MDClean: Cleaning and Enhancing Raw Metadata Files

Hiba Khalid
Université Libre De Bruxelles
Brussels, Belgium
Hiba.Khalid@ulb.ac.be

Esteban Zimányi
Université Libre De Bruxelles
Brussels, Belgium
Esteban.Zimanyi@ulb.ac.be

ABSTRACT

The importance of incorporating metadata for data discovery, provenance, resource management, and resource identification is well established. The quality of metadata significantly improves data decisions, inferences, resource reusability, and data maintenance. With the increasing importance of data quality, metadata is becoming more popular in resource management and analysis and provides a solid foundation for monitoring data quality. To achieve better data analysis, it is crucial to either obtain high-quality and representative metadata or to improve the quality of existing metadata as much as possible. To improve the quality of existing metadata, this paper discusses and introduces “sanity checks” that are quality operators for cleaning and enhancing existing metadata in metadata files. Each quality operator targets a specific problem in a metadata file and tries to improve it according to its definition.

KEYWORDS

Metadata Quality, Information Retrieval, Metadata Challenges, Data Provenance, Information Organization, Resource Management

1 INTRODUCTION

Our research explores the role of metadata in various fields, such as searching data in web repositories [1], schema matching [12], data integration [7], and exploring data lakes [9]. However, the metadata available in online repositories is usually of poor quality and scattered, leading to less than optimal use. To address this problem, we define sanity checks called "quality operators," which attempt to improve metadata quality in metadata files. The research identifies three categories of metadata files based on their raw formats: (1) Low-quality files, (2) Good quality files, and (3) No metadata files. This paper focuses on low-quality metadata. In this research paper’s context, we focus on the first category, i.e., low-quality metadata. Our research investigates how to improve metadata quality and addresses specific quality issues in metadata management. The goal of attaining metadata quality is to ensure accurate and timely information for reusability and comprehensibility [10, 14]. The metadata collection process is messy and involves collection, extraction, inference, and profiling to obtain high-quality metadata [11]. Low-quality metadata files contain incomplete, incomprehensible, unorganized, and often incorrect information that can lead to misinterpretation in data analysis and other data-related tasks.

To better understand this, we surveyed to observe patterns and the most frequent discrepancies in metadata files. We crawled 1578 metadata project files from the UKGov[1] to investigate potential metadata quality issues. We then manually inspected each project file and provided labels. During our manual annotation process, we encountered several inconsistencies in the crawled project files. Based on the label occurrence, we then select the following nine inconsistencies (every second file has these inconsistencies) and try to improve them using quality operators. Data file parsing details missing (file dialect), empty cells, and empty blocks to serve human readability or due to missing values, missing value’s property, e.g., email, file path, etc., missing property’s value, e.g., email address, URL containing file path, etc., no domain information, generate attribute list, inconclusive file names, and missing directory information. Similar to UKGov[1], we observe the same problems with the majority of files in our other datasets (Kaggle[2], DataGov[3]). In Section 2, we describe these problems and relevant quality operators in more detail. Table 1 lists the set of quality operators, where each quality operator attempts to improve the aforementioned quality problems.

Please note that our system expects a spreadsheet as a metadata file. Hence, as a pre-processing step, we convert other file formats, such as comma-separated files (CSV), JSON, XML, and TXT, to XLXS to take advantage of the stylistic peculiarities of this format and optimize the grid-based view for scanning and enhancing metadata. It would be interesting to extend our approach to other file formats, such as CSV, JSON, XML, etc., in the future.

This paper makes the following contributions:

- A set of quality operators for metadata cleaning.
- A system MDClean, to automatically clean and enhance available metadata.
- A wide range of experiments to validate MDClean and its applicability.

The rest of the paper is organized as follows: Section 2 illustrates the workflow of the MDClean and explains its main modules. Section 3 presents the experimental evaluation of MDClean. Section 4 discusses related work in the field of data cleaning for diverse domains, and Section 5 concludes our study.

