By contrast, appropriate markers or labels are used to mark the attributes themselves.

A special type of unstructured data is data that is reliably known to have a structure, but has been corrupted for any reason. Such a dataset may contain different versions of the initial structures of one or more entities that have changed over time (e.g., replacement in the order of attributes and number of attributes). It may also contain broken entity structures created when exporting structured data, owing to the use of different delimiters to separate entity attributes and a new row as a messy-delimited file. In addition to the above-mentioned characteristics that describe such a dataset, a problem arises because of the heterogeneity of the data itself. Heterogeneous sets of structured data can contain information about different data types such as dates or time intervals, explicit numeric data (e.g., year), coded numeric data (e.g., 0 or 1 for presence or absence), categorical data (e.g., countries), and text fields (e.g., short description). In addition, with such datasets, a problem is reflected in the fact that values may be missing for some attributes; that is, the value of such attributes can be left blank or filled with some default value. In addition, some of the attributes can be multi-valued, that is, vector types, not just scalar types.

This paper proposes the STCExtract algorithm for extracting original structures and tables from a messy-delimited text file. Given an input messy-delimited file, STCExtract reconstructs the table structure and the data into which the input file can be arranged. Furthermore, the STCExtract algorithm is designed to be domain-independent, modular according to machine learning algorithms, and modular according to other parameters, such as a function for calculating the distance between structures.

Machine learning (ML) is a part of artificial intelligence (AI) that allows software applications to predict outcomes accurately without being explicitly programmed. Machine learning algorithms use historical data as inputs to predict new output values, and can be trained in several ways. Based on the learning methods, machine learning algorithms can be broadly categorized as supervised or unsupervised. Supervised learning algorithms use labeled data, meaning the data have a known outcome or a target variable. Unsupervised learning algorithms use unlabeled data, meaning the data does not have a known outcome or a target variable. Many machine learning algorithms exist in the literature; however, this paper uses two machine learning algorithms, k-means and hierarchical, for the reconstruction table structure, and five machine learning algorithms (K-means, Mini-batch K-means, Birch, Gaussian mixture, and Fuzzy K-means (FCM)) for the reconstruction table columns.

The remainder of this paper is organized as follows. Section II presents the related work. In Section III, the problem formulation and the STCExtract algorithm are described. Section IV presents the experimental setup and results of the experimental analysis. Finally, Section V concludes the paper.

II. RELATED WORK

This section presents the relevant background for classifying and comparing the schemes for scheme discovery and methods. The analysis carried out in this section is primarily based on review papers [1], [2], [3], [4] in which state-of-the-art schema information extraction approaches are presented. These studies analyzed the influences of the approach to scheme discovery, techniques for scheme discovery, characterization of discovery procedures, input and output data formats, and quality of the results.

A. APPROACHES TO SCHEMA DISCOVERY

Approaches to schema discovery consist of several steps, in which schema-related information is extracted from the original input set of schema information. This information can be extracted from RDF, OEM, or other data types. Scheme discovery approaches can be grouped into three categories based on the basic algorithmic concepts behind the scheme discovery method. These categories include implicit schema discovery, explicit enrichment of existing schemas, and the discovery of structural patterns within existing schemas.

Implicit schema discovery approaches attempt to discover a schema from a data source without requiring additional schema information. This procedure aims to extract new information from a dataset by analyzing the entities and connections between the elements. These approaches begin the analysis with an empty schema and attempt to characterize this dataset in two ways: grouping similar instances and similar paths [5], [6], [7], [8], [9].

Approaches that use explicit enrichment of existing schemas begin with existing schemas and statements on the schema that can complement or enrich existing schemas. Their goal was to generate new types using existing types, or to generate other schemas. These approaches can be classified into two categories. The first approach is based on machine learning techniques, whereas the other approaches are based on statistical data analysis. These techniques primarily use the connectivity between instances [10], [11], [12].

The procedures for discovering structural patterns within existing schemas begin with identifying all possible structural patterns (versions) of entities in a given dataset of a known schema. This procedure aims to analyze the co-occurrence relationship between the properties in the dataset rather than to discover types and additional schemas, as in the explicit enrichment of existing schemas. A different set of properties describes each instance within the dataset. The set of properties that describes an instance represents its structural pattern. These approaches can be classified into two categories based on the scope of the pattern recognition. The first is based on exact pattern recognition. In contrast, the second approach is based on approximate pattern recognition. Papers describing exact pattern recognition procedures are provided in [13], [14], [15], [16], [17], [18], and [19], whereas those describing approximate pattern-discovery approaches are provided in [20], [21], and [22].

B. TECHNIQUE FOR SCHEMA DISCOVERY

Techniques that enable the discovery of the required elements of a scheme are also applied as steps to their discovery. The techniques used to identify patterns can be divided into three groups. The first group consists of techniques based on machine learning, the second on formal methods, and the third on statistical methods. Machine learning-based techniques use machine learning concepts to discover the dependencies between data in a given dataset. These techniques can be divided according to several criteria: the type of learning (unsupervised and supervised machine learning), classification algorithm (e.g., K-NN), clustering algorithm (e.g., K-means and Hierarchical clustering), and clustering algorithm rules (Frequent Pattern Mining algorithms (Apriori)). Formal methods are based on mathematical and formal logic to determine the dependencies between the data in a given dataset. These techniques include Formal Concept Analysis, which is used in pattern discovery to discover classes by generating a network of concepts, and bi-simulation, which is used to cluster similar paths in data graphs. Techniques based on statistical methods attempt to generate and quantitatively describe datasets. In schema discovery, this technique does not focus on determining the frequency of properties such as the distribution of properties across types.

C. CHARACTERIZATION, FORMATS, AND QUALITY

In the case of techniques for discovering the required scheme elements, their characterization is also observed according to their behavior when working with different data. Characterization can be divided into several groups of properties: the first group consists of characteristics related to the extensibility of the solution when working with an enormous amount of data; the second group is determined by the characteristics of whether the solution is reached simultaneously or incrementally; the third group consists of solutions that consistently provide the same set of input data in the same output solution, and the fourth group consists of solutions that process data locally or remotely.

When using techniques and algorithms to detect scheme elements and observe the input data that can be delivered to the algorithms and output data, the results that the given algorithms can produce are necessary. The input data can be divided into two categories, depending on their purpose. The first category comprises datasets and the second comprises user-defined parameters. RDF data graphs from the OEM gather data types that describe input datasets. Some approaches require the definition, even partially, of the ontology for the dataset, whereas others use only its instances. The second category consists of additional data that can be user-defined parameters required by basic algorithms, such as the similarity thresholds used to compare graph elements or the desired number of clusters. Depending on the chosen approach and technique, output data can include several elements. The output may include information related to data types and subtypes, semantic relationships between data, hierarchical relationships between data, data access plans, patterns between data, and other information.

As one of the crucial elements when analyzing the results of the use of techniques and algorithms for the detection of scheme elements, it is necessary to observe the quality of the results. The quality of the resulting schema expresses the extent to which it adequately describes an input dataset. Multiple aspects of the output schema quality can be observed, including schema relevance, schema completeness, and class accuracy. The relevance aspect involves recognizing only the elements, links, or other schema-related information provided by the approach considered in the schema. The schema completeness aspect includes the comprehensiveness of the generated schema; that is, checking whether all elements should exist in the actual schema. The precision aspect involves checking the correctness of the generated type declarations between entities and corresponding instances.

III. PROBLEM STATEMENT AND STCEXTRACT ALGORITHM OVERVIEW

Existing schema discovery algorithms predominantly focus on structured and semi-structured data, the initial structure of which is known, and attempt to discover additional features. These algorithms may not be intended for and may have difficulty with unstructured data containing multiple tables that are reliably known to have a corrupted structure. Existing schema discovery algorithms cannot be directly applied to such data, and require modifications. Therefore, these solutions were not compared to the STCExtract algorithm proposed in this study. This section presents the problem statement and details of the STCExtract algorithm.

A. TERMINOLOGY AND PROBLEM STATEMENT

A messy delimited file can be considered as a set of n rows $\{\mathrm{r}_1, \mathrm{r}_2, \ldots, \mathrm{r}_\mathrm{n}\}$, with rows r_i consisting of m fields $\{\mathrm{f}_{\mathrm{i}1}, \mathrm{f}_{\mathrm{i}2}, \ldots, \mathrm{f}_{\mathrm{im}}\}$. Each row has a different number of fields. The database from which the messy file originates comprises a set of k tables $\{\mathrm{t}_1, \mathrm{t}_2, \ldots, \mathrm{t}_\mathrm{k}\}$, where the i_{th} table t_i consists of l columns $\{\mathrm{c}_{\mathrm{i}1}, \mathrm{c}_{\mathrm{i}2}, \ldots, \mathrm{c}_{\mathrm{il}}\}$. Each table had a different number of columns. Each row r belongs to exactly one table t, and one or more fields f of a given row belong to exactly one column c of a given table.

During the formation of a messy file, consecutive rows can originate from two or more tables; some fields can be missed, field orders can be changed, column headers for the original table can be missed, and fields can contain symbols that can be used as separators. Field types may be scalar (e.g., number, string, date) or vectors of scalar fields. This enables the separation of a row into tokens, instead of separating a row into fields. For example, using different delimiters, row r_i can be split into b tokens $\{t_{i1}, t_{i2}, \ldots, t_{ib}\}$. The main goal of this paper was to extract a set of tables and their columns corresponding to the structures of the original tables and then populate the tables and columns using messy file content. The goal involves recognizing the tables using row and token values, determining the number of columns for each table, and (if necessary) determining the alignment columns (merging the columns again or adding new columns).

B. RESEARCH QUESTIONS

The research questions addressed in this paper were as follows:

- RQ1: Is it possible to propose an algorithm based on machine learning techniques that reliably restores the data structure known to have a structure that has been corrupted for any reason?
- RQ2: Can the proposed algorithm be extended by using rules and functions to detect scalar data types of columns within a structure?
- RQ3: Can the proposed algorithm be extended using rules and functions to detect the vector data types of columns within a structure?

C. STCEXTRACT ALGORITHM

This section explains the sequence of operations of the STCExtract algorithm illustrated in Figure 1.

In general, the STCExtract algorithm is based on cluster analysis. Cluster analysis or clustering involves grouping a set of objects such that objects in the same group (called a cluster) are more similar (in some sense) to each other than to those in other groups (clusters). Cluster analysis is not a specific machine learning algorithm but a general task to be solved. In the STCExtract algorithm, clustering occurred twice. In the first clustering phase, the structure and order of the tokens in each input file row are used as a set of objects. Clustering aims to discover clusters/tables and group rows into the discovered clusters. In the second clustering phase, the token values, token structure, and order in the rows belonging to the same cluster/table are used as the object set. The goal of clustering is to place tokens in the clusters/columns of a table, and group tokens belonging to the same field. The STCExtract technique is executed as a sequence of actions in an input messy-delimited file.

1) SPLIT ALL ROWS INTO TOKENS

In this step, all rows from a messy delimited file are read row-by-row and split into tokens. Splitting a row into tokens is a dialect detection problem. The dialect detection problem was solved using the hypothesis that multiple delimiters were used. Each row in the delimited file is separated by several delimiters (tabs, commas, semicolons, colons, and vertical bars). Delimiters can be added as parameters in the STCEtract algorithm. Splitting occurred across all the rows. When splitting the content of a row, the portion of each row and its position within the rows were saved. This step results in a table containing all rows of the input file and all row items (tokens) in a format (ID line, token value, and token position). At this stage, the source file is separated from known separators to generate a file for external validation.

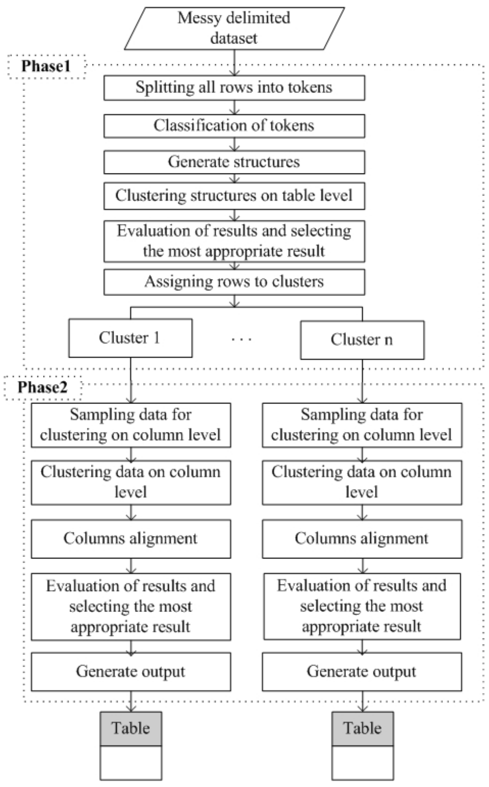

Messy delimited dataset
Phase1
Splitting all rows into tokens
Classification of tokens
Generate structures
Clustering structures on table level
Evaluation of results and selecting the most appropriate result
Assigning rows to clusters
Cluster 1
. . .
Cluster n
Phase2
Sampling data for clustering on column level
Clustering data on column level
Columns alignment
Evaluation of results and selecting the most appropriate result
Generate output
Sampling data for clustering on column level
Clustering data on column level
Columns alignment
Evaluation of results and selecting the most appropriate result
Generate output
Table
Table

FIGURE 1. A sequence of action in the STCExtract algorithm.

2) CLASSIFICATION OF TOKENS

Each extracted token is classified into one of the appropriate categories in this step. The classification of a token can be achieved based on its value, using a different set of rules and techniques. The initial rules are based on regular expressions that classify tokens into three categories: number, string, and date. Tokens in which the value is an empty string or the null value is string coded. In the STCExtract algorithm, additional rules for token categories can be added as parameters. Dictionary-based techniques can be used in addition to regular expression techniques. In this technique, a table contains a set of key-value pairs, where a key represents a token or cleaned token, and a value represents a token category. Initially, this key-value table was empty. The output from this step is a table containing all the tokens in a given format (line ID, token value, token position, cleaned value, and token category).

3) GENERATE STRUCTURES

This step created a set of objects for the first clustering phase. An object corresponds to one row in a message file containing an ordered vector of categories corresponding to tokens in a given row. Subsequently, in every row, information is attached to an additional link in the structure to which it belongs. A generated string value is added to the token category (number, string, and date) during this phase to preserve token order. Therefore, it is possible to distinguish between strings in the first and third positions. Once all the rows are linked to the structures, they are aggregated according to their values for future processing. This step results in a table containing all the objects with the same structure in a given format (object structure ID, number of appearances, number of tokens, structure of a clustering object, and description).

