MDORG: Annotation Assisted Rule Agents for 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

Metadata files are often incomplete and of inadequate quality with underlying issues such as low maintenance, lack of provenance, inaccurate tagging, limited or no annotation and, different metadata standards. Such metadata is not generally viable for analysis, large scale integration, data management, or quality metadata maintenance. These types of metadata present challenges in data analysis and production tasks. The expense associated with metadata discovery, metadata annotation, and management amounts to the time and effort of resources. To overcome the aforementioned issues, leverage can be borrowed from intelligent systems that can improve quality and reduce the manual effort associated with metadata discovery, annotation, and management. Intelligent agents can aid in identifying information if they are provided relevant labels or agents are allowed to learn over-time. Thus, the annotation for intelligent agents can improve tasks such as automated summaries, metadata management, cataloging and, feedback management. We experiment and propose the utility of annotation for rule agents to facilitate the analysis and organization of metadata files.

KEYWORDS

rule-based agents, textual metadata, metadata, intelligent agents, metadata management, metadata representation, metadata categorization

CEUR-WS.org/Vol-2840/paper12.pdf

1 INTRODUCTION

In simplistic words, metadata [23] defines data, data collection and data recording process, and details about what the data entails. The importance of metadata lies in its inherent quality of maintaining the information about data. Generally, there are three possibilities when dealing with metadata (i) Case-A: when there is no metadata available, and data profiling [24] is required to infer metadata. (ii) Case-B: when there is little to no metadata available, in such cases, mostly some sort of metadata such as creation date and publisher titles are available. (iii) Case-C: when the metadata is available, however, of inadequate quality, i.e., the metadata content is either incorrect or not recorded properly. In this case, the challenge is to extract meaning from the available metadata. There is no speculation or doubt when it comes to metadata usability [13]. The process, however, is non-deterministic and often very resource and time expensive. A beneficial understanding amongst the scientific community around the topic of publicly available metadata can yield several advantages. Provided so, the more the available metadata, the easier it would be to retrieve datasets, suggest and recommend datasets, and pre-analyze various data sources. A similar advantage is evident for data lakes [20], and data warehousing [9], where data privacy can be a prominent concern, and metadata can significantly improve data manipulation, data fetch requests, and data analysis [32].

In the context, of attaining or producing metadata, the possibilities are endless but are both resource and time expensive. Thus, there is a need for systems that can help generate, understand, extract, and populate metadata. This task is not simplistic and is multi-phase, which requires attention to intricate details such as cross-platform data sharing, data rights, data protection management, and data policies. There is still significant potential to support and exercise the quality production of metadata, and its management. We explore the opportunity of leveraging ML techniques to (i) support metadata systems and, (ii) to explore and categorize information in metadata files using rule agents to understand the information available in metadata files. Based on policies, rule agents guide the process of identifying metadata types and organize the information at hand for better use.

1.1 Metadata Types

Metadata types include: (1) Descriptive Metadata: It [33] describes a dataset or resource for discovery, profiling and identification. For example, title, authors, keywords, abstract, and comments. (2) Structural Metadata: It [12] defines the type, structure, and data relationships. This type of metadata highlights the form and organization of data under consideration. It can include the order in which information is available as well as; its original recording format. The most popular examples include information schema [2] and definition schema. (3) Administrative Metadata: As the name suggests, this type of metadata contains information that can help manage and update a source. It typically includes information such as date of creation, published date, update dates, file type, logs, file definition, access management, and access rights. Most commonly used subsets of administrative metadata include (i) Rights management metadata: this sub-category deals with data rights, intellectual property, and sharing policies. (ii) Preservation metadata [17]: includes the information used to archive resources, update and quality management, and resource preservation.

In the context of this research paper, we will be discussing the (i) usability of rule agents in understanding metadata files, (ii) define annotations on metadata files (for example, metadata file headers, metadata group headers), (iii) to observe rule agents using annotated MD files that could facilitate metadata file organization that includes identifying metadata types, content, dispersed and misplaced metadata. The paper is organized as follows: section 2 discuss the use of rule agents for different applications and use cases, section 3 discusses the working and design of intelligent agents, section 3.1 defines the need for rule agents and how they can be incorporated to support and possibly facilitate in metadata organization and categorization. Finally, we discuss the importance of using annotated files for rule agents in section 4. Section 5 provides insights on some of the challenges we encountered, and section 6 provides an overview and conclusion of our research paper. Our design and experiment include metadata files extracted from the Kaggle repository and Data.gov metadata

© Copyright 2021 for this paper by its authors.
Use permitted under Creative Commons License Attribution 4.0 International (CC BY 4.0)..

files. The examples described in this paper are derived from the UK Road Safety dataset on Kaggle[1].

2 RELATED WORK

This section reviews the most resonating relevant work in the domain of understanding metadata and the role of AI agents.

Intelligent Agents and Metadata: AIMMX[30] provides a library-based metadata model extractor from software repositories, it performs important tasks such as identifying associate resources, model extraction, and model names. In particular, metadata supportive systems are advancing and taking support from machine learning and AI to build better systems such as IBM Redbooks[26] that among other functions uses metadata from imagery for specifications. Thereby the literature gathers inspiration from rule and knowledge systems to address concerns of semi-automatic categorization and organization of metadata. Rules can have a unique impact on the type of system and available architectures, they not only affect individual agent architectures but multi-agent systems also [5]. The applicability of rule agents extends to semantic web technologies and architectures [7]. A general architecture for rule agents is devised and materialized using semantic web languages. Rule-based systems are diversely applicable, it tends to facilitate systems in taking crucial decisions such as intrusion detection [15]. Rules for intrusion detection like other rule systems have to be pre-defined to identify and differentiate between incoming packets on a network. Testing these systems also require a set of performance and evaluation criterion's that have to be predefined to mark an agents or systems performance. Another important use case for rule-based agents is Grid applications [18]. Rule-based agents perform adequately for high scale and data-intensive solutions. Contrary to belief, these applications not only perform well but are highly advisable for autonomic applications; that allows rule agents to operate on the set of the rule designed for autonomous systems [4]. The application and use of rule agents for different application systems are apparent and understandable. Based on evidence and performance of rule agents, they are deployed in various applications and systems.