2 THE MDCLEAN PROCESS

First this section defines and differentiates between sanitized and non-sanitised metadata and describes the problem statement.Then, this section describes the quality operators and their scope. Followed by the workflow of MDClean as shown in Figure1. MDClean takes a non-sanitized metadata file along with a data file as input and outputs the sanitized metadata file along with a meta metadata file.

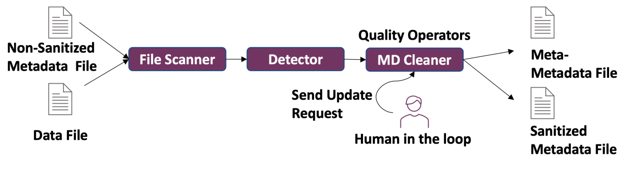

Figure 1: The workflow of MDClean

[1]https://data.gov.uk/

[2]https://www.kaggle.com/
[3]https://www.data.gov/

2.1 Non-Sanitized and Sanitized Metadata

Definition 2.1 (Non-Sanitized Metadata). Metadata with inconsistencies, such as missing properties, incorrect references, empty blocks, which can be improved by applying quality operators.

Definition 2.2 (Sanitized Metadata). After applying quality operators, the metadata we get contains a correctly computed attribute list, enhanced metadata values, their properties, and information about the origin and parsing of the linked data file.

To sanitize metadata, we address two sub-problems. First, we detect incorrect values or erroneous values, or missing values that are in the metadata file but not near its counterpart, e.g., `property{email} : value{bob@example.com}`, and try to remove them in case of empty or impute the property or value near the counterpart to clean metadata. Second, we add additional information to the metadata file, e.g., missing parsing details for the data file or missing attribute list information.

We now formally define our problem as follows: *Given an input metadata file with non-sanitized metadata, systematically apply quality operators and produce an output metadata file containing sanitized metadata.*

2.2 Scanning and Detection

This section discusses the details of the scanning and detection module that supply the processed metadata file for quality operators. The scanning and detection module of our quality operators' system operates hand in hand with each other. The expected outcome of the scanning and detection module is finding the file layout by scanning and detecting file content.

2.2.1 *Scanning Metadata File.*

The scanning module recognizes data types in metadata files by generating a color-coded image of cell contents. This helps identify what is available in the file before cleaning. Each data type is assigned a color, such as strings as purple and numbers as green. Annotated metadata cells are then sent to the detection module.

2.2.2 *Detecting Metadata File Content.*

The detection module assigns labels to strings based on semantics on top of scanning labels, such as *description* or *email*. It also improves the annotation of empty cells from the scanning module. For example, it looks for *property* and *property value* and changes the empty annotation to a missing value annotation if one of the values from the pair is missing.

2.3 Quality Operators

Table 1 lists the quality operators we introduced in this research. A quality operator is a method that receives a detected metadata inconsistency as input, inserts new values or updates the given inconsistent values, and outputs a set of values that are of higher quality or more useful. Cleverly selecting and applying quality operators on metadata can improve the latter to a cleaner state. The selected quality operators are meaningful for the datasets we annotated and can be easily applied to other domains due to their generalizability. Furthermore, we divide our quality operators into two categories, (1) metadata cleaners and (2) metadata enhancers. Metadata cleaner checks improve the metadata by cleaning the value or removing redundant elements such as empty values or empty blocks. In contrast, metadata enhancers add value to existing metadata by inserting necessary details, e.g., inserting domain inference or adding a missing attribute list.