4) CLUSTERING STRUCTURES ON TABLE LEVEL

In this step, the first clustering phase is performed. This phase aims to discover clusters/tables based on the structure of tokens and their order in each row of the input file, the frequency of their appearance, and the similarity between them. Each discovered cluster is a table.

Clustering is a task for which many algorithms have been proposed. However, no clustering technique is universally applicable and different techniques support different clustering objectives. Therefore, an understanding of the clustering problem and technique is required in order to apply a suitable method to a given problem. This paper used hard clustering in which every data point (structure) belonged to a single cluster.

Two customized algorithms were tested during this phase: hierarchical clustering and K-means clustering. For each algorithm, six different distance functions were used (linear, linear_weight, log, log_weight, reverse and reverse_weight); also, five different cluster center detectors algorithms were used (MinDistance – cluster center is an object with smallest distance to all other objects in cluster; MaxFreq – cluster center is object with highest number of appearance in cluster; PositionMax – cluster center is an generated object that has on each position a type that appears the most on that pasition in a cluster; GroupMax – cluster center is a generated object that has on each position a type that appears the most on that position in a cluster ignoring appearance of the same consecutive tokens; GroupMax2 – cluster center is a generated object that has on each position a type that appears the most in a cluster ignoring appearance of the same consecutive tokens), and three object comparator were used (Matcher – combines length similarity score and position similarity score in equal manner, Matcher1 – combines length of sequences of equal type similarity score and position of sequences of equal type similarity score in equal manner, Matcher2 – combines length similarity score, position similarity score in an equal manner, length of sequences of equal type similarity score, and position of sequences of equal type similarity score in an equal manner). The results of this step are the sets of detected clusters/tables.

5) EVALUATION OF RESULTS AND SELECTING THE MOST APPROPRIATE RESULT – PHASE1

Clustering validation has long been recognized as a vital issue for the success of clustering applications. There are two methods for evaluating the cluster quality. One is an external evaluation in which the truth labels in the datasets are known beforehand. Another method is an internal evaluation performed without labels or with the dataset itself. The proposed algorithm was evaluated using external validation.

6) ASSIGNING ROWS TO CLUSTERS

In this step, each row was assigned to one detected table based on the relationship between the rows and comparison objects. The comparison objects were then assigned to the clusters in the detected table. Each row is assigned to a specific cluster for processing, based on the associated table. In subsequent steps, row tags are assigned to the corresponding columns of a detected table.

7) SAMPLING DATA FOR CLUSTERING ON COLUMN LEVEL – OPTIONAL

Data sampling was performed during this step. This step aims to reduce the computational time of tag clustering, resulting in a subset of rows and their tags being used in the phase-clustering columns. It is essential to note that all rows were clustered based on this sampling, not just the sampled rows. This step is optional.

Sampling was performed using all rows to be structured, from which only p percent (20% of the data) was randomly selected. A row containing a structure with only one appearance is selected as the sample. In this step, the token content was changed. If a raw file continues the same token at multiple positions, a new token value is created using a combination of initial value, position, and class type. If the token values appear more than twice at the row level, then the token value is changed by comparing the initial value, position, and class type.

8) CLUSTERING DATA ON COLUMN LEVEL

In this step, a second clustering phase was performed. This phase aims to discover columns based on the value of tokens and their order in each row of the input file, frequency of their appearance, and similarity between them. Each discovered cluster represents a column.

During this phase, five standard clustering algorithms were tested: K-means, Mini-batch K-means, Birch, Gaussian mixture, and Fuzzy K-means (FCM). A structure with the most rows belonging to the cluster is selected to determine the optimal number of clusters or parameter k. The number of elements in this structure determines the parameter k. If the two structures have the same highest number of rows, a structure with fewer columns is selected. It was assumed that most data within a cluster were obtained from the same original table.

The results of this step are the sets of detected clusters/columns. Each set corresponded to one of the selected clustering algorithms.

9) COLUMN ALIGNMENT

In this step, multiple tokens were combined and aligned into a single column. Column alignment was performed in two phases:

- In the first phase, adjacent columns belong to the same cluster and the values attributed by the clustering algorithm are merged. Here, the token value type is not

considered. This is because some tokens are recognized as numbers, but are part of the messy text.
- In the second phase, redundant columns are still attached to the last column and the value assigned by the algorithm to the previous position is maintained.

An alignment process was performed for each clustering algorithm. The results of this step are sets of aligned tokens in the clusters/columns.

10) EVALUATION OF RESULTS AND SELECTING THE MOST APPROPRIATE RESULT – PHASE2

In this step, a set of detected clusters/columns is selected. Each set contains clusters. Each cluster corresponded to an extracted column in the selected table. One set of results was selected for external validation at the table and column level. This step resulted in the detection of the selected tables and the population of the columns.

11) GENERATE OUTPUT

In this step, the datasets produced in the previous phases are exported for use in various software environments.

D. POSITIONING

Based on previous research, this algorithm can be classified as an implicit schema discovery algorithm, because no additional information is available in the input messy file. The STCExtract algorithm attempts to characterize a dataset by grouping similar instances. This algorithm uses an unsupervised machine learning technique and enables the use of multiple clustering algorithms. The scalability of the STCExtract algorithm relies on the selected algorithms and parameters. The solution was reached simultaneously locally, and the results were stable. Input data are messy text files with unknown data types but with predefined patterns for detecting primitive types. The output of the algorithm represents a structure in the form of tables and columns, corresponding to the original dataset. The quality of a solution can be measured by calculating its accuracy, precision, recall, or adequate substitutes.

IV. EXPERIMENTAL SETUP

This section describes the data sources used in our experiments. The problem addressed in the three experiments formally declared a procedure for creating and selecting a final messy-delimited test file. Four experiments were conducted, as illustrated in Figure 2.

The overall experiment was set up such that Experiments 1, 2, and 3 were used to determine the input parameters for the STCExtract algorithm. Each experiment investigated problems defined by the research questions addressed in the previous section. In Experiment 1, the algorithm results were tested using datasets with tables containing only columns with a scalar data type. In Experiments 2 and 3, the algorithm's results were tested using datasets with tables containing columns with scalar data types and one or more

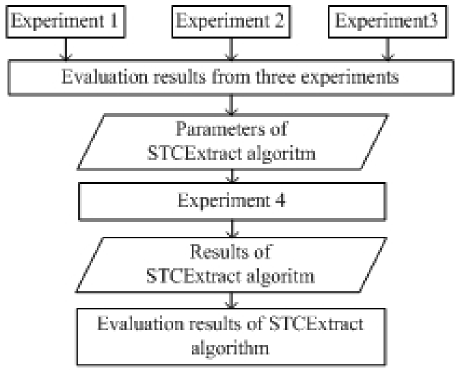

Experiment 1
Experiment 2
Experiment3
Evaluation results from three experiments
Parameters of STCExtract algoritm
Experiment 4
Results of STCExtract algoritm
Evaluation results of STCExtract algorithm

FIGURE 2. Experimental setup.

columns with vector data types, respectively. The evaluation results of these three experiments determined the parameters of the STCExtract algorithm that were used as the input parameters for Experiment 4.

The evaluation results of these three experiments determined the parameters of the STCExtract algorithm that were used as the input parameters for Experiment 4. The evaluation of the results of Experiment 4 represents the final results of this paper and was performed according to the methodology described in [23].

A. DATA

Data were collected from two publicly available sources: the Internet Movie Database and Kaggle. Data sources are described in the following sections. Short information regarding the used is provided in Table 1 and more detailed information is provided in Appendix.

1) IMDB DATABASE

The Internet Movie Database (IMDb) contains information and statistics on films, television programs, actors, directors, and other professionals in the film industry. The official IMDb database contained seven datasets. Each dataset is in a tab-separated compressed file (TSV) in UTF-8 format and updated daily. In addition, IMDb datasets are available for customer access for personal and noncommercial purposes. Four datasets were used in this database [24].

2) THE GAME REVIEWS

The Game Review Dataset is a structured dataset comprising four files with information on game reviews. Only the review_info.csv file was used in the experiments. One column is removed from each file [25].

3) THE CALENDAR

The Calendar database is a catalog of over three million records with limited information about data sources. This data source was used because it contained data fields in the dataset [26].

TABLE 1. Information regarding datasets.

Data sources	# datasets	#columns	#columns with scalar data types	#columns with vector data types	#empty values in columns	#columns with unique values
IMDb database	title.basic	9	8	1	32	1
	title.principals	6	6	0	0	0
	title.episode	4	4	0	0	1
	name.basics	6	4	2	2505480	1
The Game Reviews	review_info	9	9	0	0	1
Calendar	calendar	7	7	0	1856	0
Barcelona Traffic Accidents	accidents_opendata	17	17	0	20136	0
Monthly electricity production	data	12	12	0	17105	0
Port Shipment Dataset	shipping_data	8	8	0	2861	0
Large Movie Dataset	movies_dataset	5	4	1	0	0
Every Pub in England	open_pubs	9	9	1	2	0
IIT Admissions Dataset	student_data	10	10	0	0	0

4) BARCELONA TRAFFIC ACCIDENTS
The dataset was created by reconciling yearly traffic accident reports in Barcelona recorded by Guardia Urbana and provided by OpenDataBCN (Public Barcelona government data portal). The released dataset contains 27 columns. Seventeen columns were used in the experiments [27].

5) MONTHLY ELECTRICITY PRODUCTION
The data were collected from the International Energy Agency (IEA) website and included monthly information on energy production in various countries from 2010 to 2022. Energy production is measured in gigawatt-hours (GWh) and covers a range of energy products including hydro, wind, solar, geothermal, nuclear, and fossil fuels [28].

6) PORT SHIPMENT DATASET
This dataset contains information on the international shipments of ports. The dataset includes the name of the product, its price, weight, length, width, and height as well as the date on which it was shipped and the destination port. This dataset was not based on actual shipments or real-world data [29].

7) LARGE MOVIE DATASET
This dataset contains over 200k movies watched by 7k+ unique users, ratings, and genres. The dataset includes the ID of a user, name of the movie watched, rating given by the user, and movie genre [30].

8) EVERY PUB IN ENGLAND
This dataset includes information on 51,566 pubs. This dataset contains the following columns: the Food Standard Agency's ID for this pub, name of the pub, address fields separated by commas, postcode of the pub, easting, northing, latitude, longitude, and local authority this pub falls under. The data were derived from the Food Standard Agency's Food Hygiene Ratings and ONS Postcode Directory. The data were licensed under an open government license [31].

9) INSTITUTES OF TECHNOLOGY ADMISSIONS DATASET
This dataset contained 200,000 student profiles from the Indian Institute of Technology (IIT). The dataset provides valuable information on student admissions, academic background, and financial aspects. It includes fields such as student ID, date of birth, field of study, specialization, year of admission, expected year of graduation, current semester, fees, and discounts on fees [32].

B. EXPERIMENTS
1) EXPERIMENT 1
The objective of Experiment 1 was to determine the combination of parameters and applied algorithms that performed best in the clustering of structures, if the input data contained only columns with scalar data types.

Structures from various tables, with all columns containing only scalar data types, were clustered in experiment 1. There are three test datasets where the first dataset was created based on two tables (exp1_case1), the second based on three tables (exp1_case2), and the third based on five tables (exp1_case3) from the input dataset. (Table 2)

2) EXPERIMENT 2
The objective of Experiment 2 was to determine the combination of parameters and the applied algorithms that best performed structure clustering when the input data contained columns of scalar and vector data.

The clustering of structures belonging to various tables was performed in Experiment 2, with each table containing columns of scalar data type and one table containing one column of vector data type, in addition to scalar data type. Four test datasets are used in this paper. The first dataset was created based on two tables (exp2_case0), the second on three tables (exp2_case1), the third on four tables (exp2_case2), and the fourth on six tables (exp2_case3) of the input dataset. The datasets for Experiment 2 were generated by using the tables used in Experiment 1. Experiment 2 used an additional table with one column and vector data type. (Table 2)

3) EXPERIMENT 3
The objective of Experiment 3 was to determine whether the combination of parameters and applied algorithms performed best in the clustering of structures, if the input data contained columns with scalar and multiple vector data types.

The clustering of structures belonging to various tables was performed in Experiment 3, with each table containing columns of scalar data types and one table containing two columns of vector data types, and scalar and vector data types. There were four test datasets. The first dataset was created based on two tables (exp3_case0), the second was based on three tables (exp3_case1), the third was based on four tables (exp3_case2), and the fourth dataset consisted of six tables (exp3_case3) from the input dataset. The datasets for Experiment 3 were created using the tables from Experiment 1. Experiment 3 used an additional table with two columns and vector data type. (Table 2)

TABLE 2. **Details of the test datasets, including the number of converted structures used for Experiments 1, 2, and 3.**

datasets	Experiments EXP1 CASE1	CASE2	CASE3	EXP2 CASE0	CASE1	CASE2	CASE3	EXP3 CASE0	CASE1	CASE2	CASE3
title.episode	4	4	4	4	4	4	4	4	4	4	4
title.principals	9	9	10		10	9	11		10	9	11
review_info		3	3			3	3			3	3
accidents_opendata			30				34				31
calendar			6				6				6
title.basics				153	164	157	143				
name.basics								27	28	27	27

4) EXPERIMENT 4

The objective of Experiment 4 was to re-evaluate clustering using the STCExtract algorithm with selected combinations of input parameters for Phase 1 for clustering structures, and to use clustering algorithms selected for Phase 2 for clustering table columns. Experiment 4 used five new datasets, and one dataset from Experiment 2.

In Experiment 4, the clustering of structures belonging to different tables was performed, where the tables had only columns with scalar data types (case1s, case1n, and case2s). Structure clustering was also performed for tables with only columns with a scalar data type, tables with columns with a scalar data type, and columns with a vector data type (case1v and case2v). Five test datasets were used: the first dataset was created based on two tables (exp4_case1s); the second was based on three tables (exp4_case2s); the third was based on three tables (exp4_case1n); the fourth was based on three tables (exp4_case1v), and the fifth dataset consisted of four tables (exp4_case2v) from the input dataset. (Table 3)

TABLE 3. **Details of the test dataset, including the number of converted structures used for Experiment 4.**

datasets	EXP4 CASE1s	CASE1v	CASE1n	CASE2s	CASE2v
data	27	26	28	29	24
shipping_data	5	7	6	6	6
movies_dataset		30			
open_pubs			49		
student_data				3	3
title.basic					85

C. CREATION OF MESSY DELIMITED FILE

The test messy-datasets required for the experiments are not publicly available. Therefore, messy datasets for the experiments were created based on the publicly available tabular datasets. Depending on the experiment, two or more datasets (Table 1) were selected as listed in Table 2 or Table 3.