Metadata Management and Profiling: Metadata management has taken many forms with time, it is important to understand how metadata has evolved over the years in terms of applications and tools. In general metadata [27] and metadata management [11] [16] are important components of data warehousing and large scale data integration systems. Amongst many challenges in the field of metadata, standards [28] [19] [8], data profiling [21], and format normalization is one of the most prominent concerns. Moreover, the need to maintain and provide reproducible metadata in terms of lineage and provenance is one of the most promising aspects of metadata in semantic web [3].

3 INTELLIGENT AGENTS

Intelligent agents are capable of performing tasks that are instructed to them. The intuition behind introducing intelligent agents for metadata file understanding and categorization stems from the idea of eliminating or reducing the manual effort required in cleaning, processing, and organizing metadata files. We make use of rule agents for assisting in the process of metadata tagging [25], metadata annotations, and metadata file organization. Rule agents, or more commonly known as policy agents, function on a set of pre-defined policies to perform a task. However, this is not simplistic metadata management, and understanding for intelligent agents poses many challenges such as (1) lack of metadata availability, (2) poorly orchestrated metadata, (3) lack of governance [14], (4) bad quality metadata and, (5) lack of metadata standards [28]. The rule agent's design aims to semi-automatically analyze available metadata in files and perform actions such as re-arranging, assigning useful tags, designating metadata types, and metadata organization for profiles. The rule sets for different groups of agents were distinct in nature to observe how comprehensive it was for agents to deal with different types of metadata and information available in the metadata files.

3.1 Rule Agents

Rule agents [22] are simple condition action agents that operate on a provided environment by using the available rules and completing the assigned task. Each rule agent accesses the rules from the rule library and uses its allowed set of actions to analyze and categorize the files. The rule library comprises rule sets that contain multiple rules for a particular category. The rule library contains many rules that correspond to different uses case in metadata file categorization. Each rule set can contain two or more than two policies. Rule agents use the available rules to organize and categorize information inside a metadata file. The rule library contains all rules that are accessible by the rule agent [6] see table 1. Each rule set in table 1 comprises subset rules that satisfy the main aspect of that ruleset. For example, ruleset R_1002 i.e, "count columns", contains a total of 4 rules that (1)identifies columns, (2) column names, (3) keyword matching, and (4) counts the total column names that appear in a metadata file. Similarly, other rulesets contain appropriate rules to identify and target the main purpose of the ruleset. Some rule sets don't contain a lot of rules and are designed more simplistically. For example, the rule set R_1018 contains only one rule i.e. to search for empty line annotation. In our problem design, we work with three libraries, the rule library (contains rule sets), the file library(contains metadata collection), and the agent library(contains rule agents). The metadata file organization and content identification are obtained as a result of policy application using rule agents on collected metadata files.

Rule-based systems are designed for knowledge storage, knowledge manipulation, and informed feedback from the system. Rules are designed as a part of the rule-based system. A rule-based system first gathers the knowledge and then manipulates this knowledge to derive useful information, conclusions, or set of actions. Rule-based systems are called expert systems; that can understand, store, and derive inference from available information. In the context of our research goals, we intend to identify different types of the available metadata, comprehensively deduce metadata information, and categorize metadata content properly. Thus, we work with the most authentic and traditional type of rule-based system called Deduction expert systems (DES) [10]. The DES works with a domain-specific knowledge base and rules to produce (1) deductions and (2) actions based on choices. The choices can be curated or manually programmed into the expert system. In our case, we deal with both types of choices that can render multiple actions. We group types of agent actions as 'Action Set' to centralize the different types of agent actions a rule agent can perform or attain. For an expert system to function end to end, it has to be intuitively designed according to the domain problem. However, the basic component of all expert systems

[1] https://www.kaggle.com/tsiaras/uk-road-safety-accidents-and-vehicles

comprises the following: (i)Knowledge Acquisition: The process of collecting knowledge on a domain, defining methods, boundaries, outliers, and special cases, etc. (ii)Rule Base or Knowledge Base: this comprises a collection, list or set of indicative rules to render action and choices., (iii)Inference Engine: this engine is responsible for deducing or understanding the rules listed.

We designed a multi-agent system for metadata categorization, file understanding, and organization. However, in the context and scope of this research paper, we will be discussing the working and performance of 5 rule agents and omit the rest of the agents due to space limitations. The agent continues to apply rules until the metadata file is completely analyzed see figure 1. Each agent is randomly assigned a metadata file from the mixed collection. Once each agent has its file, a number of actions are performed by each agent to understand the file and its content. For example, in figure 2 we illustrate how agent RA_101 first identifies the prominent file features that are expected to be in the file such as headers, group headers, empty lines, and metadata content. After this step, each agent takes on the metadata contents identified and starts applying rules by accessing the rulesets from the rule library. Each rule application involves the use of allowed agent actions such as read, write, match, drop, delete, and annotation request. An agent request for annotation if there is no associated rule pre-defined in the rule set, or it requests for annotation if the item cannot be categorized, i.e. when an agent is indecisive about an annotation and its associated action. In figure 4, there are five types of information for a rule agent to identify in this use case. Selective information is listed below:

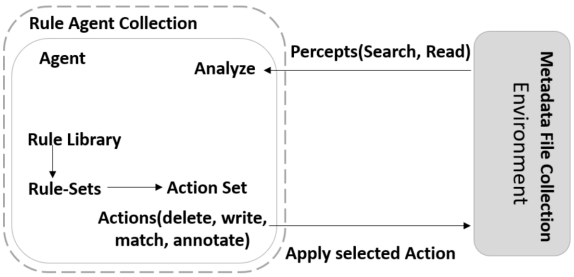

Figure 1: The overall depiction of how agents interact with metadata file collection and uses percepts and actions to analyze metadata files.

3.1.1 Metadata Group Headers: the annotated file in figure 4 is assigned to a policy agent. The policy or rule-based agent utilizes the annotation to identify the group header and looks for data inside each group header. The group headers in figure 4 are 'Usage Information','Maintainers' and, 'Updates'. Policies are designed to equip agents in providing accurate results.