2.3.1 *Dialect Details.*

CSV file dialect detection is a well-known problem in the industry[4,5,6] and academia [2, 15]. Since our research task is not a parsing problem, we use the state-of-the-art parsers [4] [2, 15] in dialect detectors for this quality operator implementation. The dialect detection quality operator has primarily two functions, the first is to search for the label that indicates information about the file dialect within the metadata file, and we get this label from the detection module. If no label is detected by the detection module, it is assumed that the dialect information for passing the data file is missing. Then, the second task is to infer the dialect information using an ensemble method and add the details about the file dialect to the sanitized metadata file. As mentioned before, we make use of state-of-the-art dialect detectors for the dialect ensemble (see Figure 2). Since we have different data file formats, as a preprocessing step, if a data file is not CSV, the module first converts the file to the extension (.csv) and then sends it to these parsers since the parsers expect a CSV file as an input. The ensemble selects the dialect information by passing the data file to each parser and selecting the most popular of them all (majority vote ensemble). The ensemble returns the file dialect details, which are then inserted into the metadata file.

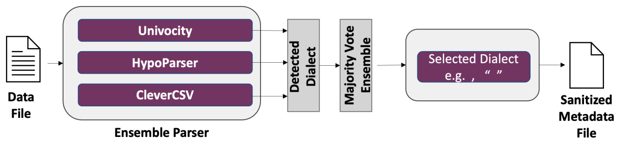

Figure 2: Dialect detection ensemble

2.3.2 *Trim Whitespace & Empty Blocks.*

the whitespace trimmer is a quality operator that detects and removes irregular and consecutive whitespace from metadata content. It scans for whitespace in different parts of the metadata, including neighboring cells, to avoid removing important information. The trimmer uses regex and defines classes for every operation it takes. It also considers neighboring cells and evaluates if they are continuous, interrupted, or span over multiple rows and columns. If a cell is next to a detected metadata element, it is labeled as a missing value rather than an empty cell.

2.3.3 *Value's Property Imputation.*

the metadata files in our dataset are often empty or contain scattered metadata. Our metadata element imputation targets files with dispersed data and aims to improve properties and values hand in hand. We detect metadata properties and identify cells with valuable values but no corresponding properties. Our quality operator scans file labels for value's property and property's value tags and suggests missing property names. We use KeyBERT, dictionaries, and a link-text extractor to create a meaningful title, header, and subject for the metadata value's property. Finally, the sanitized metadata file is appended with meta information.

2.3.4 *Property's Value Imputation.*

imputing missing values in metadata files is a complex task. Traditional methods, such as calculating column means, do not work for metadata. Two approaches are used in our research: property-targeted imputation and manual imputation. In property-targeted imputation, the system searches for relevant values related to the property for which

[4]https://www.univocity.com
[5]https://www.trifacta.com
[6]https://www.tableau.com/products/prep

#	Quality Operators	Category	Description
1	Dialect Details	Enhancer	Detects and adds information about the file's dialect
2	Trim Whitespace & Empty Blocks	Cleaner	Detects and removes extra whitespace
3	Value's Property Imputation	Cleaner	Detects and adds property for a value
4	Property's Value Imputation	Cleaner	Detects and adds value for a property
5	Domain Inference	Enhancer	Infer domain from the given textual description
6	Generate Attribute List	Enhancer	Extracts and adds the attribute information
7	Assign Meaningful File Name	Cleaner	Updates inconclusive file names
8	Self Directory Inference	Enhancer	Includes self directory for provenance
9	Other Directory Inference	Enhancer	Includes data file(s) directory for provenance

Table 1: Quality operators, their categories and description

a value is missing, and then evaluates all metadata elements with complete values. The most relevant value is mapped against the property based on relevancy scores calculated using keyword analysis, data type information, and categorical transformation. In manual imputation, the user adds metadata values against the corresponding metadata element. The system adds a text label manually for user readability next to the metadata value cell.