During the selection process, column names were eliminated, column positions were changed in randomly selected rows, and the fields were randomly deleted. For the first three experiments, 100 messy datasets were created for 11 cases, that is, 1100 messy datasets. The same number of rows represent each table from the original dataset to create a balanced dataset. The rule that the converted structures do not share rows between the input datasets was used as an additional criterion for selecting the final test sample.

D. DESIGN OF EXPERIMENTS 1, 2 AND 3

Each Design of experiments (DOE) comprises a series of steps: planning, execution of the experiment, and analysis of the collected experimental data using various statistical methods to draw valid and objective conclusions [33]. An experiment can be defined as a series of tests in which a set of input variables or factors (x) is changed by an experimenter in a controlled manner (c) to observe and identify how the response (y) of the system is affected by these changes [34]. The experiments were performed systematically using factorial experiments (so-called factorial experimental designs/arrays), wherein several factors were altered during each experimental run to examine their impact of several factors and interactions on the response quantity [33]. A factorial experiment whose design consists of all possible combinations of chosen factors and levels is called full-factorial design (FFD) [35].

Each experiment consisted of two phases. The goal of Phase 1 is to obtain clusters corresponding to the data tables from the input dataset. Steps 1–6 of the STCExtract algorithm create a cluster at table level. The goal of Phase 2 is to obtain clusters corresponding to the columns of the corresponding tables based on the clusters created after Phase 1. Steps 7–11 of the STCExtract algorithm are used for clustering at the column level.

In Phase 1, six distance functions, five cluster center detectors, three object comparators, and two algorithms for clustering structures were tested. The distance functions, object comparators, cluster center detectors, and algorithms for clustering structures represent the input parameters of the proposed STCExtract algorithm or factors with different levels. In Phase 1, the number of levels for each factor was not the same; thus, these experiments were symmetrical or mixed-factorial, and the number of unique factor combinations was $\#Runs = 6^1 x 5^1 x 3^1 x 2^1 = 180$ (in this paper, it will be called ResultsPh1). For 11 cases in three experiments, the ResultsPh1 was generated. In total, 1980 ResultsPh1 were obtained.

In Phase 2 of the STCExtract algorithm for column clustering, five machine learning algorithms were tested: K-means, FCM (Fuzzy C-means), Gaussian Mixture, Birch (balanced iterative reducing and clustering with hierarchies), and Mini

Batch K-means The proposed STCExtract algorithm includes these machine learning algorithms for clustering columns as an additional parameter.

Combining all the mentioned parameters from Phase 1 and Phase 2 of the STCExtract algorithm, $6^1x5^1x3^1x2^1x5^1 = 900$ combinations for clustering the input dataset for each case were tested. A total of 9900 column-level clustering results (hereafter referred to as ResultsPh2) were obtained.

E. TOOLS

The Python 3.10 programming language in PyCharm 2021.3.1 (Community Edition) was used to implement the STCExtract algorithm. For data analysis, Python scripts were created in the Jupyter Notebook 6.5.4. MS SQL Server 2008R2 was used for data analyses. Testing was performed on three personal computers as follows:

- 11th Gen Intel® Core™ i5-1135G7 Processor with 8 GB RAM.
- Intel® Core™i7-6700 Processor with 16 GB RAM.
- Intel® Core™i9-9900 Processor with 64.0 GB RAM.

F. EVALUATION METRICS

After the formation of the cluster, that is, the creation of the generated output tables, the obtained results were compared with the original data at the level of rows belonging to the corresponding tables. Based on this comparison, external measures, such as Adjusted Mutual Information (AMI) and Adjusted Rand index (ARI), were calculated for each ResultsPh1.

To determine the number of correctly recognized rows that belong to tables that have all columns with a scalar data type, as well as the number of correctly recognized rows that belong to tables that, in addition to columns with a scalar data type, and columns with a vector data type, the following values were calculated:

$$\text{accuracy}_{\text{ph1}} = x_{ph1}/y_{ph1} \tag{1}$$

$$\text{accuracy}_{\text{ph1\_scalar}} = x_{ph1\_scalar}/y_{ph1\_scalar} \tag{2}$$

$$\text{accuracy}_{\text{ph1\_vector}} = x_{ph1\_vector}/y_{ph1\_vector} \tag{3}$$

In Equation (1), x_{ph1} is the total number of correctly recognized rows for each input table and y_{ph1} is the sum of the rows from all input tables. Assuming that each cluster contained data from various input tables, the number of correctly recognized rows, x_{ph1} was obtained from the table with the maximum number of correctly recognized rows in the given cluster.

In Equation (2), x_{ph1_scalar} is the number of correctly recognized rows for each input table where all columns have a scalar data type, and y_{ph1_scalar} is the number of correctly recognized rows for each input table where all columns have a scalar data type.

In Equation (3), x_{ph1_vector} is the number of correctly recognized rows for each input table with a column/columns containing vector data type. y_{ph1_vector} is the total number of rows of all the input tables with columns of a vector data type.

After the formation of clusters at the column level, that is, the generation of the generated output tables, the obtained ResultsPh2 were compared with the original data at the row and column levels belonging to the corresponding tables, assuming that there are data from several input tables in one cluster, as the number of correctly recognized rows per table and column is taken from the table with the maximum number of correctly recognized rows in the given cluster.

To determine the number of correctly recognized rows on the rows and columns that belong to tables that have all columns with a scalar data type, as well as the number of correctly recognized rows that belong to tables that, in addition to columns with a scalar data type and columns with a vector data type, the following values were calculated:

$$\text{accuracy}_{\text{ph2}} = x_{ph2}/y_{ph2} \tag{4}$$

$$\text{accuracy}_{\text{ph2\_scalar}} = x_{ph2\_scalar}/y_{ph2\_scalar} \tag{5}$$

$$\text{accuracy}_{\text{ph2\_vector}} = x_{ph2\_vector}/y_{ph2\_vector} \tag{6}$$

In Equation (4), x_{ph2} is the total number of correctly recognized rows for each input table and all its columns and y_{ph2} is the sum of the rows from all input tables. Suppose there are data from several input tables in one cluster. In this case, the number of correctly recognized rows and columns, x_{ph2} was obtained from the table with the maximum number of correctly recognized rows with all columns in the given cluster.

In Equation (5), x_{ph2_scalar} is the number of correctly recognized rows for each input table and its columns, where all columns contain scalar data types, and y_{ph2_scalar} is the total number of rows for all input tables, where all columns contain scalar data types.

In Equation (6), x_{ph2_vector} is the number of correctly recognized rows and columns for each input table with column/columns containing the vector data type, and y_{ph2_vector} is the total number of rows for all input tables with columns containing the vector data type.

G. EVALUATION PHASE 1 – CLUSTERING STRUCTURES

After Phase 1 of the STCExtract algorithm, three analyses were performed.

- The Adjusted Rand index (ARI) metric results for each experiment and case were analyzed independently for each parameter of the proposed STCExtract algorithm to determine whether any parameter significantly affected ResultsPh1 and how they interacted.
- The number of recognized clustered structures (k: number of clusters) and the structure of the original dataset (t: number of tables) were analyzed independently.
- Regardless of the experiment, the best results for the Adjusted Mutual Information (AMI) and Adjusted Rand index (ARI) metrics for 1980 ResultsPh1 were compared.

In the first two analyses, each input parameter was analyzed independently to determine whether any parameter significantly affected ResultsPh1 at the case level, and how they interacted with each other at the experimental level. For each ResultsPh1, a consolidated measure was calculated. This measure was obtained by comparing the ARI values of ResultsPh1 among the experimental cases and choosing the minimum ARI value. Descriptive statistics (measures of central tendency and variability) were used to evaluate ResultsPh1 for ARI values of ResultsPh1. In addition, a one-way analysis of variance (ANOVA) test, with the significance set at 0.05, was used to compare the means of the input parameters for each experiment. ANOVA is a statistical relevance tool designed to evaluate whether a null hypothesis can be rejected while testing hypotheses. This determines whether the means of three or more groups are equal. The null hypothesis was that any difference between the comparator functions of the three objects would be due to chance. An ANOVA test typically uses a statistic called the 'p-value.' The null hypothesis was rejected if the 'p-value' was less than the significance level. A more detailed explanation has been provided in [35]. The Tukey's Honest Significant Difference (Tukey's HSD) test was used to determine which groups differed.

In addition, at the case level, the number of clusters k obtained after Phase 1 and the number of tables t used when creating a test dataset with a broken structure were calculated for each ResultsPh1. Based on these two parameters for each case and each experiment, the number of ResultsPh1, where the value of k is equal to or greater than or less than the value of t, was calculated. Suppose k<t; this indicates the merging of several data tables within one cluster, which can lead to the degradation of the solution's performance. In this case, the unstructured input data must be clustered correctly; there is not one table in one cluster, but many different tables. If k>t, all tables may be recognized correctly, but divided into several parts, and a certain number of table rows are incorrectly classified. Suppose the value of k is equal to t. In this case, all input tables may be recognized correctly; however, inside one of the k clusters, the rows may be incorrectly clustered and may belong to another table.

1) ANALYSIS – OBJECT COMPARATOR

The results for the three object comparators (ElementMatcher, ElementMatcher1, and ElementMatcher2) in Experiments 1, 2, and 3 are shown in Figures 3, 4, and 5, respectively. The results based on ARI values from ResultsPh1 are shown in Figures 3a, 4a, and 5a. Figures 3b, 4b, and 5b compare the number of identified clusters k with the number of input tables t. The circles in Figures 3a, 4a, and 5a represent mean values.

Table 4 shows the mean values of the three object comparators according to the descriptive statistics for each case in experiment 1.

This step tested the influence of the three object comparator functions on the ARI values of 180 ResultsPh1. ResultsPh1 was divided into three groups by matcher_type, with 60 results each. The test was repeated for each case in Experiment 1 and a consolidated measure. The null hypothesis was

TABLE 4. Experiment 1 - Mean ARI value for object comparator.

matcher type	exp1_case1	exp1_case2	exp1_case3	exp1_min
ElementMatcher	0.657	0.741	0.564	0.358
ElementMatcher1	0.750	0.678	0.618	0.527
ElementMatcher2	0.639	0.652	0.546	0.428

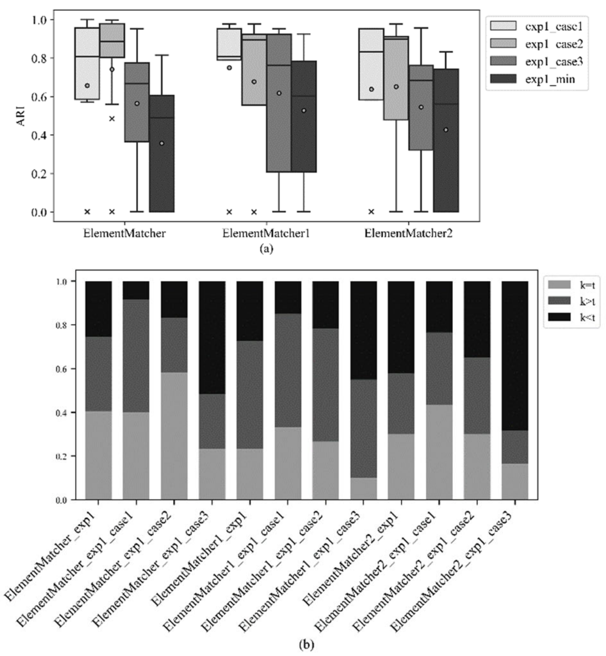


FIGURE 3. Experiment 1 - Object comparators: (a) ResultsPh1 based on ARI values; (b) results based on number recognized cluster –k, and number input table – t.

rejected for the consolidated measure. Tukey's HSD test for consolidated measures showed a p-value of 0.012 between ElementMatcher and ElementMatcher1.

In Experiment 1, ElementMatcher yielded the highest percentage of results (74.4%) when the number of recognized clusters was greater than or equal to the number of input tables. ElementMatcher1 comes second at 72.8% and ElementMatcher2 comes third at 57.8%.

The ElementMatcher and ElementMatcher1 functions had a more significant impact on ResultsPh1 than ElementMatcher2 according to the analysis based on the comparison of the number of clusters and tables, ANOVA, and Tukey's HSD test for a consolidated measure of Experiment 1.

Table 5 shows the mean values of the three object comparators according to the descriptive statistics for each case in experiment 2.

TABLE 5. Experiment 2 - Mean ARI value for object comparator.

matcher_type	exp2_case0	exp2_case1	exp2_case2	exp2_case3	exp2_min
ElementMatcher	0.732	0.695	0.678	0.480	0.355
ElementMatcher1	0.638	0.440	0.578	0.474	0.302
ElementMatcher2	0.468	0.334	0.544	0.382	0.203

As in Experiment 1, an ANOVA test was performed for all the cases in Experiment 2. The results showed that the null

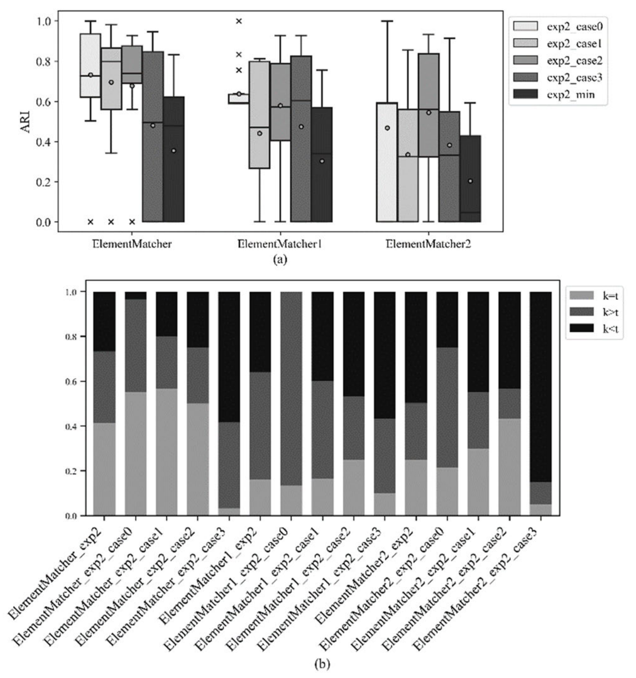


FIGURE 4. Experiment 2 - Object comparators: (a) ResultsPh1 based on ARI values; (b) results based on number recognized cluster –k, and number input table – t.