3.1.2 Metadata Content: the second type of information a policy agent encounters is 'metadata'. This is regarded as data in our problem or more precisely as value, and this is not annotated. Thus, the policy agent identifies a group header, locates data, and tags it accordingly, for example, 'data: Group Header_Name'. For example, the first bulk of data would be stored as 'data[License: Database: Open Database, Contents: Database Contents, Visibility: Public]: Usage Information'. Now, this information is available for all policies to explore and exploit. The data inside group headers will be utilized to categorize the type of metadata or information available in the next phases. Most of the annotation

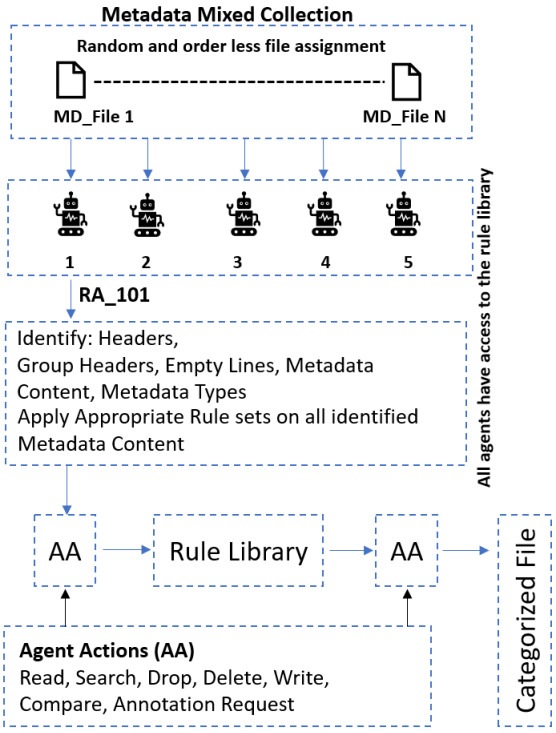

Figure 2: The overall depiction of how agents interact with metadata file collection to analyze metadata files.

in metadata files falls under the metadata content category. As, most of the information besides headers, group headers, and empty lines is metadata content.

3.1.3 Empty Lines: the third type of information a policy agent encounters for this metadata file shown in figure 4 is empty lines. The empty lines are annotated as 'empty lines' for policy agents to quickly differentiate between data, empty lines, and empty values.

Rule-based agents typically function on a set of pre-programmed policies or a set of rules [29]. These are basic instruction set for an agent to use and perform actions. In the case of metadata categorization, organization, and understanding, rule based agents are promising. However, autonomy and self-reliance of decisions are difficult to achieve even for intelligent agents. Annotation is a technique that allows the rule agents to (i) make sense of the information (ii) facilitate future autonomous actions. In our research, we use manual annotation and tagging to observe how intelligent agents perform and categorize metadata information provided to them. To facilitate and provide an understanding of the problem to rule agents, we established annotation criteria for rule agents, figure 4 demonstrates the application of established annotations from table 2 highlights the annotations provided to rule agents for different types of metadata files.

For manual annotation, we group the basic contents of a metadata file that an intelligent agent can come across for a certain file type(e.g., text, CSV, TSV, JSON). After pre-processing of different file types into spreadsheets the annotation process is further

RuleSet ID	Rule Identifier	Rule Description
R_1001	Data Type Identifier	Identifies the type of data encountered as text, numeric, date type etc.
R_1002	Count Columns	It searches the file for total number of columns. It uses keywords and counts if column names appear in textual descriptions.
R_1003	Identify Attribute Names	It iterates to find attribute names, if avaible in file creates a list.
R_1004	Generate Attribute Lists	It generates a list of attributes to be inserted into metadata file.
R_1005	Look Up: Rows	It iterates over textual metadata using headers and group headers to find 'total rows' in dataset.
R_1006	Look Up: Missing Values	It looks for null value indicators, and identifies if headers or metadata lacks value counterparts.
R_1007	Look Up: Date of creation	It identifies and looks for date of creation in a metadat file using group header and types of metadata.
R_1008	Look Up: Update Dates	It identifies and looks for data update in a metadat file using group header and types of metadata and keywords.
R_1009	Look Up: License Information	It identifies and looks for license information in a metadat file using group header and types of metadata and keywords.
R_1010	Look Up: Publisher	It identifies and looks for publisher in a metadata file using group header and types of metadata and keywords.
R_1011	Look Up: Data Privacy	It identifies and looks for data provacy details in a metadat file using group header and types of metadata and keywords.
R_1012	Version Management	It identifies and looks for version policies or details in a metadat file using group header and types of metadata and keywords.
R_1013	Domain Identifier	This aims to derive a topic sentence or domain of data using metadata file, textual summary and file names.
R_1014	Search: Keywords	This policy particularly searches for 'keyword' word in metadata file to collect keywords and save.
R_1015	Search: Comments	The comments can be identified in two ways (i) using headers and (ii) using annotation tags.
R_1016	Search: Context CSV	It looks for csv's that might contain metadata in primary downloads.
R_1017	Search: Headers	It looks for annotated headers in a metadata file.
R_1018	Search: Empty Lines	It looks for annotated empty lines in a metadata file.
R_1019	Search: Group Headers	It looks for annotated group headers in a metadata file.
R_1020	Search: Metadata Types	It looks for annotated metadata types in a metadata file.
R_1021	Search: File Type	It looks for annotated file types in a metadata file.
R_1022	Search: Archives	It looks for archives in a metadata file using header and keyword information.
R_1023	Search: Data Summary	It looks for data summary in a metadata file using header and keyword information.
R_1024	Look Up: Hyperlinks	It looks for hyperlinks inside a metadata file and stores them.

Table 1: Sample Policy description for Rule based agents

Annotation Type	Annotation Description
Metadata File Headers	A metadata file header can include file title, the data set title, and other headings or sections in a metadata file such as Keywords etc.
Metadata Group Headers	A group header is a header that can represent one or more headers, a collection of textual description can contain tables and column information.
Metadata Content	This is actual value or content against metadata headings, the content inside metadata headers and group headers, values against metadata types such as created by ,
Metadata Types	This identifies different types of metadata such as descriptive, administrative, technical etc.
Empty Lines	An empty line, group of empty lines or empty space that contains no actual metadata.