2.3.5 Domain Inference. our research proposes a quality operator that uses natural language processing techniques to search metadata files for dataset descriptions and other relevant information. The operator operates on three use cases, identifying and pre-processing potential resource description cells, extracting keywords, analyzing them using n-gram combinations, and assigning domain topics using the Zero-Shot Method. Case I deals with metadata files containing descriptive text, footnotes, or comments. Case II processes mixed descriptions containing text, numbers, and hyperlinks. Case III allows users to add domain information manually. The pre-processing pipeline maintains and records all changes made to the text, ensuring the comprehensibility and readability of the metadata files.

2.3.6 Generate Attribute List. to handle metadata files that lack attribute lists for datasets, our system uses relevant tags to identify contenders for a given property during the detection phase. In attribute list concatenation, we first identify a property that describes or lists the attributes in a spreadsheet cell that might be misplaced or dispersed. Then, the quality operator attempts to check the attribute list in three cases, including adding a new property and appending a list of attributes extracted from the data file, searching for dispersed metadata values that could be a list of attributes, and appending the attribute list extracted from the data file when there is no potential list within the metadata file. The quality operator uses a string similarity check and threshold of 60% to conduct a transformation cycle for cases where lists are not an exact match.

2.3.7 Inconclusive File Names. it is necessary to have meaningful file names because it is not only easy for humans to read and understand what the file is about. Our system extracts inconclusive or non-representative metadata file names and updates them with meaningful file names. It updates metadata file names with a metadata prefix-domain inferred title - -date-file extension suffix. This quality operator depends on the domain inference check since file names should be representative and meaningful for the reader.

2.3.8 Self Directory Inference. data scientists and machine learning engineers often search data from open-source repositories to test their experiments. During this process, if the link to the file source is missing, problems arise in tracking down the file or citing the file source for copyright issues. This quality operator ensures the source link is not lost during metadata collection. When the metadata files are crawled, the system downloads the metadata and data files and the web link information to get the source information. The downloaded information is appended to the metadata file as meta-meta data along with the retrieval date, web link, time, type, and size of the metadata file. If the download was bulk, i.e., more than one metadata file, then a unique metadata file ID is added to meta-meta data in the file. For the files that have already been downloaded without source links, this quality operator cannot find the relevant information and prompts the user to manually enter the path for the downloaded file.

2.3.9 Other Directory Inference. this is similar to self-directory inference but for the data files. This quality operator maintains the information about the data files that were crawled with their web source link, name, title, file size, file type, domain text (if available), etc. Each metadata file could have one or more associated data files. In the case of one metadata file representing more than one data file, we assign a unique metadata file ID and append it to the directory information for data files. In this way, for a set of files, the same metadata information can be reused for accessibility, pre-processing, and analysis. Similar to the metadata file, if the data files that have already been downloaded without source links, this quality operator cannot find the relevant information and prompts the user to enter the path for the downloaded file manually.

3 ANALYSIS AND RESULTS

This section first presents the datasets we used, their description, the necessary preprocessing steps, and the annotation process. We then, in the following, evaluate the performance of our system, (1) Evaluation queries, their scope, and performance on metadata file. (2)Effects of quality operators on different file formats. (3) Performance evaluation of quality operators. (4) quality operator applicability on non-sanitized metadata files.

3.1 Datasets

To evaluate the performance of our system, we crawled datasets from three open data repositories, namely: Kaggle (www.kaggle.com), UKGov (www.data.gov.uk), and DataGov (www.data.gov), and randomly selected a total of 1123 metadata files for our experiments. Table 2 lists the datasets we used for our experiments. To evaluate our system, we created a ground truth by annotating every file in our collection. The annotation process was divided into three sections: scanning, content detection, and cleaning. For the scanning module, we manually labeled the layout based on

Dataset Source	Total Files	Pre-Processing	Misplaced Metadata	MD Accessibility
Kaggle	375	91%	Yes, mostly	dispersed, multiple files (not in one place)
DataGov	374	35%	Partially	metadata inside files
UKGov	374	84%	Yes, mostly	dispersed, multiple files (not in one place)

Table 2: The Table lists the datasets we used to develop and evaluate MDCLEAN, and provides insights related to the amount in percentage for metadata pre-processing, misplaced metadata, and metadata accessibility.

data types, while for the detection module, we annotated semantics on top of the layout. Finally, we annotated the cell values for the cleaning module and inserted details such as dialect, attribute list, and domain inference, while leaving missing property values empty to avoid filling in false information.