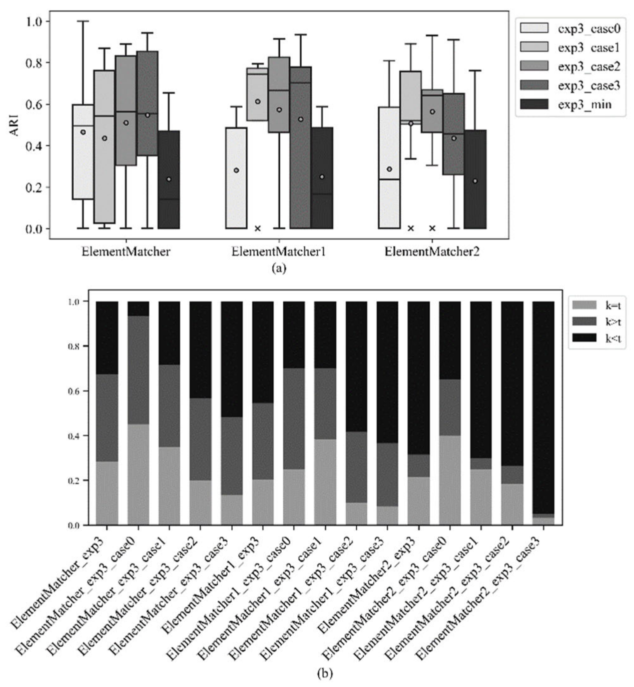


FIGURE 5. Experiment 3 - Object comparators: (a) ResultsPh1 based on ARI values; (b) results based on number recognized cluster –k, and number input table – t.

hypothesis should be rejected for all other cases and the consolidated measure, except for exp2_case3. Tukey's HSD test for consolidated measures showed a p-value of 0.006 between ElementMatcher and ElementMatcher2. Regarding the comparison of the number of recognized clusters and input tables, in Experiment 2, ElementMatcher had the largest number of results, in which the number of recognized clusters was greater than or equal to the number of input tables (73.33%), ElementMatcher1 (64.17%), and ElementMatcher2 (50.42%). According to the second criterion for analyzing the results to determine the object comparator in Experiment 2, ElementMatcher achieved the best results. In the case of ANOVA and Tukey's HSD test for the consolidated measure, ElementMatcher and ElementMatcher2 influenced ResultsPh1.

Table 6 shows the mean values of the three object comparators according to the descriptive statistics for each case in experiment 3.

TABLE 6. Experiment 3 - Mean ARI value for object comparator.

matcher_type	exp3_case0	exp3_case1	exp3_case2	exp3_case3	exp3_min
ElementMatcher	0.466	0.435	0.510	0.547	0.238
ElementMatcher1	0.281	0.613	0.573	0.528	0.249
ElementMatcher2	0.287	0.506	0.564	0.436	0.230

As in the previous two experiments, an ANOVA test was performed for all the cases in Experiment 3. The results showed that the null hypothesis should be rejected for exp3_case0 and exp3_case1. Since the ANOVA test did not show that the three object comparators significantly impacted ResultsPh1, the consolidated measure's descriptive statistics results were consulted. It was determined that ElementMatcher2 has the highest maximum value (0.76), followed by ElementMatcher (0.65) and ElementMatcher1 (0.59). As shown in Figure 5, ElementMatcher1 has a median value of 0.17 for the consolidated measure, followed by ElementMatcher at 0.14 and ElementMatcher2 at 0.00. Regarding the comparison of the number of recognized clusters, in Experiment 3, ElementMatcher had the most significant number of results, in which the number of recognized clusters was greater than or equal to the number of input tables by 67.5%, ElementMatcher1 by 54.58%, and ElementMatcher2 by 31.67%.

According to the second criterion for analyzing the results and determining the functions for comparing objects in Experiment 3, ElementMatcher achieved the best results. Regarding descriptive statistics for the consolidated measure, ElementMatcher1 had the most significant influence on the results of clustering structures (ResultsPh1).

2) ANALYSIS – DISTANCE FUNCTIONS

The comparative results for the six distance functions (linear, linear_weight, log, log_weight, reverse, and reverse_weight) in Experiments 1, 2, and 3 are shown in Figures 6, 7, and 8. The results based on the ARI values from ResultsPh1 are shown in Figures 6a, 7a, and 8a. Figures 6b, 7b, and 8b compare the number of identified clusters k with the number of input tables t. The circles in Figures 6a, 7a, and 8a represent mean values.

Table 7 shows the mean values of the six distance functions according to the descriptive statistics for each case in experiment 1.

TABLE 7. Experiment 1 - Mean ARI value for distance functions.

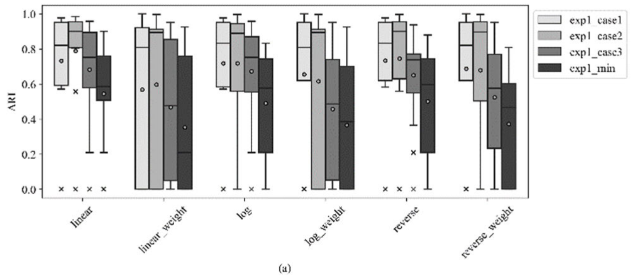
distance type	exp1_case1	exp1_case2	exp1_case3	exp1_min
linear	0.732	0.788	0.683	0.544
linear_weight	0.568	0.596	0.468	0.353
log	0.718	0.718	0.673	0.490
log_weight	0.654	0.615	0.458	0.366
reverse	0.734	0.745	0.651	0.501
reverse_weight	0.687	0.679	0.523	0.372

This step tested the influence of the sixth distance function on ARI values using 180 ResultsPh1. The results were divided into six groups according to distance type, with 30 results each. An ANOVA was used to determine whether there was a significant difference between the six distance functions. The test was repeated for each case and the consolidated measure in Experiment 1. The analysis showed that the null hypothesis was not confirmed only in the case of exp1_case3. When comparing the number of recognized clusters, the linear function had the most significant results in Experiment 1 (82.22%), in which the number of recognized clusters was greater than or equal to the number of input tables, followed by the log function (77.78%). Since the ANOVA test did not show that any of the six distance functions significantly impacted ResultsPh1, the results of descriptive statistics for the consolidated measure were used. Regarding the maximum value at the consolidated measure level, the functions linear_weight and log_weight had the highest recorded values (0.93), followed by linear (0.90), reverse (0.88), log (0.83), and reverse_weight (0.81). However, if we consider the median for the consolidated measure, the reverse function has the highest median value (0.90), followed by the linear and log functions (0.58).

A linear function can be selected according to the second criterion to determine the influence of the distance functions on the results of structure clustering. Based on the analysis of the ARI values of the consolidated measures, the linear_weight and log_weight functions can be selected according to the maximum value, and the reverse function according to the median.

Table 8 shows the mean values of the six distance functions according to the descriptive statistics for each case in experiment 2.

Based on the ANOVA test, the null hypothesis was not confirmed for the exp2_case0. Regarding the comparison of the number of recognized clusters and input tables, the linear and reverse functions had the highest number of experimental results (73.33%), in which the number of recognized clusters was greater than or equal to the number of input tables, followed by the log function (70.83%). Because ANOVA did not show that any of the six distance functions had a significant



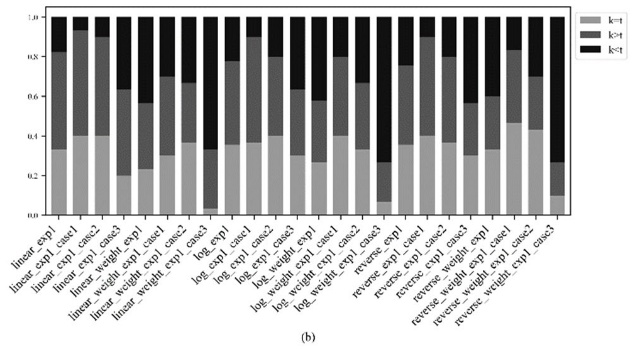


FIGURE 6. Experiment 1 – Distance functions: (a) ResultsPh1 based on ARI values; (b) results based on number recognized cluster –*k*, and number input table – *t*.

TABLE 8. Experiment 2- Mean ARI value for distance functions.

distance_type	exp2_case0	exp2_case1	exp2_case2	exp2_case3	exp2_min
linear	0.673	0.472	0.589	0.530	0.305
linear_weight	0.463	0.433	0.575	0.447	0.273
log	0.701	0.477	0.601	0.484	0.330
log_weight	0.536	0.474	0.591	0.351	0.236
reverse	0.706	0.509	0.622	0.504	0.333
reverse_weight	0.596	0.574	0.622	0.355	0.245

impact on the results of the clustering of structures, the results of the descriptive statistics for the consolidated measure were consulted. For the maximum value at the consolidated measurement level, the linear_weight function has the highest recorded value (0.83), followed by the log_weight function (0.73). However, if we consider the median of the combined measures, the highest value of the median has a reverse function (0.37) followed by linear and log functions (0.34).

A linear function can be selected according to the second criterion to determine the influence of the distance functions on the results of structure clustering. Based on the analysis of the ARI values of the consolidated measures, the linear_weight function can be selected according to the maximum value, and the reverse function according to the median.

Table 9 shows the mean values of the six distance functions according to the descriptive statistics for each case in experiment 3.

Based on the ANOVA test, the null hypothesis was rejected only for exp3_case0. Regarding the comparison of the number of recognized clusters, the linear and reverse functions had the highest number of results in Experiment 3 (65%),

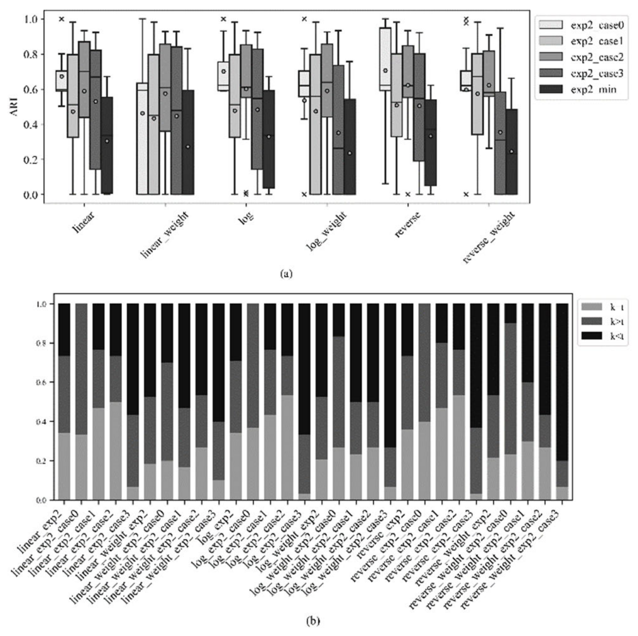


FIGURE 7. Experiment 2 – Distance function (a) ResultsPh1 based on ARI values; (b) results based on number recognized cluster –*k*, and number input table –*t*.

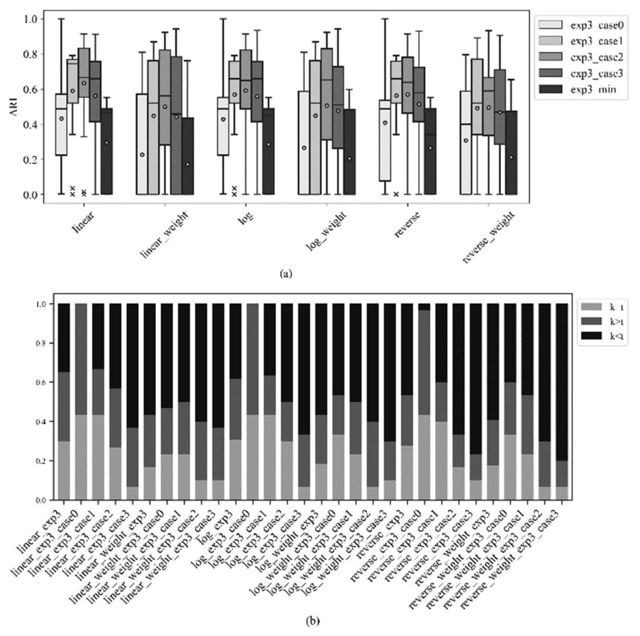


FIGURE 8. Experiment 3 – Distance function (a) ResultsPh1 based on ARI values; (b) results based on number recognized cluster –*k*, and number input table -*t*.

TABLE 9. Experiment 3 - Mean ARI value for distance functions.

distance_type	exp3_case0	exp3_case1	exp3_case2	exp3_case3	exp3_min
linear	0.432	0.590	0.634	0.562	0.297
linear_weight	0.226	0.447	0.500	0.442	0.171
log	0.428	0.569	0.591	0.558	0.284
log_weight	0.266	0.448	0.505	0.477	0.205
reverse	0.408	0.563	0.569	0.514	0.266
reverse_weight	0.307	0.491	0.495	0.468	0.211

in which the number of recognized clusters was greater than or equal to the number of input tables, followed by the log function (61.67%). Since the ANOVA test did not show that any of the six distance functions significantly influenced ResultsPh1, the results of descriptive statistics for the consolidated measure were consulted. For the maximum value of the consolidated measure, the linear_weight function had the highest recorded value (0.76), followed by the reverse_weight function (0.65). However, considering the median of the combined measures, the highest value had a linear function (0.47) followed by a log function (0.45).

According to the second criterion for determining the influence of distance functions on the results of the clustering structure, linear and reverse functions can be obtained. Based on an analysis of the ARI values of the consolidated measures, the linear_weight function can be selected according to the maximum value, and the linear function can be selected according to the median.

3) ANALYSIS OF CLUSTER CENTER DETECTORS

The comparative results for the five cluster center detectors (GroupMax, GroupMax2, MaxFreq, MinDistance, and PositionMax) in Experiments 1, 2, and 3 are shown in Figures 9, 10, and 11, respectively. The results based on the ARI values of ResultsPh1 are shown in Figures 9a, 10a, and 11a. Figures 9b, 10b, and 11b compare the number of identified clusters *k* with the number of input tables *t*. The circles in Figures 9a, 10a, and 11a denote mean values.