Table 2: Sample Policy description for Rule based agents

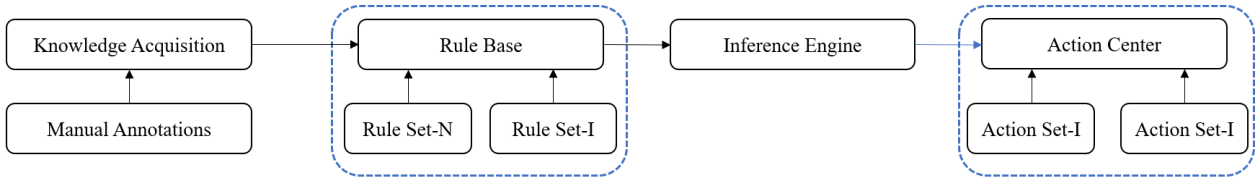

Figure 3: The overall system process from knowledge acquisition to the action center where agents perform different sets of actions.

carried on. After grouping, we add an annotation that rule-based agents can use to categorize, organize, and categorize content in metadata files. We refer to this phase as Anatomy Tagging i.e., in this step, we annotate sections that represent the anatomy of a metadata file. For example, table 2 illustrates the anatomical annotation contents. Headers and group headers are annotated to support deterministic and scalable search through a metadata file. Empty lines, empty paragraphs, or empty chunks can indicate the end of files, termination triggers, etc. To deal with this, we annotated empty lines in a metadata file and set actions for rule agents to delete and discard provided it does not change information at hand. For instance, textual metadata files contain empty spaces, headers such as a Description that includes a bunch of text about the dataset. The agents need context and understanding of what classifies as a header and what is the actual content or metadata value.

For this purpose, we pre-process the metadata files and assign meaningful annotations so the rule agents can process the files, extract, categorize, and organize information more accurately. Thus, incorporating the kind of metadata available in a file for the rule-based agents is critically important. We dealt with this challenge by annotating limited information in three broad metadata categories and their subcategories(see section 1.1). Mostly metadata files are messy and unprepared; the objective is to utilize rule agents to organize and make sense of the messy files and generate profiles. To achieve this, the agents must understand the kind of information they are dealing with. For this, we annotate examples from different types of metadata so rule agents can categorize and organize metadata files. Figure 4 indicates the annotation strategy applied for a small file.

3.2 Annotation Example

Let us look at a simple use case from collected and annotated metadata file repository, the metadata file acquired from UK GOV Road Safety Data (2005-2015) from Kaggle [2]. Figure 4 depicts a simple metadata file extracted from Kaggle web page. It does not contain a metadata file; as a response, the agent adds a title inferred from other metadata files for the UK GOV Road Safety dataset. It is important to note that there is plenty of more metadata available on the Kaggle web-link, but it is unorganized, and it is not marked or labeled as metadata. There are typically three use cases that we address namely:, (i) Case-I: when there is no metadata available, in this case, we use techniques such as exploratory data analysis (EDA) [31] and data profiling [1, 24] to derive meaning from the data itself. (ii) Case-II: when metadata is available in high quality. Most of the time, metadata is either unavailable or in poor quality (iii) Case-III: is when metadata is

[2]https://www.kaggle.com/tsiaras/uk-road-safety-accidents-and-vehicles/metadata

Annotation Type	Indicator, Tag
Metadata File Headers	orange, (MFH)
Metadata Group Headers	green, (MGH)
Metadata Content	yellow, (MC)
Metadata Types	blue, (MT)
Empty Lines	grey, (EL)
Usage Information	
	Database: Open Database, Contents: Database
License	Contents
Visibility	Public
Maintainers	
Dataset Owner	Thanasis
Updates	
Expected update frequency	Not specified
Last updated	15/01/2019
Date created	14/08/2018
Current version	Version 3

Figure 4: Annotation For Different Cases in a Metadata File: The image indicates annotation key i.e. the process used to annotate different cases such as comments, empty lines, metadata types, etc.

available but it is either misplaced or of bad quality. This case is where most of this research paper focuses. We have worked with a dataset that does provide textual metadata, descriptive metadata, metadata tags, headers, context. But, it is misplaced or, more precisely, unorganized. Our goal was to identify this metadata using rule agents. We have designed an instruction set for metadata files to facilitate our rule agents in detecting, identifying, and categorizing metadata file contents. We annotate these instruction set categories and supply them to rule-based agents (RBA). Table 2 indicates the annotations we perform the first time files are accessed. These files are annotated and are then further processed by rule agents. The table 2 includes the main headers that might appear in a metadata file, empty lines are identified to avoid repetitive cycles for AI agents. It is important to note that if there are sub-headings or, 'headers'. We annotate it as a 'Metadata group header'. It helps the agent in differentiating between file headers and group headers that can hold a chunk of metadata or information. The chunks of metadata are treated as data and are further analyzed to identify the types of metadata it contains. This is also an annotation category identified as 'metadata types' in table 2. Once the annotation is complete, now we have to design rule sets that can be utilized by policy agents to render a decision. The collection of files, annotation, metadata cleaning & organization, and preparation comprise the knowledge acquisition phase. Figure 3 illustrates four main stages for a

policy-based or rule-based agent. The second step involves Rule Base, a rule base is a collection of rule sets or policy sets. These policy sets are designed to support agents in performing different actions. The rule sets are analyzed by an inference engine that makes sense of how each rule matters and applies to the case. Finally, an action center is a set of actions an agent is permitted or allowed. Thus, a policy agent can perform a certain limited set of actions based on an inference derived by rule application on metadata files.

4 EVALUATION

The role of annotated metadata files for rule agents was observed by conducting experiments that evaluated and compared the rule agent's ability to categorize and organize metadata files with and without annotations. Another objective of the experiments was to observe how the rule agents performed if rules were altered or manipulated. The experiments were conducted on a spreadsheet dataset that was gathered from Kaggle metadata files, Kaggle resource descriptions, and Data.gov metadata files from various categories. A total of 1123 metadata files were retrieved from both repositories and added to a "mixed collection" to the working repository for rule agents. Throughout the research paper, we will refer to the mixed collection as a collection of metadata files from both Kaggle and Data.gov. As far as our research contribution is concerned, to the best of our knowledge, there are no substantial contributions that directly address and aim at the exact definition of the problem. However, there are related tools such as Metanome [24] that can be tested with the technique we have developed to assess how their performance improves or deteriorates. With regard to the scope of this research paper, we would like to emphasize that the experiments carried out are based on the usability of the technique and how meaningful observations have been made with regard to metadata management using rule agents. It would go beyond the scope of this research paper to compare and examine the improvement of existing tools by the technique we have developed. Nevertheless, our future work will be concentrated on observing performance change and the applicability of our technique to other tools.