3.2 Evaluation Queries

We developed evaluation queries to assess the quality operators for metadata cleaning, comparing the results for sanitized and non-sanitized metadata files. The queries assess the state of information available in the files and are listed in Table 3. Figure 3 shows the average query performance for both types of files. The first query retrieves and compares dialect details, where non-sanitized files perform better due to missing parsing details in sanitized files. The third query compares irregular whitespace blocks, where non-sanitized files have more blocks than sanitized files. Overall, the query performance is better for sanitized files generated by our system than for non-sanitized files. In cases where the query cannot find information, it is because our quality operator could not find the correct details, and the parser cannot parse the input files accurately.

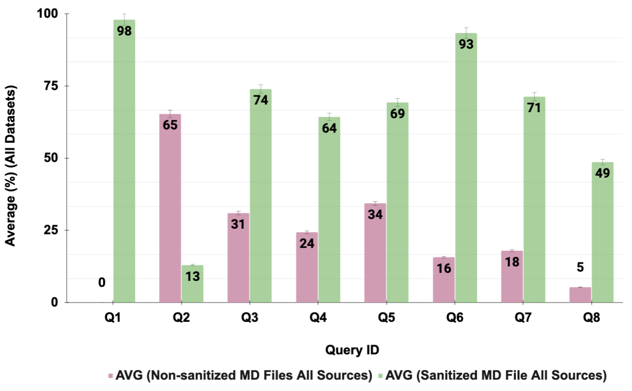

Figure 3: Evaluation queries performance on sanitized and non-sanitized metadata files

3.3 Quality Operators on Different File Format

As mentioned above, we used data from different sources, and each source has different file formats. To develop MDCLEAN, we use TXT, CSV, and XLSX file formats and convert TXT and CSV file formats to XLSX. This experiment aimed to find out how information from different file formats affects the applicability of quality operators. We observed several aspects; the most important is the amount and scope of metadata present in a file. It is essential to understand that if metadata is present in a file, quality operators can improve the readability and understanding of the file. If this is not the case, our system can also insert properties based on the underlying issues (see Figure 4).

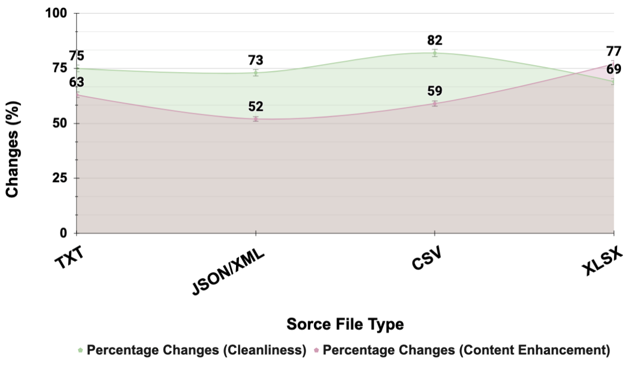

Figure 4: Percentage changes by file format

3.4 Performance Evaluation of Quality Operators

We used the manually annotated ground truth to evaluate the performance of quality operators. Each metadata file is cleaned based on two essential prerequisites: either the information was present in the non-sanitized metadata file, or the information could be derived based on quality operators. Figure 5 shows the average accuracy score of quality operators across all files on all three Kaggle, UKGov, and DataGov datasets.