Table 10 shows the mean values of the five cluster center detectors according to the descriptive statistics for each case in experiment 1.

TABLE 10. Experiment 1 - Mean ARI value for cluster center detectors.

combiner_type	exp1_case1	exp1_case2	exp1_case3	exp1_min
GroupMax	0.470	0.665	0.508	0.315
GroupMax2	0.819	0.658	0.592	0.481
MaxFreq	0.724	0.829	0.835	0.656
MinDistance	0.740	0.764	0.686	0.569
PositionMax	0.657	0.536	0.259	0.168

In this step, the influence of the five cluster-center detector functions on the ARI values was tested using 180 ResultsPh1. The results were divided into five groups by combiner_type, with 36 results each. The ANOVA test determined whether there was a significant difference between the five cluster center detectors. The test was repeated for each case in Experiment 1. The analysis showed that the null hypothesis was not confirmed in any case, or at the consolidated measure level. Based on Tukey's HSD test, the interdependence between most functions for detecting the cluster center was determined in the case of the consolidated measure. All the other functions interacted with PositionMax; only PositionMax yielded the worst results. Analysis of the results of descriptive

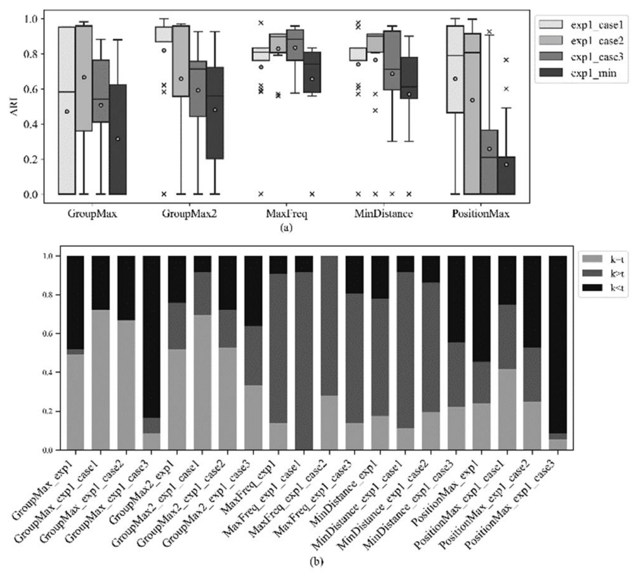


FIGURE 9. Experiment 1 – Cluster center detectors (a) ResultsPh1 based on ARI values; (b) results based on number recognized cluster –*k*, and number input table – *t*.

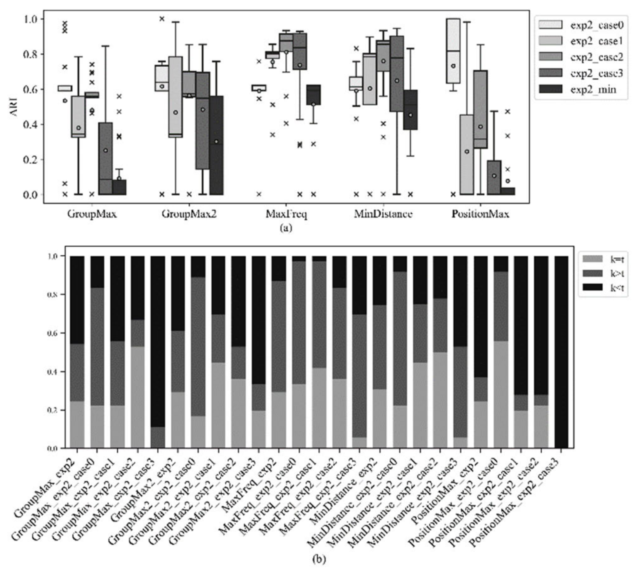


FIGURE 10. Experiment 2 – Cluster center detectors (a) ResultsPh1 based on ARI values; (b) results based on number recognized cluster –*k*, and number input table - *t*.

statistics for the consolidated measure determined that in the case of the maximum value at the level of the consolidated measure, the GroupMax2 function had the highest recorded value (0.93), followed by MinDistance (0.90). However, if we consider the median of the combined measures, the MaxFreq function has the highest median value (0.74) followed by the MinDistance function (0.61). Regarding the comparison of the number of recognized clusters, the MaxFreq function had the highest number of results in Experiment 1 (90.74%), in which the number of recognized clusters was greater than or equal to the number of input tables, followed by the MinDistance function (77.78%).

The MaxFreq function can be obtained according to the second criterion for determining the influence of distance functions on ResultsPh1. In the case of analysis based on the ARI values of the consolidated measures, the GroupMax2 function can be selected according to the maximum value and the MaxFreq function according to the median.

Table 11 shows the results for the mean values of the five cluster center detectors according to the descriptive statistics for every case in experiment 2.

TABLE 11. Experiment 2 - Mean ARI value for cluster center detectors.

combiner_type	exp2_case0	exp2_case1	exp2_case2	exp2_case3	exp2_min
GroupMax	0.534	0.378	0.481	0.251	0.091
GroupMax2	0.616	0.467	0.564	0.484	0.302
MaxFreq	0.590	0.755	0.811	0.738	0.513
MinDistance	0.591	0.604	0.760	0.648	0.453
PositionMax	0.732	0.244	0.386	0.107	0.076

In the case of the consolidated measure based on Tukey's HSD test, the interdependence between most of the functions for detecting the cluster center was determined. All the other functions interacted with PositionMax; only PositionMax yielded the worst results. Analysis of the results of descriptive statistics for the consolidated measure in Experiment 1 indicated that, in the case of the maximum value at the level of the consolidated measure, the MinDistance function had the highest recorded value (0.83), followed by GroupMax2 (0.76). However, if we consider the median for the consolidated measure, the MaxFreq function has the highest median value (0.59) followed by the MinDistance function (0.51). Regarding the comparison of the number of recognized clusters, the MaxFreq function had the most significant results (86.81%), in which the number of recognized clusters was greater than or equal to the number of input tables, followed by the MinDistance function (74.31%).

The MaxFreq function can be selected according to the second criterion for determining the influence of the distance functions on the results of structure clustering. Based on the analysis of the ARI values of the consolidated measures, the MinDistance function was selected according to the maximum value and the MaxFreq function was selected according to the median.

Table 12 shows the mean values of the five cluster center detectors according to the descriptive statistics for each case in experiment 3.

TABLE 12. Experiment 3 - Mean ARI value for cluster center detectors.

combiner_type	exp3_case0	exp3_case1	exp3_case2	exp3_case3	exp3_min
GroupMax	0.175	0.351	0.375	0.494	0.083
GroupMax2	0.160	0.456	0.507	0.503	0.143
MaxFreq	0.520	0.780	0.800	0.758	0.496
MinDistance	0.503	0.630	0.658	0.657	0.436
PositionMax	0.365	0.374	0.405	0.105	0.036

The ANOVA test showed that the null hypothesis was not confirmed for all cases in Experiment 3, or for the

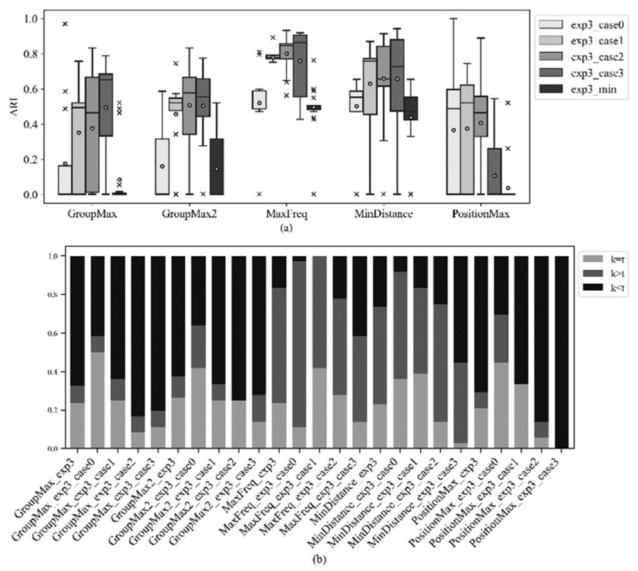


FIGURE 11. **Experiment 3 – Cluster center detectors (a) ResultsPh1 based on ARI values; (b) results based on number recognized cluster –*k*, and number input table – *t*.**

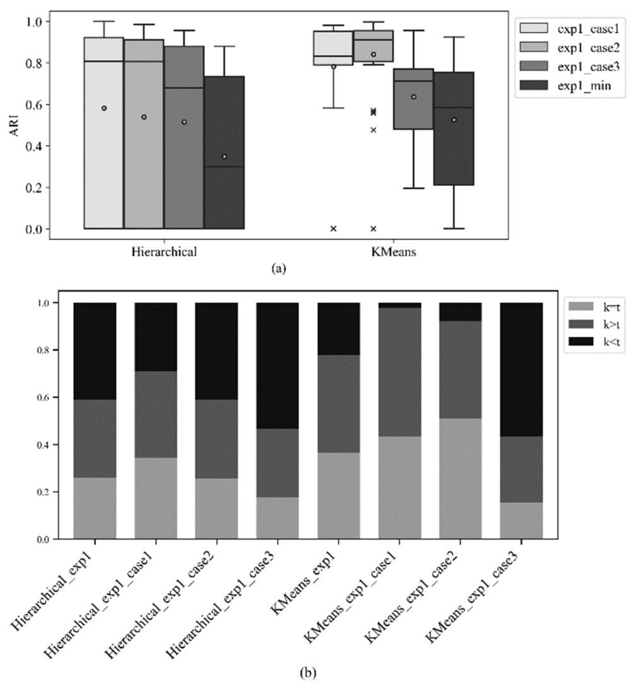


FIGURE 12. **Experiment 1 – Clustering algorithms (a) ResultsPh1 based on ARI values; (b) results based on number recognized cluster –*k*, and number input table – *t*.**

consolidated measure. Tukey's HSD test was used to establish the interdependence between most cluster center detectors. Regarding the comparison of the number of recognized clusters, the MaxFreq function had the most significant number of results, where the number of clusters was greater than or equal to the number of input tables (83.33%), followed by the MinDistance function (73.61%). An analysis of the descriptive statistics for the consolidated measure in Experiment 3 revealed that the MaxFreq function had the highest recorded maximum value (0.76), followed by MinDistance (0.65). However, for the median of the consolidated measure, the MaxFreq function had the highest median value (0.49), followed by MinDistance (0.47). Regarding the comparison of the number of recognized clusters, the MaxFreq function has the highest number of results (83.33%), followed by the MinDistance function (73.61%).

The MaxFreq function can be selected according to the first and second criteria for determining the impact of the cluster center detectors on ResultsPh1.

4) ANALYSIS – CLUSTERING ALGORITHMS

The comparative results of the two clustering algorithms (Hierarchical and K-means) for Experiments 1, 2, and 3 are shown in Figures 12, 13, and 14, respectively. The results based on the ARI values of ResultsPh1 are shown in Figures 12a, 13a, and 14a. Figures 12b, 13b, and 14b compare the number of identified clusters k with the number of input tables t. The circles in Figures 12a, 13a, and 14a denote mean values.

This step tested the influence of the two clustering algorithms on the ARI values of the ResultsPh1. The data analysis was performed using descriptive statistics. The analysis of the results based on the ARI value showed that at the level of all three cases of Experiment 1 and the consolidated measure, the K-means algorithm had a higher median and mean value than the Hierarchical algorithm. K-means has a higher maximum value for a consolidated measure (0.93) than for the Hierarchical algorithm (0.88). When comparing the number of recognized clusters, the K-means algorithm had a higher percentage of Experiment 1 results (77.78%) than the Hierarchical algorithm (58.89%), where the number of recognized clusters was greater than or equal to the number of input tables.

Based on previous analysis, K-means clustering can be used in Experiment 1.

An analysis of the results based on the ARI values showed that the K-means algorithm had higher mean and median values than the hierarchical algorithm in all four cases of Experiment 2, and in the consolidated measure. Based on the consolidated measure, the Hierarchical algorithm achieves the highest maximum value (0.83), followed by K-means (0.76).

When comparing the results of Experiment 2 for the number of recognized clusters, the Hierarchical algorithm yielded slightly more results (62.78%) than the K-means algorithm (62.50%), where the number of recognized clusters was greater than or equal to the number of input tables.

The Hierarchical algorithm can be used according to the second criterion to determine the impact on the Results ResultsPh1. Based on the analysis of the ARI values of the consolidated measures, the hierarchical algorithm can be chosen if the selection criterion is a maximum value, and the K-means algorithm if the selection criterion is the median.

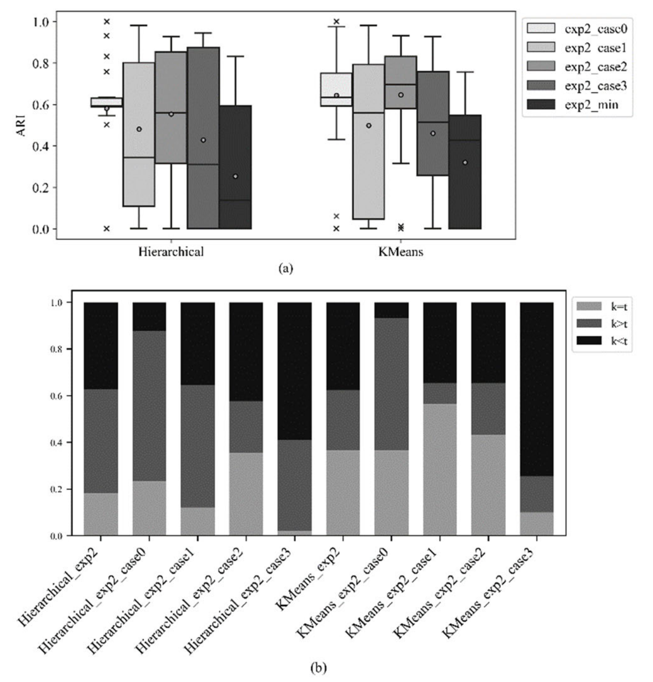


FIGURE 13. Experiment 2 – Clustering algorithms (a) ResultsPh1 based on ARI values; (b) results based on number recognized cluster –*k*, and number input table – *t*.