4.1 Evaluating Rule Agent Performance

To understand the role of annotated metadata files for metadata categorization and file organization. It is crucial to observe how agents behave with and without annotations. This experiment was performed with a set of five rule agents. In the first phase, these five rule agents were provided metadata files that were not annotated, or labeled. The same set of agents were analyzed and observed in a setting with annotated metadata files. Each agent was randomly assigned metadata files from the entire mixed collection metadata files retrieved from Kaggle and Data.gov. Figure 5 illustrates the performance of each agent with and without annotations. For instance, the AR1 represents rule agent_001 with annotated metadata files and, UR1 represents the performance of the same agent with unannotated metadata files. Each agent in both categories (annotated, unannotated files) was evaluated on four factors: (1) accurate identification of metadata headers, (2) correct identification of metadata group headers, (3) MD content or values against metadata types, and (4) accurate categorization of available metadata into metadata types. File annotation request was also examined for all agents, to understand how many times an expert user was prompted for a manual annotation request. This happened when the rule agent could not decide on the categorization of metadata value at hand or the occurrence of empty blocks, empty lines, etc (see figure 6). The experiments in figure 6 highlight the frequency of manual annotation request of 5 rule agents(case-I agents-with annotated files, and case-II 5 rule agents-unannotated metadata files). Each time an agent cannot categorize content i.e. data object, header, content, text block, etc., it sends a request for manual annotation to the user. This represents indecisiveness as these agents are simple policy agents and are not reinforced, agents. From this experiment, we expected to learn about the usability of adding annotations for metadata headers, group headers, content, and types. As a result, we observed that unannotated files required significantly more annotation requests as compared to files that were pre-annotated.

4.2 Detecting Metadata Types and Analyzing Policy Frequency

The second experiment was to observe if rule agents can independently detect metadata types with and without annotations. The objective was to alter policies for rule agents and observe if it would affect the overall change in metadata type detection. Firstly, rule agents without annotations were 80% of the times requesting for a manual annotation if a metadata type had to be detected. Figure 7 indicates detectable metadata types (Descriptive (DMD), Administrative (AMD), Structural (SMD), Rights (RMD), Preservation (PMD)) in a number of files(recorded in percentages). We observed that most of the metadata files did not contain the exact information such as resource link to the original data repository, update policies, copyrights, etc. In most cases, this metadata could however be retrieved, we deal with this and regard it as "misplaced metadata". Thus, rule agents with annotated files were able to detect more metadata types and were intelligent enough to identify discrepancies or missing information from files. In regards to "misplaced metadata", we made a few observations on collected files. Some of the most prominent observations were as follows: **Lack of metadata support and updates:** most websites, resources, and portals do not maintain a complete metadata update cycle. The information on the dataset description page and the actual metadata file is observed to be incoherent. **Lack of metadata download support:** repositories and resources describe a resource on the main page or portal but disregard the facility to download this valuable metadata. For example, the repository pages at Kaggle and Data.gov both contain essential descriptive and rights metadata. This information such as resource description, keywords is not readily available for downloads. It is also not properly tagged and is not automatically embedded or added to downloadable metadata files. In most cases, it has to be manually downloaded or added to the metadata collection.

Another important experiment was to observe how many times a particular rule was accessed and applied by the rule agents. This was observed by maintaining a Rule log in each cycle for all rule agents. Figure 8 depicts the frequency (recorded in percentages) of rules fired by each rule agent. The most useful aspect of this observation experiment was to understand the type and frequency of metadata content prevalent in collected files. For example, R_1006 firing frequency is quite high indicating that the rule agents encountered null and empty values in the majority of metadata files. Similarly, R_1022 firing frequency was quite low in comparison to other rules indicating that the majority of the metadata files did not contain information relevant to data archives, metadata archives, or legacy support files and

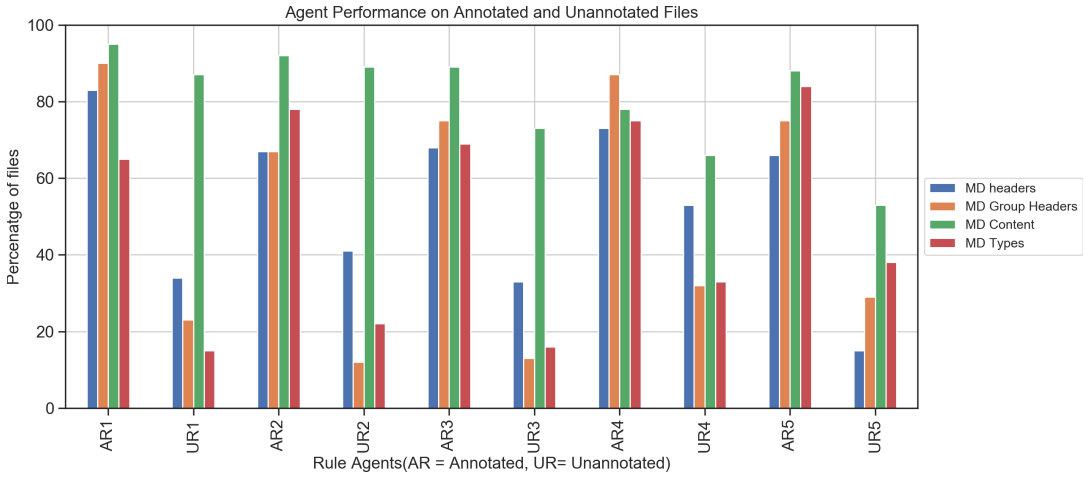

Figure 5: The overall action performance of agents on annotated and unannotated metadata files that involves use of content, annotations and set of action sequences (mixed collection).