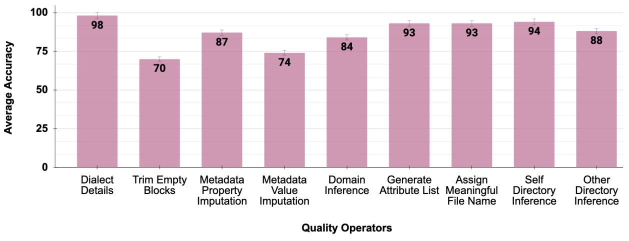

Figure 5: Quality operator's average accuracy score across all files on all datasets

Non-sanitized metadata files of our datasets were missing the *dialect details* of the data files. Our dialect detail quality operator evaluated each file and used ensemble methods to extract the dialect information and append it to the metadata file. The average accuracy across all files of all datasets of our quality operator is 98%. The few cases where our approach does not determine the correct dialect are due to the scope of our parser that we use when parsing the CSV files to determine the dialect details. The parsers worked perfectly for the Kaggle and DataGov files. For UKGov, on the other hand, some of the files were messy and contained multiple tables or broken dialects [8] that our parsers could not repair and parse.

Deriving and adding properties and values to a metadata file improves file quality, processing, and readability. The *imputation checks* are also notable in their performance, with an average accuracy of over 80%. These checks carefully inferred missing properties from values. While for missing property values, the quality operator looks across the file and tries to impute the correct value. However, value imputation imposes its challenges, and to overcome this; we experimented with human-in-the-loop

Query-Id	Query Title	Query Description
QR-01	Check Dialect Details	Retrieve dialect details from sanitized and non-sanitized metadata files
QR-02	Check Empty Blocks	Compare total empty blocks between sanitized and non-sanitized metadata files
QR-03	Check Value Property	Count value properties in sanitized and non-sanitized files
QR-04	Check Property Value	Count property values in sanitized and non-sanitized files
QR-05	Check Domain Inference	Look for domain inference in sanitized and non-sanitized files
QR-06	Check Attribute List	Look for attribute list in sanitized and non-sanitized files
QR-07	Check Data File Source	Fetch source file details from sanitized and non-sanitized files
QR-08	Check Metadata File Source	Fetch metadata source file details from sanitized and non-sanitized files

Table 3: Queries and their description to evaluate the impact of quality operators on metadata files

operations to assist our system with domain expertise for filling and updating wrongly interpreted property values.

Another challenging use case for our research was *domain inference*. Non-sanitized metadata files contained information about the domain or subject of the dataset, but none of this information was precisely referenced. It had to be inferred or extracted from the available information in the Non-sanitized metadata files. In this case, the domain inference quality operator yielded, on average, accurate domains across all files for 84% of the files. This accuracy is validated by manual expert comments for datasets on the subject in conjunction with scraped web information from hosting websites. Apparently, it is a trivial task *trimming whitespace blocks*, but it imposes challenges when the whitespace is spanned to multiple columns. The missing `property:value` pairs cause ambiguity in most cases. Consequently, *empty block trimmer* takes the wrong decision, and thus, the score for this quality operator is low compared to other checks.

4 RELATED WORK

To the best of our knowledge, there is no directly related work on cleaning up metadata files. However, we have collected some notable research contributions that focus on understanding the metadata file layout and content of metadata files for metadata management.

Khalid et al. used rule agents and conducted a comprehensive study on how disorganized open-source metadata files are. In addition, the authors annotated many files and showed that agents work well with the annotated files due to the cleaned information and better structural layout [4]. Smutz et al. detected malicious PDF files based on metadata and structural features [13]. The authors presented how metadata elements can influence such decisions by carefully arranging metadata in files. Liu et al. presented automatic metadata extraction from documents using deep networks [5]. The authors focused on extracting metadata from academic papers, which can be helpful for scientific searches and digital libraries. Ha et al. introduced the OCRMiner system, which uses the layout features of structured documents by applying optical character recognition (OCR) and extracting the indexing metadata [3]. Tabular presentation of information is wildly used in the scientific community. However, searching for tables online takes much work. Liu et al. presented TableSeer, which searches digital libraries and recognizes tables in documents by extracting table metadata [6].