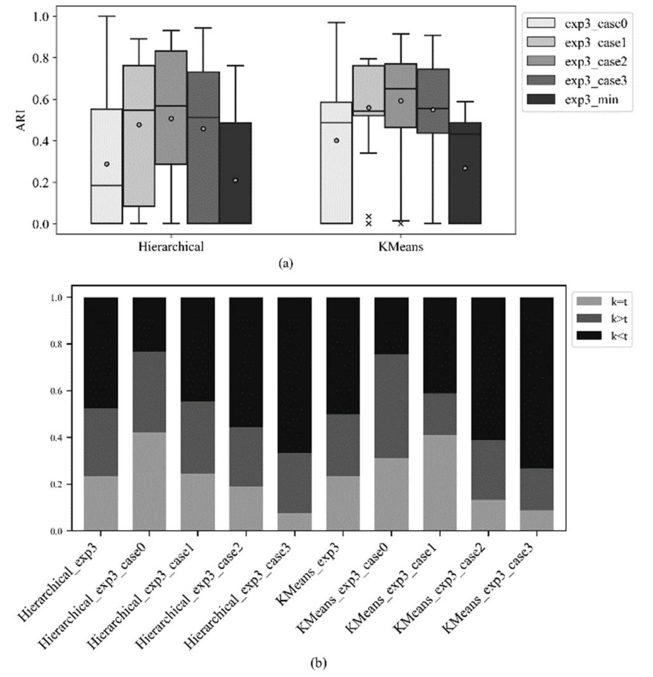


FIGURE 14. Experiment 3 – Clustering algorithms (a) ResultsPh1 based on ARI values; (b) results based on number recognized cluster –*k*, and number input table – *t*.

An analysis of the results based on the ARI values showed that the K-means algorithm had higher mean and median values than the Hierarchical algorithm in all four cases of Experiment 3 and for the consolidated measure.

When comparing the results of Experiment 3 for the number of recognized clusters, the Hierarchical algorithm yielded slightly more results (52.50%) than the K-means algorithm (50.00%), where the number of recognized clusters was greater than or equal to the number of input tables.

The Hierarchical algorithm can be used according to the second criterion to determine the impact on the Results ResultsPh1. Based on the analysis of the ARI values of the consolidated measures, the Hierarchical algorithm can be chosen if the selection criterion is a maximum value and the K-means algorithm if the selection criterion is a median.

5) ANALYSIS OF THE INTERDEPENDENCE OF ALL PARAMETERS

The results were observed separately in previous analyses at the case and experimental levels. In contrast, the results of the three experiments were observed together to determine which input parameter combination provided the best solution for all three experiments. An additional consolidated measure ('exp_mul') was added to this analysis to consider the three consolidated measures created at the practical level and to connect the three experiments. Adjusted Rand Information (ARI) and Adjusted Mutual Information (AMI) for every ResultsPh1 were used for this analysis. Namely, ResultsPh1 by ARI and AMI values for the consolidated measure 'exp_mul' were compared.

When the ARI and AMI scores are approximately zero, indicating randomly labeled clusters, this value will not be further considered in the analysis. The analysis found that the consolidated measure 'exp_mul' with AMI values equals zero for 50% of ResultsPh1. The analysis also found that for 53.3% of ResultsPh1, the consolidated measure 'exp_mul' with ARI values equals zero. In the next step, ResultsPh1 was taken for 'exp_mul' different from zero in both AMI and ARI values. These two sets of ResultsPh1 differed by six results, that is, 3.33% more ResultsPh1 were contained in the set of AMI values. The 'exp_mul' values ranged from 0.011 to 0.014 for all six results.

Given that this is approximately equal to zero, the decision to continue working with the set of 84 ResultsPh1 can be accepted, because they appear in both the ARI and AMI sets of ResultsPh1. Then, from ResultsPh1, where 'exp_mul' is not equal to zero, the results where the number of clusters is less than the number of tables are excluded. Fifty-six results were excluded and 28 structure-clattering results were obtained. The result of the previous actions is a set of ResultsPh1 (further called SelectedResulsPh1) where 'exp_mul' differs from zero according to ARI and AMI values in each case. In addition, the number of generated clusters is greater than or equal to the number of input tables during the structure clustering.

By simply counting the SelectedResulsPh1 (28 results) using all input parameters, ElementMatcher was found to participate in 14, ElementMatcher1 in 11, and ElementMatcher2 in three results. In the case of the cluster center detectors, only three functions appeared in SelectedResulsPh1: MaxFreq

TABLE 13. The best result for parameters STExtract algorithms selected by counting.

matcher_type	distance_type	combiner_type	selector_type	quotient	Adjusted Rand Index (ARI) Value	position	Adjusted Mutual Information (AMI) value	position
ElementMatcher	log	MaxFreq	Hierarchical	exp_mul	0.16986834	13th	0.17056096	12th
				exp_average	0.77572727	16th	0.78918182	17th
ElementMatcher	linear	MaxFreq	Hierarchical	exp_mul	0.16986834	13th	0.17056096	12th
				exp_average	0.77381818	17th	0.78681818	18th

TABLE 14. The best result for parameters STExtract algorithms selected by maximal exp_mul and the average value.

matcher_type	distance_type	combiner_type	selector_type	quotient	Adjusted Rand Index (ARI) Value	position	Adjusted Mutual Information (AMI) value	position
ElementMatcher2	linear_weight	MaxFreq	Hierarchical	exp_mul	0.33572885	1st	0.40379935	1st
				exp_average	0.81890909	1st	0.84945455	1st
ElementMatcher	linear_weight	MaxFreq	Hierarchical	exp_mul	0.29723047	2nd	0.33606878	2nd
				exp_average	0.81572727	3rd	0.84027273	2nd
ElementMatcher1	log_weight	MaxFreq	KMeans	exp_mul	0.24396752	8th	0.29606564	4th
				exp_average	0.81500000	4th	0.83645455	3rd
ElementMatcher1	linear_weight	MaxFreq	KMeans	exp_mul	0.24396752	8th	0.29606564	4th
				exp_average	0.81454545	5th	0.83509091	4th
ElementMatcher1	log	MinDistance	Hierarchical	exp_mul	0.26448749	6th	0.28451986	5th
				exp_average	0.81590909	2nd	0.82772727	9th

TABLE 15. Experiment 1 – Results of data clustering algorithms and column level for selected combinations input parameters.

Experiment 1	Accuracy	The first combination of input parameters					The second combination of input parameters				
		Birch	FCM	Gaussian Mixture	KMeans	Mini Batch KMeans	Birch	FCM	Gaussian Mixture	KMeans	Mini Batch KMeans
case1	accuracy_{ph1}	0.881	0.881	0.881	0.881	0.881	0.881	0.881	0.881	0.881	0.881
	accuracy_{ph2}	0.848	0.848	0.848	0.848	0.848	0.848	0.848	0.848	0.848	0.848
case2	accuracy_{ph1}	0.927	0.927	0.927	0.927	0.927	0.917	0.917	0.917	0.917	0.917
	accuracy_{ph2}	0.880	0.874	0.880	0.880	0.880	0.880	0.880	0.880	0.880	0.880
case3	accuracy_{ph1}	0.852	0.852	0.852	0.852	0.852	0.953	0.953	0.953	0.953	0.953
	accuracy_{ph2}	0.783	0.783	0.786	0.786	0.786	0.882	0.882	0.886	0.886	0.886

in 18, GrupMax2 in 5, and MinDistance in 5. Regarding the distance functions of SelectedResulsPh1, the linear function yielded eight results: six for linear_weight, eight for log inside, three for log_weight, two for reverse, and one for reverse_weight. When examining the clustering algorithm, it was found that the Hierarchical algorithm yielded 19 results and K-means yielded nine results.

The average value for all (11) cases was calculated based on the AMI and ARI values for further selection from SelectedResulsPh1. Each unique 'ex_mul' and 'exp_average' position value was assigned. In the case of 'exp_mul,' according to ARI, 19 unique values and 25 for 'exp_average' were determined. In the case of 'exp_mul' according to AMI, 16 unique values and 25 for 'exp_average' were determined. An insight into the results showed that there are parameter combinations with the same 'exp_mul' or 'exp_average' value. In this case, the data are processed without rounding. Two possible combinations of parameters that yielded the best results were obtained based on independent counting of the input parameters. The results for these two input parameter combinations are listed in Table 13. Considering the number of unique values for ARI and AMI, it was determined that these combinations of input parameters were not in the top ten results. Table 14 shows the results that fall in the top 10 values per position for two values,'exp_mul' and 'exp_average' for ARI and two values for 'exp_mul' and 'exp_average.' The first two parameter combinations listed in Table 14 are among the top three combinations. These were selected as the best combination of input parameters, for which the results of the Phase1 STCExtract algorithms were further analyzed at the column level.

H. EVALUATION PHASE 2 – CLUSTERING COLUMNS

The accuracy metric results of Phase 2 were independently analyzed for each parameter of Phase 2 (machine learning algorithm for clustering columns) of the proposed STCExtract algorithm for each experiment and case within the experiment for the combination of input parameters that after Phase 1 gave the best results clustered structures.

Table 15, Table 16, and Table 17 show the combined results of Phase 1 (the structure clustering phase) and Phase 2 (the column clustering phase) for two combinations of input parameters (selected clustering algorithm, distance function, cluster center detectors, and object comparators) that yielded

TABLE 16. Experiment 2 – Results of data clustering algorithms and column level for selected combinations input parameters.

Experiment 2	Accuracy	The first combination of input parameters					The second combination of input parameters				
		Birch	FCM	Gaussian Mixture	KMeans	Mini Batch KMeans	Birch	FCM	Gaussian Mixture	KMeans	Mini Batch KMeans
case0	$accuracy_{ph1}$	0.661	0.661	0.661	0.661	0.661	0.894	0.894	0.894	0.894	0.894
	$accuracy_{ph1\_scalar}$	0.788	0.788	0.788	0.788	0.788	0.788	0.788	0.788	0.788	0.788
	$accuracy_{ph1\_vector}$	0.534	0.534	0.534	0.534	0.534	1.000	1.000	1.000	1.000	1.000
	$accuracy_{ph2}$	0.623	0.623	0.623	0.623	0.623	0.623	0.623	0.811	0.811	0.811
	$accuracy_{ph2\_scalar}$	0.747	0.747	0.747	0.747	0.747	0.747	0.747	0.747	0.747	0.747
	$accuracy_{ph2\_vector}$	0.499	0.499	0.499	0.499	0.499	0.499	0.499	0.874	0.874	0.874
case1	$accuracy_{ph1}$	0.765	0.765	0.765	0.765	0.765	0.898	0.898	0.898	0.898	0.898
	$accuracy_{ph1\_scalar}$	0.881	0.881	0.881	0.881	0.881	0.881	0.881	0.881	0.881	0.881
	$accuracy_{ph1\_vector}$	0.532	0.532	0.532	0.532	0.532	0.931	0.931	0.931	0.931	0.931
	$accuracy_{ph2}$	0.731	0.725	0.731	0.731	0.731	0.856	0.725	0.856	0.856	0.856
	$accuracy_{ph2\_scalar}$	0.848	0.839	0.848	0.848	0.848	0.848	0.839	0.848	0.848	0.848
	$accuracy_{ph2\_vector}$	0.496	0.496	0.496	0.496	0.496	0.872	0.496	0.872	0.872	0.872
case2	$accuracy_{ph1}$	0.947	0.947	0.947	0.947	0.947	0.927	0.927	0.927	0.927	0.927
	$accuracy_{ph1\_scalar}$	0.930	0.930	0.930	0.930	0.930	0.921	0.921	0.921	0.921	0.921
	$accuracy_{ph1\_vector}$	1.000	1.000	1.000	1.000	1.000	0.944	0.944	0.944	0.944	0.944
	$accuracy_{ph2}$	0.880	0.783	0.880	0.880	0.880	0.787	0.876	0.880	0.880	0.880
	$accuracy_{ph2\_scalar}$	0.883	0.877	0.883	0.883	0.883	0.883	0.877	0.883	0.883	0.883
	$accuracy_{ph2\_vector}$	0.872	0.499	0.872	0.872	0.872	0.499	0.872	0.872	0.872	0.872
case3	$accuracy_{ph1}$	0.953	0.953	0.953	0.953	0.953	0.794	0.794	0.794	0.794	0.794
	$accuracy_{ph1\_scalar}$	0.952	0.952	0.952	0.952	0.952	0.846	0.846	0.846	0.846	0.846
	$accuracy_{ph1\_vector}$	0.958	0.958	0.958	0.958	0.958	0.532	0.532	0.532	0.532	0.532
	$accuracy_{ph2}$	0.817	0.817	0.883	0.883	0.883	0.746	0.743	0.746	0.746	0.746
	$accuracy_{ph2\_scalar}$	0.881	0.881	0.885	0.885	0.885	0.795	0.792	0.795	0.795	0.795
	$accuracy_{ph2\_vector}$	0.496	0.496	0.872	0.872	0.872	0.496	0.496	0.496	0.496	0.496

the best results for Phase 1. The input parameters "ElementMatcher2 - linear_weight - MaxFreq – Hierarchical" are represented by the column labeled "The first combination of input parameters," while the input parameters "ElementMatcher - linear_weight - MaxFreq – Hierarchical" are represented by the column labeled "The second combination of input parameters."

Table 15 shows the results of Experiment 1, where tables with columns can only have a scalar data type and only values $accuracy_{ph1}$ and $accuracy_{ph2}$ because $accuracy_{ph1} = accuracy_{ph1\_scalar}$ and $accuracy_{ph2} = accuracy_{ph2\_scalar}$.

Based on the results shown in Table 15, Table 16, and Table 17, it can be observed that the Gaussian mixture, K-means, and Mini-batch K-means yielded the same results for correctly recognized rows at the table and column levels. By contrast, the results for the remaining two algorithms were the same or slightly worse. Suppose that the results for both combinations of input parameters for the Gaussian mixture, K-means, and Mini-batch k-means algorithms are compared. In this case, it can be said that for Experiment 1, the second combination of input parameters provided a better result for Case 3. The results were identical for both input-parameter combinations in the first two cases. In Experiment 2, the first combination of input parameters yielded better results in both cases and the second combination of input parameters yielded better results in the other two cases. In Experiment 3, both combinations of input parameters yielded the same results in one case. For the two cases, the first combination of input parameters yielded a better result, and for one case, the second combination of input data yielded a better result. The results show that the STCExtract algorithm consistently recognizes tables with all scalar-type columns, and does not recognize neighboring vector-type columns (e.g., exp3_case3).