Annotation Percentage	RA_101	RA_102	RA_103	RA_104	RA_105
25%	47%	45%	57%	59%	43%
50%	65%	78%	69%	75%	84%
75%	95%	92%	89%	78%	88%

Table 3: The table illustrates the change in behavior of rule agents based on different annotation levels of 25%, 50%, and 75%.

Dataset	Total Files	Pre-Processing	Rule Applicability Rate	Rule Failure Rate	Rule In-applicability Rate
Kaggle	563	91%	73%	11%	16%
Data GOV	560	45%	87%	8%	5%

Table 4: Dataset Collection Comparison and Analysis.

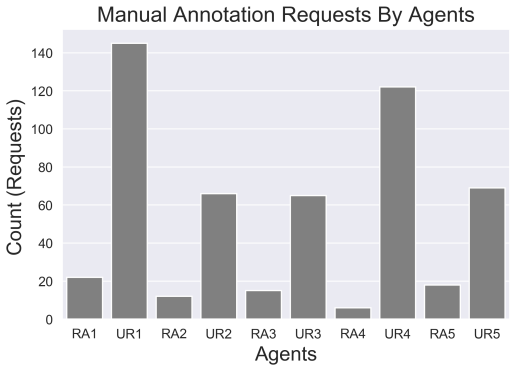

Figure 6: The number of manual annotations requested by agents with annotated and unannotated files mixed collection. The performance of agents depicted in the above figure is based on 75 percent annotation level of metadata files from the mixed collection.

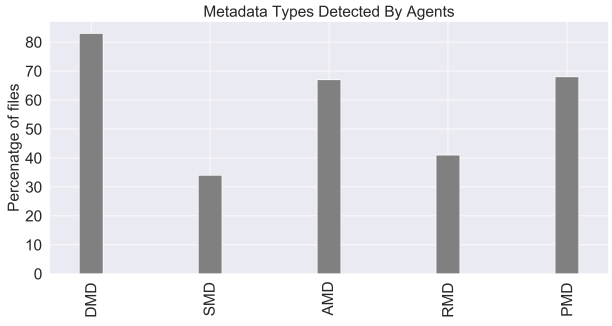

Figure 7: The figure represents the percentage of files that rule agents were able to detect different metadata types (mixed collection).

hyperlinks. Another aspect that can be observed from figure 8 is that not all metadata files contain relevant hyperlinks. As indicated R_1024 was accessed by agents a couple of times.

The table 1 includes a sample of 24 rule sets that are measured in figure 8 that demonstrates the rule firing frequency by rule agents on the mixed collection. It visualized how each rule is fired and how many times it is utilized on average by each agent. The table 1 provides insight into policies available to all rule agents for both annotated resources and unannotated resources. Each policy defined in table 1 is made available to rule agents based on their tuning. We refer to tuning, i.e."a set of policies" as profile setting available to a particular agent. In our research, we experiment with agents in different settings by increasing or decreasing the number of policies accessible by a particular agent. For example, R_1010 searches the file for keywords as "publisher", synonyms, and approximate matches for this keyword group. The

functionality allows a rule agent to scan headers, group headers, and text blocks to locate a valid publisher and its details in a metadata file.

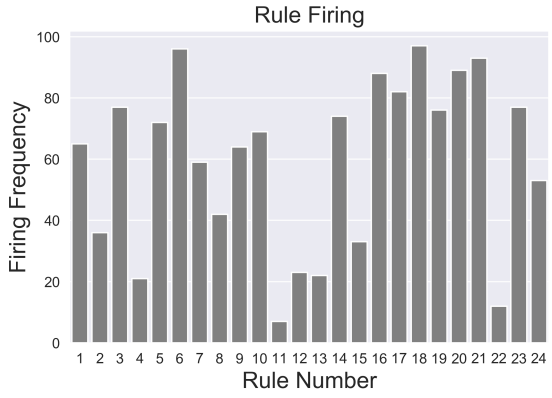

Figure 8: An observation of average rule firing frequency by rule agents on mixed collection

4.3 Agent Actions

The policy agents are assigned a set of actions that can be chosen to attain the next state and complete the assigned task (based on available policies provided to each agent). The allowed set of actions for any rule agent in this experimental setup include the following: **Read:**when the agent scans and reads the file ingested into the system. **Search:**a search action that allows the agent to scan the file, gather insight and look for common words, indicators and headers. **Write:** allows the agent to write into a new metadata file. **Match:** is an action where the agent matches its knowledge of annotation with available tags in the metadata file and cases where there is a need to search for synonyms, etc. **Drop:** is the case where a content is dropped or considered invaluable by the agent and added in the metadata file as 'supplementary information'. **Delete:** is when an agent decides to delete a piece of content such as consecutive delimiters, misspelled words. **Compare:** when two or more headers, sub-headers, or content pieces are compared for duplicate data identification. **Request Annotation:** when an agent is indecisive on a set of sentences such as comments (if not annotated), the agent requests for a manual annotation from the user(this action is typically requested after the agent completes the read cycle and before the write operation). In table 5, five rule agents are observed for action behaviors on the same metadata file. The experiment observes the role of policies and how it changes the action initiation sequence for each agent. For example, RA_101 does not exclude the dropped and deleted items. Thus, the read items are the same as written items in new metadata files but with a different context. On the other hand, agents like RA_102, RA_103, etc., do not add these items into the total write count for newly generated metadata files. The difference in operations is based on the accessibility of policies and action preferences assigned to each agent. All five agents were observed for different settings to understand how actions and operations differ based on agent action allowances. Thus, by changing policies we were able to observe significant changes in the performance of agents for the same metadata files.

Actions	RA_101	RA_102	RA_103	RA_104	RA_105
Read	1189	1189	1189	1189	1189
Search	1458	1650	1687	1465	1093
Write	1189	1192	1125	1143	1061
Match	4567	4987	4976	3456	1148
Drop	10	120	56	34	26
Delete	0	3	8	12	8
Compare	201	189	300	450	120
Annotation Req	109	80	344	450	122

Table 5: An overview of average agent actions on a metadata file. The table illustrates the importance of different profile settings for rule agents and their impact on agent actions.