5 CONCLUSION

This paper introduces MDClean, a system that automatically detects and corrects metadata content in unorganized and non-sanitized metadata files, using quality operators as quality indicators to target specific issues in the files. The system is evaluated using metadata files from three open data repositories, with 1123 files randomly selected for experimentation and annotations used as ground truth for performance evaluation. Future work will address semantic challenges to support further automating metadata processes for resource management and enhance the system's usability through a graphical user interface. In summary, MDCleanis an automated system that improves the management and organization of metadata content in large-scale data systems.

REFERENCES

[1] Dan Brickley, Matthew Burgess, and Natasha Noy. Google dataset search: Building a search engine for datasets in an open web ecosystem. In *The World Wide Web Conference*, pages 1365–1375, 2019.

[2] Till Döhmen, Hannes Mühleisen, and Peter Boncz. Multi-hypothesis csv parsing. In *Proceedings of the 29th International Conference on Scientific and Statistical Database Management*, pages 1–12, 2017.

[3] Hien T Ha, Marek Medved, Zuzana Nevěřilová, and Aleš Horák. Recognition of ocr invoice metadata block types. In *International Conference on Text, Speech, and Dialogue*, pages 304–312. Springer, 2018.

[4] Hiba Khalid and Esteban Zimányi. Mdorg: Annotation assisted rule agents for metadata files. In *DOLAP*, pages 81–90, 2021.

[5] Runtao Liu, Liangcai Gao, Dong An, Zhuoren Jiang, and Zhi Tang. Automatic document metadata extraction based on deep networks. In *National CCF Conference on Natural Language Processing and Chinese Computing*, pages 305–317. Springer, 2017.

[6] Ying Liu, Kun Bai, Prasenjit Mitra, and C Lee Giles. Tableseer: automatic table metadata extraction and searching in digital libraries. In *Proceedings of the 7th ACM/IEEE-CS joint conference on Digital libraries*, pages 91–100, 2007.

[7] Renée J Miller. Open data integration. *Proceedings of the VLDB Endowment*, 11(12):2130–2139, 2018.

[8] Johann Mitlöhner, Sebastian Neumaier, Jürgen Umbrich, and Axel Polleres. Characteristics of open data csv files. In *2016 2nd International Conference on Open and Big Data (OBD)*, pages 72–79. IEEE, 2016.

[9] Fatemeh Nargesian, Erkang Zhu, Renée J Miller, Ken Q Pu, and Patricia C Arocena. Data lake management: challenges and opportunities. *Proceedings of the VLDB Endowment*, 12(12):1986–1989, 2019.

[10] Xavier Ochoa and Erik Duval. Automatic evaluation of metadata quality in digital repositories. *International journal on digital libraries*, 10(2-3):67–91, 2009.

[11] Thorsten Papenbrock, Tanja Bergmann, Moritz Finke, Jakob Zwiener, and Felix Naumann. Data profiling with metanome. volume 8, pages 1860–1863. VLDB Endowment, 2015.

[12] Erhard Rahm. Towards large-scale schema and ontology matching. In *Schema matching and mapping*, pages 3–27. Springer, 2011.

[13] Charles Smutz and Angelos Stavrou. Malicious pdf detection using metadata and structural features. In *Proceedings of the 28th annual computer security applications conference*, pages 239–248, 2012.

[14] Divesh Srivastava, Monica Scannapieco, and Thomas C Redman. Ensuring high-quality private data for responsible data science: vision and challenges. *Journal of Data and Information Quality (JDIQ)*, 11(1):1–9, 2019.

[15] Gerrit JJ van den Burg, Alfredo Nazábal, and Charles Sutton. Wrangling messy csv files by detecting row and type patterns. *Data Mining and Knowledge Discovery*, 33(6):1799–1820, 2019.