I. RESULTS AND DISCUSSION

The results of Experiment 4, which evaluated the accuracy of the STCExtract algorithm, are presented in Tables 18, 19, and 20. The answers to the research questions that validate the proposed STCExtract algorithms are presented below.

1) ANSWER TO RQ1

The proposed STECxtract algorithm comprises of two phases. Data structures are recognized in the first phase (the tables from which the data originate) and second phase (the columns of the tables from which the data originate). Machine learning techniques were used at each stage to identify appropriate table and table column structures.

The STCExtract algorithm correctly arranged the structures of the tables with an accuracy of 94.4% to 100%, as shown in Table 18 from Experiment 4. The accuracy of the STCExtract algorithm in the second phase (when the data were allocated to columns) ranged from 59.7% to 90.2%. From Table 18, it can be observed that the accuracy of the proposed algorithm varies from case to case. However, to better cover the behavior of the algorithm in its two phases, the lowest and highest accuracies recorded between the phases

TABLE 17. Experiment 3 – Results of data clustering algorithms and column level for selected combinations input parameters.

Experiment 3	Accuracy	The first combination of input parameters					The second combination of input parameters				
		Birch	FCM	Gaussian Mixture	KMeans	Mini Batch KMeans	Birch	FCM	Gaussian Mixture	KMeans	Mini Batch KMeans
case0	accuracy_{ph1}	0.886	0.886	0.886	0.886	0.886	0.702	0.702	0.702	0.702	0.702
	$\text{accuracy}_{ph1\_scalar}$	0.785	0.785	0.785	0.785	0.785	0.785	0.785	0.785	0.785	0.785
	$\text{accuracy}_{ph1\_vector}$	0.988	0.988	0.988	0.988	0.988	0.619	0.619	0.619	0.619	0.619
	accuracy_{ph2}	0.574	0.574	0.648	0.648	0.648	0.574	0.574	0.574	0.574	0.574
	$\text{accuracy}_{ph2\_scalar}$	0.746	0.746	0.746	0.746	0.746	0.746	0.746	0.746	0.746	0.746
	$\text{accuracy}_{ph2\_vector}$	0.400	0.400	0.549	0.5 49	0.549	0.400	0.400	0.400	0.400	0.400
case1	accuracy_{ph1}	0.796	0.796	0.796	0.796	0.796	0.796	0.796	0.796	0.796	0.796
	$\text{accuracy}_{ph1\_scalar}$	0.882	0.882	0.882	0.882	0.882	0.882	0.882	0.882	0.882	0.882
	$\text{accuracy}_{ph1\_vector}$	0.625	0.625	0.625	0.625	0.625	0.625	0.625	0.625	0.625	0.625
	accuracy_{ph2}	0.703	0.703	0.703	0.703	0.703	0.703	0.697	0.703	0.703	0.703
	$\text{accuracy}_{ph2\_scalar}$	0.848	0.848	0.848	0.848	0.848	0.848	0.840	0.848	0.848	0.848
	$\text{accuracy}_{ph2\_vector}$	0.411	0.411	0.411	0.411	0.411	0.411	0.411	0.411	0.411	0.411
case2	accuracy_{ph1}	0.940	0.940	0.940	0.940	0.940	0.849	0.849	0.849	0.849	0.849
	$\text{accuracy}_{ph1\_scalar}$	0.922	0.922	0.922	0.922	0.922	0.922	0.922	0.922	0.922	0.922
	$\text{accuracy}_{ph1\_vector}$	0.995	0.995	0.995	0.995	0.995	0.630	0.630	0.630	0.630	0.630
	accuracy_{ph2}	0.803	0.803	0.803	0.803	0.803	0.766	0.766	0.766	0.766	0.766
	$\text{accuracy}_{ph2\_scalar}$	0.884	0.884	0.884	0.884	0.884	0.884	0.884	0.884	0.884	0.884
	$\text{accuracy}_{ph2\_vector}$	0.558	0.558	0.558	0.558	0.558	0.411	0.411	0.411	0.411	0.411
case3	accuracy_{ph1}	0.821	0.821	0.821	0.821	0.821	0.899	0.899	0.899	0.899	0.899
	$\text{accuracy}_{ph1\_scalar}$	0.859	0.859	0.859	0.859	0.859	0.952	0.952	0.952	0.952	0.952
	$\text{accuracy}_{ph1\_vector}$	0.629	0.629	0.629	0.629	0.629	0.629	0.629	0.629	0.629	0.629
	accuracy_{ph2}	0.732	0.730	0.732	0.732	0.732	0.806	0.806	0.806	0.806	0.806
	$\text{accuracy}_{ph2\_scalar}$	0.796	0.793	0.796	0.796	0.796	0.885	0.885	0.885	0.885	0.885
	$\text{accuracy}_{ph2\_vector}$	0.412	0.412	0.412	0.412	0.412	0.412	0.412	0.412	0.412	0.412

TABLE 18. Rows that are correctly recognized by tables and by table's columns.

Exp.4	Accuracy	The first combination of input parameters			The second combination of input parameters		
		Gaussian Mixture	KMeans	Mini Batch KMeans	Gaussian Mixture	KMeans	Mini Batch KMeans
case1s	accuracy_{ph1}	0.944	0.944	0.944	1.000	1.000	1.000
	accuracy_{ph2}	0.892	0.892	0.892	0.902	0.902	0.902
case1v	accuracy_{ph1}	1.000	1.000	1.000	1.000	1.000	1.000
	$\text{accuracy}_{ph1\_scalar}$	1.000	1.000	1.000	1.000	1.000	1.000
	$\text{accuracy}_{ph1\_vector}$	1.000	1.000	1.000	1.000	1.000	1.000
	accuracy_{ph2}	0.601	0.601	0.601	0.601	0.601	0.601
	$\text{accuracy}_{ph2\_scalar}$	0.902	0.902	0.902	0.902	0.902	0.902
	$\text{accuracy}_{ph2\_vector}$	0.000	0.000	0.000	0.000	0.000	0.000
case1n	accuracy_{ph1}	0.994	0.994	0.994	0.994	0.994	0.994
	accuracy_{ph2}	0.597	0.597	0.597	0.597	0.597	0.597
case2s	accuracy_{ph1}	0.962	0.962	0.962	0.991	0.991	0.991
	accuracy_{ph2}	0.911	0.911	0.911	0.917	0.917	0.917
case2v	accuracy_{ph1}	0.996	0.996	0.996	0.996	0.996	0.996
	$\text{accuracy}_{ph1\_scalar}$	0.995	0.995	0.995	0.995	0.995	0.995
	$\text{accuracy}_{ph1\_vector}$	0.999	0.999	0.999	0.999	0.999	0.999
	accuracy_{ph2}	0.902	0.902	0.902	0.902	0.902	0.902
	$\text{accuracy}_{ph2\_scalar}$	0.912	0.912	0.912	0.912	0.912	0.912
	$\text{accuracy}_{ph2\_vector}$	0.872	0.872	0.872	0.872	0.872	0.872

in each case of Experiment 4 were used as the range of the accuracy measure.

This algorithm does not address the recognition of data subtypes or names of the corresponding columns. The objective of new research on this issue might be to recognize the names of data subtypes, columns, tables, and their relationships.

2) ANSWER TO RQ2

In the first and second phases of the STCExtract algorithm, the impact of six different independent parameters was analyzed to identify the similarity of tables or columns. Based on Experiment 4 and its evaluation, the STCExtract algorithm better recognized tables with all columns and scalar data types. For case1s and the second combination of parameters,

TABLE 19. Rows that are correctly recognized by tables but not by table's columns.

Exp.4	Accuracy	The first combination of input parameters			The second combination of input parameters		
		Gaussian Mixture	KMeans	Mini Batch KMeans	Gaussian Mixture	KMeans	Mini Batch KMeans
Case1s	$accuracy_{ph2\_scalar}$	0.051	0.051	0.051	0.098	0.098	0.098
Case1v	$accuracy_{ph2\_scalar}$	0.098	0.098	0.098	0.098	0.098	0.098
	$accuracy_{ph2\_vector}$	0.000	0.000	0.000	0.000	0.000	0.000
Case1n	$accuracy_{ph2\_scalar}$	0.064	0.064	0.064	0.064	0.064	0.064
Case2s	$accuracy_{ph2\_scalar}$	0.051	0.051	0.051	0.074	0.074	0.074
Case2v	$accuracy_{ph2\_scalar}$	0.083	0.083	0.083	0.083	0.083	0.083
	$accuracy_{ph2\_vector}$	0.127	0.127	0.127	0.127	0.127	0.127

TABLE 20. Rows that are correctly recognized by tables but not by table's columns in case of substitution or deleting column.

Exp.4	Accuracy	The first combination of input parameters			The second combination of input parameters		
		Gaussian Mixture	KMeans	Mini Batch KMeans	Gaussian Mixture	KMeans	Mini Batch KMeans
Case1s	$accuracy_{ph2\_scalar}$	0.046	0.046	0.046	0.048	0.048	0.048
Case1v	$accuracy_{ph2\_scalar}$	0.048	0.048	0.048	0.048	0.048	0.048
	$accuracy_{ph2\_vector}$	0.000	0.000	0.000	0.000	0.000	0.000
Case1n	$accuracy_{ph2\_scalar}$	0.032	0.032	0.032	0.032	0.032	0.032
Case2s	$accuracy_{ph2\_scalar}$	0.048	0.048	0.048	0.040	0.040	0.040
Case2v	$accuracy_{ph2\_scalar}$	0.049	0.049	0.049	0.049	0.049	0.049
	$accuracy_{ph2\_vector}$	0.050	0.050	0.050	0.050	0.050	0.050

the results in Table 18 show that the STCExtract algorithm correctly allocated 100% of the rows according to the tables to which they belong. By contrast, in Phase 2, when allocating the columns to the tables to which they belong, the STCExtract algorithm correctly allocated 90.2% of the data. Table 19 shows that 9.8% of the rows belong to the tables but not to the content of the columns. Table 20 shows that 9.8% of the rows and 4.8% of the rows substitute or delete the columns. For case2s, the STCExtract algorithm did not successfully allocate 0.9% of the rows per table and column.

The columns were allocated successfully to 91.7% of the rows and 7.4% were incorrectly recognized. Table 20 shows that 7.4% of the rows and 4% of the rows substitute or delete columns. For each case, the number of rows in which columns were substituted or deleted was approximately 5%. A difference of 1% refers to rows incorrectly allocated by the tables. For case1n, the STCExtract algorithm correctly allocated 99.4% of the rows depending on the corresponding tables. In Phase 2, the STCExtract algorithm correctly allocated 59.7% of row columns to the corresponding tables. The STCExtract algorithm also successfully grouped 6.4% of the data into columns, of which 3.2% were in rows where the columns were substituted or deleted. According to Tables 18 and 19, the total number of rows allocated by the table and columns can be determined, which for case1n is 66.1%. Only 1% of the rows in case1n were not adequately allocated by the tables and columns. A difference of 33.9% was achieved by the rows successfully allocated by the tables; however, the number of columns was required to be recognized correctly. This is one of the reasons for the large percentage of poorly clustered columns in the open_pubs table (which, in the input test dataset, makes up 33.35% of the data). The open _pubs table contains one column with a structured address consisting of four values separated by a comma. Consequently, it is possible to assert that the STCExtract algorithm cannot recognize the number of columns in a table when a column is intended to have structured data separated by characters, which the STCExtract algorithm recognizes as a delimiter parameter. For example, address fields can be separated using commas. The STCExtract algorithm divides the columns into independent columns.

The STCExtract algorithm is modular and can be extended by using new rules and functions to detect scalar data types. This problem may be the subject of future work that would solve the problem of determining the number of columns and sorting the data by columns, only for part of the data where it is not well done (e.g., case1n).

3) ANSWER TO RQ3

In the first and second phases of the STCExtract algorithm, the influence of the six independent functions on detecting the similarity of tables or columns was investigated. Based on Experiment 4 and its evaluation, it was found that the STCExtract algorithm does not recognize the data structure at the column level, where the minimum number of elements of most vectors is greater than one, as shown in case1v. However, for case2v, the STCExtract algorithm allocated 90.2% of the rows at the column level and practically all rows (99.6%) according to the corresponding tables. The STCExtract algorithm correctly assigned 99.5% of the rows in the table where the table had all columns with a scalar data type, and correctly assigned 99.9% of the rows in the belonging table with columns with a scalar and a vector data type. The STCExtract algorithm correctly allocated 91.2 percent of the rows at the column level during Phase 2, also known

TABLE 21. Information regarding datasets.

Data sources	# datasets	#rows in datasets	#columns	Dtype	#columns with scalar data type	#columns with vector data type	#columns values	#empty values in columns	#columns with unique values	memory usage
IMDb database	title.basic.tsv.gz	8982588	9	object(9)	8	1	80843301	32	1	616.8+ MB
	title.principals.tsv.gz	50682389	6	int64(1), object(5)	6	0	304094340	0	0	2.3+ GB
	title.episode.tsv.gz	6746852	4	object(4)	4	0	26987412	0	1	205.9+ MB
	name.basics.tsv.gz	11692915	6	object(6)	4	2	70157496	2505480	1	535.3+ MB
The Game Reviews	review_info.csv	999999	9	bool(1), float64(1), int64(6), object(1)	9	0	9000000	0	1	62.0+ MB
Calendar	calendar.tsv	31050093	7	float64(2), int64(1), object(4)	7	0	217350658	1856	0	1.6+ GB
Barcelona Traffic Accidents	accidents_opendata.csv	110654	17	float64(6), object(11)	17	0	1881135	20136	0	14.4+ MB
Monthly electricity production	data.csv	181914	12	float64(4), int64(3), object(5)	12	0	2182980	17105	0	16.7+ MB
Port Shipment Dataset	shipping_data.csv	263820	8	float64(5), object(3)	8	0	2110568	2861	0	16.1+ MB
Large Movie Dataset	movies_dataset.csv	25000094	5	float64(1), int64(2), object(2)	4	1	125000475	0	0	953.7+ MB
Every Pub in England	open_pubs.csv	51565	9	float64(1), int64(2), object(6)	9	1	464094	2	0	3.5+ MB
IIT Admissions Dataset	student_data.csv	199999	10	int64(6), object(4)	10	0	2000000	0	0	15.3+ MB

as the column clustering phase, for tables with all scalar data-type columns. Of the other 8.3% of rows where the number of columns was correctly clustered, a discrepancy occurred in 4.9% owing to the substitution or deletion of columns. The remaining 3.4% of poorly clustered columns may be because some column values contain a delimiter such as a vertical bar. However, the column value is remerged from multiple columns during phase reconstruction, and the STCExtract algorithm uses a coma. Thus, poor remerging of only one column can degrade the overall accuracy in Phase 2. Because the number of columns was not successfully determined, the column clustering process was unsuccessful in tables with a column of vector data type (case1v). For case2v, almost all rows (99.9%) were correctly allocated by tables, whereas columns were successfully allocated for 87.2% of the rows. Of the other 12.7% of rows in which the number of columns was correctly recognized, a discrepancy of 5% was due to the substitution or deletion of columns. In the remaining 7.7% of the cases, the values in one or more columns likely contained a delimiter.