4.4 Agent Performance and Analysis

To obtain a better picture and understanding of agent performance we designed three major categories of experimental observation. Firstly, we tested our technique and rule agents on mixed collection i.e., a collection of metadata files obtained from Kaggle and Data.gov. Figure 9 illustrates the performance of each rule agent on the mixed collection. The second experimental setting was to observe how agents performed with different levels of annotated datasets. Table 3 provides an analysis of each agent's performance in accordance with the annotation percentage. We defined three annotation levels starting from 25%, 50%, and concluding with 75% annotated metadata files. By observing how agents performed we were able to understand that annotated metadata files do significantly improve the agent performance even in the most difficult cases such as metadata type identification. Most importantly, from this experiment, we observed how each agent was affected based on its primary policy selection (i.e., setting for each agent in terms of allowed or accessible policies). For example, in table 3.

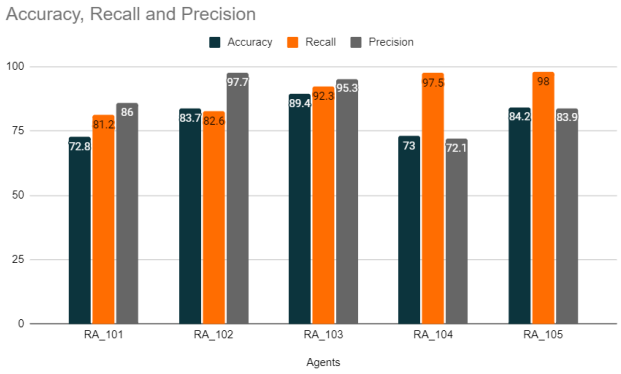

Figure 9: An observation of overall agent performance in terms of accuracy, precision, and recall.

4.5 Metadata Collection Analysis

It is very critical to understand how the technique responds to different types of datasets. In our case, the aim was to first collect metadata files that can be analyzed by rule agents. In the context of our research problem, we did not limit the collection of metadata files to a certain domain. Thus, the collection comprised metadata files from various domains such as accidents, Geospatial metadata files, pharmacy, music, movies to name a few. The

challenge was, however, to understand and find relevance (if it persists) in these metadata files and secondly to identify different constituents of metadata. Based on our collection, we worked with a total of 1123 metadata files. A total of 563 files were obtained from the Kaggle repository (on numerous domains and topics), and a total of 560 metadata files were retrieved from the Data.gov repository (on numerous domains and topics). Table 4 depicts our overall observations and understanding of metadata collection. We have separately analyzed both collections on the following criteria:

4.5.1 Pre-Processing. : this metric identifies the amount of pre-processing required on files before it can be fed to the rule agents for further categorization and analysis. Pre-processing is an important aspect as most metadata files are raw and do not come processed with preambles, labels, and annotations. Moreover, the file structures and standards vary for the majority of the cases. We observed that metadata files from Kaggle required more pre-processing in terms of missing values, incorrect values, and file standardization. Also, metadata from Kaggle had many cases of "misplaced metadata" making pre-processing a necessary step in the process. On the other hand, metadata retrieved from Data.gov required basic cleaning and file preparation and was already in structured formats.

4.5.2 Rule Applicability Rate. : we also observed how many rules were applicable on a pre-processed metadata file from both collections. Table 4 identifies a relative difference between rule applicability on both data collections. However, due to more header and tag oriented file structure, the metadata files from Data.gov were perceived to be more rule applicable in comparison to Kaggle collection.

4.5.3 Rule Failure Rate. : the rule failure rate is described as a failed attempt of rule application by a designated rule agent. If an agent executes a rule and there is no profitable outcome such as no conclusive agent action is performed and no task is completed it is recorded as a failure rate. The files from Kaggle had an 11%failure rate and metadata collection from Data.gov had an 8% failure rate as depicted in table 4.

4.5.4 Rule In-applicability Rate. : this is defined in terms of lack of metadata elements or lack of metadata content inside metadata files. The unavailability of information in metadata files causes a rule in-applicability. It means that certain rules were never applied by agents as those elements, keywords, headers, or information content was not available. From a collection of agent polices, a total of 16% rules were inapplicable in different phases on files retrieved from Kaggle, and a total of 5% rules were inapplicable on metadata gathered from Data.gov (see table 4).

5 CHALLENGES

Some of the most eminent challenges we faced were: Metadata collection: Metadata files were available for some datasets. However, most of the metadata was misplaced or can be gathered from other resources. Thus, one file with all metadata was a rarity. In this section, we discuss some of the most difficult challenges we encountered. for some of these challenges, we aim to provide partial solutions and extend them to our future work.

Metadata Pre-processing In almost all of the files collected from the Kaggle repository and Data.gov, the metadata had to be cleaned and prepared before it can be utilized by our system. The prepossessing is a particular challenge as all files are in different formats, the information does not strictly adhere to standards, and the files are populated differently for each dataset. Thus, metadata file pre-processing is one of the challenges that we currently deal with and address. Most of the metadata files were raw, and metadata was dispersed and misplaced a pre-processing step had to be established.

Conflicting Labels and Label Cleaning We address this by designing labels to avoid conflicts in the ground truth. The strategy we defined includes a growth pattern that allows for labels to evolve and change over time if needed. We set this up by providing functions that allow the user to manipulate non-critical labels (the implementation details of this feature are beyond the scope of this research paper). For the context of the current research paper, the labels added are marked for each file, and a non-conflicting hierarchy of labels is executed to avoid confusion for the rule agents. For example, if a file contains more than one main header, it should explicitly contain separate information and should be a different word. One file cannot have two main headers title says "Descriptive Metadata" or "Publisher". Such hierarchical control allows for information categorization and avoids conflicting labels. We have extended our work on dealing with conflicting labels and intend to improve as we move along.

Nested Files and Data In the context of our research we only deal with non-nested semi-structured files.

Failure Despite Annotations we prepared ground truth to deal with the most frequent and available information in metadata files by providing annotations. However, even with annotation practices, the rule agents were unable to provide a decisive result on cases such as identification of the most recent metadata file version. This was a failed use case for rule agents. The annotation and content information was insufficient to make a necessary conclusion.

Unreadable Text The collection contained metadata files that had discrepancies in terms of incorrect (text in English with incorrect spellings etc.)and indiscernible text (text in another language). In both cases, the rule agents were unable to classify the text in any of the categories. Since we do not provide multilingual support and many metadata files contain information from different languages such as French, Chinese, Italian, etc. This remains a challenge for our current research.