The STCExtract algorithm is modular and can be expanded by using new rules and functions to detect vector data types. This problem may be the subject of future work that would solve the problem of determining the number of columns and sorting the data by columns only for parts of the data where it is well done (e.g., case1v).

After evaluating the STCExtract algorithm, the best results were obtained when ElementMatcher was selected as an object comparison function, linear_weight as a function to determine the distance of the given object from the cluster center, MaxFreq as the cluster center detection function, a hierarchical algorithm for the clustering of structures at the table level, and clustering columns for previously clustered converted structures from Phase 1 using the Gaussian mixture, K-means, and Mini-batch K-means algorithms.

J. THREATS TO VALIDITY

Although the data used in the experiments were obtained from various sources, they may not accurately reflect all the actual datasets or their structures. Twelve tables from various datasets are used as the basis for testing the proposed algorithm. Tables 2 and 3 show that data from the various datasets were combined. A maximum of six datasets was combined. The algorithm is yet to be tested on more than six messy datasets.

Other concerns are related to the number of columns in the dataset. STCExtract is yet to be tested if the dataset contains fewer than three or more than 17 columns. Also, STCExtract was not tested if the dataset contained more than two columns with a vector data type.

The STCExtract algorithm was tested only for balanced test datasets containing almost equal rows in each table. Minor discrepancies in the number of rows selected by the dataset occurred when the labeled dataset was created.

Additional concerns are related to the clustering methods and hard clustering. In hard clustering, data points belong entirely to a cluster. The STCExtract algorithm employs a hard clustering approach for clustering structures, in which a structure can be in only one cluster. However, multiple tables with distinct column names and data can be described in real datasets with the same converted structure (structure of data type). When testing the STCExtract algorithm, the hard clustering approach did not affect the percentage of rows and columns correctly recognized, because the algorithm always used the final test dataset, where the converted structures could not belong to different tables. However, this may have occurred in real datasets.

A table with a column that contains information about the Uniform Resource Locator (URL) (for example, a typical URL could have the form http://www.example.com, which indicates a protocol (HTTP) and hostname (www.example.com)), Time in ISO Time Format (for example, HH:mm:ss.SSSXXX), ISO Date Time Format (for example, yyyy-MM-dd'T' HH:mm:ss.SSSXXX) were not used for testing the STCExtract algorithms because when the data is tokenized, it separates this data into multiple columns because they contain a punctuation mark 'colon.' The algorithm divides tokens into multiple parts. For example, a URL token can be divided into protocols, addresses and ports. Time was divided into hours, minutes, and seconds.

The algorithm was tested, and it was found that if the data contained a structured address with parts separated by a comma (for example, Country, City, Town), it does not recognize the number of columns.

When clustering columns, the knee function was not used to determine the parameter k (number of columns), but the result of phase one was used, and the parameter k for clustering at the column level was determined based on the number of columns of the structure that had the largest number of rows in the cluster.

If the minimum number of elements in each vector is greater than one, the STCExtract algorithm must better recognize the structures at the column level for vector-type data.

In addition, the proposed algorithm can be further validated using additional industry-based datasets, and by comparing the proposed solution with similar approaches for clustering structures.

V. CONCLUSION

This paper aimed to develop an algorithm for restoring structures within a dataset that is reliably known to have damaged structures. This paper investigated datasets containing tables without metadata, multiple delimiters used inside the datasets, and columns that substitute or change places. The proposed algorithm was developed as a two-phase process, in which the original data tables were recognized in the first phase and the columns of the original data tables in the second phase. The algorithm was developed in Python using existing machine-learning algorithms. To cluster the corresponding structures, the existing machine learning algorithms were modified, that is, they were expanded in terms of using a more significant number of parameters than standard implementations (determining the center of the cluster and the distance from the cluster, machine learning algorithms). Open datasets are used to evaluate the proposed algorithm. These datasets were modified to break their structures and to make them suitable for testing the proposed algorithm. Research questions were posed regarding the performance of the algorithm in recognizing data structures, that is, the recognition of tables and columns of appropriate types.

The number of structures increases with the heterogeneity of the data in the input tables. This experiment demonstrated that, although the input dataset consists of tables of different structures, several tables can have broken structures that are common to several different tables during the process of breaking the structure. The results show that the STCExtract algorithm correctly arranged the structure of the tables with an accuracy of 94.4% to 100%. The accuracy of the STCExtract algorithm in the second phase (when the data were allocated to columns) ranged from 59.7% to 90.2%. The results show that the algorithm is more successful when allocating input data to the tables. Significant deviations were observed in the case of data allocation by columns. This algorithm detects multiple structures, and is an innovative solution that should be compared with similar solutions using additional industry-based datasets.

APPENDIX

Information regarding datasets.

See Table 21.

REFERENCES

[1] K. Kellou-Menouer, N. Kardoulakis, G. Troullinou, Z. Kedad, D. Plexousakis, and H. Kondylakis, "A survey on semantic schema discovery," *VLDB J.*, vol. 31, no. 4, pp. 675–710, Jul. 2022.

[2] F. Bai, J. Kang, G. Stanovsky, D. Freitag, and A. Ritter, "Schema-driven information extraction from heterogeneous tables," 2023, *arXiv:2305.14336*.

[3] M. U. Hassan, K. Shaukat, D. Niu, S. Mahreen, Y. Ma, F. Haider, M. Mubashir, and X. Zhao, "An overview of schema extraction and matching techniques," in *Proc. 2nd IEEE Adv. Inf. Manag., Communicates, Electron. Autom. Control Conf. (IMCEC)*, Xi'an, China, May 2018, pp. 1290–1294.

[4] J. Chen and M. Reformat, "Learning categories from linked open data," in *Proc. Int. Conf. Inf. Process. Manage. Uncertainty Knowl.-Based Syst. (IPMU)*, vol. 444, 2014, pp. 396–405.

[5] E. Rahm and P. A. Bernstein, "A survey of approaches to automatic schema matching," *VLDB J.*, vol. 10, no. 4, pp. 334–350, Dec. 2001.

[6] K. Menouer and Z. Kedad, "Schema discovery in RDF data sources," in *Proc. Conceptual Model. 34th Int. Conf. (ER)*, vol. 9381, 2015, pp. 481–495.

[7] M. Kirchberg, E. Leonardi, Y. Tan, S. Link, R. Ko, and B. Lee, "Formal concept discovery in semantic web data," in *Proc. Int. Conf. Formal Concept Anal. (ICFCA)*, vol. 7278, 2012, pp. 164–179.

[8] P. Li, Y. Gong, and C. Wang, "Schema extraction on semi-structured data," 2021, *arXiv:2012.08105*.

[9] S. Jain, A. de Buitléir, and E. Fallon, "A review of unstructured data analysis and parsing methods," in *Proc. Int. Conf. Emerg. Smart Comput. Informat. (ESCI)*, Mar. 2020, pp. 164–169.

[10] R. Bouhamoum, Z. Kedad, and S. Lopes, "Schema discovery in large web data sources," in *Proc. Int. Conf. Big Data Cyber-Secur. Intell. (BDCSIntell)*, 2022, pp. 1–8.

[11] A. E. Ezugwu, A. M. Ikotun, O. O. Oyelade, L. Abualigah, J. O. Agushaka, C. I. Eke, and A. A. Akinyelu, "A comprehensive survey of clustering algorithms: State-of-the-art machine learning applications, taxonomy, challenges, and future research prospects," *Eng. Appl. Artif. Intell.*, vol. 110, Apr. 2022, Art. no. 104743.

[12] A. S. Shirkhorshidi, S. Aghabozorgi, T. Y. Wah, and T. Herawan, "Big data clustering: A review," in *Proc. Int. Conf. Comput. Sci. Appl. (ICCSA)*, vol. 8583, 2014, pp. 707–720.

[13] H. Dong, S. Liu, S. Han, Z. Fu, and D. Zhang, "TableSense: Spreadsheet table detection with convolutional neural networks," in *Proc. AAAI*, vol. 33, 2019, pp. 69–76.

[14] C. Christodoulakis, E. B. Munson, M. Gabel, A. D. Brown, and R. J. Miller, "Pytheas: Pattern-based table discovery in CSV files," *Proc. VLDB Endowment*, vol. 13, no. 12, pp. 2075–2089, Aug. 2020.

[15] G. J. J. van den Burg, A. Nazábal, and C. Sutton, "Wrangling messy CSV files by detecting row and type patterns," *Data Mining Knowl. Discovery*, vol. 33, no. 6, pp. 1799–1820, Nov. 2019.

[16] L. Jiang, G. Vitagliano, and F. Naumann, "Structure detection in verbose CSV files," in *Proc. Int. Conf. Extending Database Technol. (EDBT)*, 2021, pp. 193–204.

[17] E. Koci, M. Thiele, O. Romero, and W. Lehner, "Table identification and reconstruction in spreadsheets," in *Proc. Int. Conf. Adv. Inf. Syst. Eng.* Cham, Switzerland: Springer, 2017, pp. 527–541.

[18] J. C. Roldán, P. Jiménez, and R. Corchuelo, "On extracting data from tables that are encoded using HTML," *Knowl.-Based Syst.*, vol. 190, Feb. 2020, Art. no. 105157.

[19] G. Yuan, J. Lu, Z. Yan, and S. Wu, "A survey on mapping semi-structured data and graph data to relational data," *ACM Comput. Surveys*, vol. 55, no. 10, pp. 1–38, Oct. 2023.

[20] A. O. Shigarov and A. A. Mikhailov, "Rule-based spreadsheet data transformation from arbitrary to relational tables," *Inf. Syst.*, vol. 71, pp. 123–136, Nov. 2017.

[21] H. Elmeleegy, J. Madhavan, and A. Halevy, "Harvesting relational tables from lists on the web," *Proc. VLDB Endowment*, vol. 2, no. 1, pp. 1078–1089, Aug. 2009.

[22] *The Internet Movie Database (IMDb)*. Accessed: Jul. 9, 2023. [Online]. Available: https://www.imdb.com/interfaces/

[23] S. Stojanović, Z. Radivojević, and M. Cvetanović, "Approach for estimating similarity between procedures in differently compiled binaries," *Inf. Softw. Technol.*, vol. 58, pp. 259–271, Feb. 2015.

[24] *The Game Reviews Dataset*. Accessed: Jul. 9, 2023. [Online]. Available: https://www.kaggle.com/datasets/sridharstreaks/game-reviews-dataset

[25] *The Calendar Database*. Accessed: Jul. 9, 2023. [Online]. Available: https://www.kaggle.com/datasets/uphaardarbari/calender

[26] *Barcelona Traffic Accidents*. Accessed: Jul. 9, 2023. [Online]. Available: https://www.kaggle.com/datasets/emmanuelfwerr/barcelona-car-accidents

[27] *Monthly Electricity Production*. Accessed: Jul. 9, 2023. [Online]. Available: https://www.kaggle.com/datasets/ccanb23/iea-monthly-electricity-statistics

[28] *Port Shipment Dataset*. Accessed: Jul. 9, 2023. [Online]. Available: https://www.kaggle.com/datasets/mikoajfish99/port-of-los-angeles

[29] *Large Movie Dataset*. Accessed: Jul. 9, 2023. [Online]. Available: https://www.kaggle.com/datasets/chaitanyahivlekar/large-movie-dataset

[30] *Every Pub in England*. Accessed: Jul. 9, 2023. [Online]. Available: https://www.kaggle.com/datasets/rtatman/every-pub-in-england?select=open_pubs.csv

[31] *Institutes of Technology Admissions Dataset*. Accessed: Jul. 9, 2023. [Online]. Available: https://www.kaggle.com/datasets/goyaladi/iit-admissions-dataset

[32] A. Jankovic, G. Chaudhary, and F. Goia, "Designing the design of experiments (DOE)—An investigation on the influence of different factorial designs on the characterization of complex systems," *Energy Buildings*, vol. 250, Nov. 2021, Art. no. 111298.

[33] N. M. P. Bianchesi, E. L. Romão, M. F. B. P. Lopes, P. P. Balestrassi, and A. P. De Paiva, "A design of experiments comparative study on clustering methods," *IEEE Access*, vol. 7, pp. 167726–167738, 2019.

[34] J. Antony, "6—Full factorial designs," in *Design of Experiments for Engineers and Scientists*, 2nd ed. Amsterdam, The Netherlands: Elsevier, 2014, pp. 63–85.

[35] D. C. Montgomery, *Design and Analysis of Experiments*, 8th ed. Hoboken, NJ, USA: Wiley, 2012.

BRANISLAVA CVIJETIĆ received the B.Sc. and M.Sc. degrees in automatics and informatics from the University of East Sarajevo, Bosnia and Herzegovina, in 2002 and 2013, respectively. She is currently pursuing the Ph.D. degree with the Department of Computer Engineering and Information Theory, University of Belgrade School of Electrical Engineering, Republic of Serbia. Her research interests include data analyses and machine learning.

ZAHARIJE RADIVOJEVIĆ received the B.Sc., M.Sc., and Ph.D. degrees in computer engineering from the University of Belgrade School of Electrical Engineering, in 2002, 2006, and 2012, respectively. He is currently an Associate Professor with the Department of Computer Engineering and Information Theory, University of Belgrade School of Electrical Engineering, teaching several courses on computer architecture and organization, e-business infrastructure, and mobile device programming. His research interests include computer architecture and organization, concurrent and distributed programming, data analysis, simulations, and reverse engineering.