Incomprehensible Metadata Files The collection comprised of files that contained metadata information such as schema and legacy links contained inside an image. This was a challenge and a limitation of our system since we do not process image files. Thus, we had to exclude such files from our collection.

Inconclusive Column Names This is a semantic challenge that is not a part of our problem description. As we deal with metadata files, and categorize the information inside them, clean it for better reusability. Nonetheless, we identified in about 25% of metadata files in the extended collection (before scrutiny, the final collection contained 1123 files only) contained column names that did not represent the information accurately. Also, most of the time column names and column values were conflicting. If the column name was, for example, "Publisher" it would be expected to contain the dataset publisher name, date, or resource link. But it contained a phone number or city name.

Empty Files We address this issue partially in our current work by identifying blocks of empty data and empty files. This is another challenge that is more oriented towards understanding file structures and identifying different tables or contents inside a file. It is beyond the scope of our research, we only identify

empty metadata files from our extended collection and disregard them from the main collection i.e., 1123 files.

Inconclusive Headers This is a challenge that we address in our current research and we are extending on this challenge in our future work as well. The problem inhibits naming conventions and identifying whether a title or text is to be considered a header or sub-header. Primarily we deal with this by intuitive annotation techniques in our ground truth. We intend to extend this approach by developing a tree-based structure for the title, header, and sub-header. The tree would be designed to identify different levels of headers and text to remove the ambiguity of inconclusive headers and their hierarchy in metadata files.

Other Challenges Another case was misplaced commas and delimiters, rule agents failed because preparation was manually done and no rules directly addressed punctuation, delimiters, misplaced commas, etc. Another example of rule agent failure includes the case of broken text paragraphs and empty columns. Most of the time the rule agents were able to identify an empty row or column if it was labeled, on the other hand, we did observe rule agents in a cycle of interpretation of empty rows in sheets. Finally, missing values were also an issue even with annotations, as rule agents work by example meaning, the rule agents failed in identifying if the content or "metadata value" was placed against the correct metadata category.

6 CONCLUSION

In this paper, we have discussed the prospect of utilizing rule agents that could facilitate the metadata categorization based on two types of metadata files i.e., annotated MD files and unannotated metadata files. To support and enhance the use of metadata in data management and business information systems it is crucial to minimize the amount of time spent on pre-processing and categorization. We provide a policy-based system attributed by policy agents to identify, observe, and categorize available metadata files for reusability and easy understanding. To obtain this, we explored the possibilities of changing policies and observing the behavior of agents. By our observation, change in available and accessible policies affected the agent's ability to correctly categorize a piece of metadata in a file. We also observed that different files contain information that might be completely irrelevant based on the policies defined in our research. This method of experimentation helped us in understanding the core components of metadata files and how policy agents can best fit a metadata file categorization use-case. The second experiment we conducted was to utilize manually annotated files for rule agents to observe improvement and ease in metadata file categorization and processing. As a result, we observed that annotated files are helpful to policy agents as the search space is more reasonable, and the policies can easily identify the type of item in the file. We intend to take a few steps to improve the readability and categorization of metadata files using rule agents. In our upcoming experiments, we intend to develop self-learning and service-based AI agents that can learn from their success and failure rate as well as user feedback. We will be focusing on allowing agents to learn from a feedback network and how it affects the user decision process. A feedback system that can identify how each policy contributed to metadata categorization.

ACKNOWLEDGMENT

The work of Hiba Khalid is supported by the European Commission through the Erasmus Mundus Joint Doctorate project *Information Technologies for Business Intelligence-Doctoral College* (IT4BI-DC).

REFERENCES

[1] Ziawasch Abedjan, Lukasz Golab, and Felix Naumann. 2017. Data profiling: A tutorial. In *Proceedings of the 2017 ACM International Conference on Management of Data*. 1747–1751.
[2] Carlo Batini, Maurizio Lenzerini, and Shamkant B. Navathe. 1986. A comparative analysis of methodologies for database schema integration. *ACM computing surveys (CSUR)* 18, 4 (1986), 323–364.
[3] Rafael Berlanga, Oscar Romero, Alkis Simitsis, Victoria Nebot, Torben Bach Pedersen, Alberto Abelló, and María José Aramburu. 2012. Semantic web technologies for business intelligence. In *Business Intelligence Applications and the Web: Models, Systems and Technologies*. IGI global, 310–339.
[4] Sanjay P Bhat and Dennis S Bernstein. 2000. Finite-time stability of continuous autonomous systems. *SIAM Journal on Control and Optimization* 38, 3 (2000), 751–766.
[5] Costin Bădică, Lars Braubach, and Adrian Paschke. 2011. Rule-based distributed and agent systems. In *International Workshop on Rules and Rule Markup Languages for the Semantic Web*. Springer, 3–28.
[6] Serena H Chen, Anthony J Jakeman, and John P Norton. 2008. Artificial intelligence techniques: an introduction to their use for modelling environmental systems. *Mathematics and computers in simulation* 78, 2-3 (2008), 379–400.
[7] Jens Dietrich, Alexander Kozlenkov, Michael Schroeder, and Gerd Wagner. 2003. Rule-based agents for the semantic web. *Electronic Commerce Research and Applications* 2, 4 (2003), 323–338.
[8] Erik Duval. 2001. Metadata standards: What, who & why. *Journal of universal computer science* 7, 7 (2001), 591–601.
[9] Neil Foshay, Avinandan Mukherjee, and Andrew Taylor. 2007. Does data warehouse end-user metadata add value? *Commun. ACM* 50, 11 (2007), 70–77.
[10] Dov M Gabbay. 1985. Theoretical foundations for non-monotonic reasoning in expert systems. In *Logics and models of concurrent systems*. Springer, 439–457.
[11] S Christopher Gladwin, Matthew M England, Dustin M Hendrickson, Zachary J Mark, Vance T Thornton, Jason K Resch, and Dhanvi Gopala Krishna Kapila Lakshmana Harsha. 2009. Metadata management system for an information dispersed storage system. US Patent 7,574,579.
[12] Jane Greenberg. 2009. Metadata and digital information. In *Encyclopedia of library and information sciences*. CRC Press, 3610–3623.
[13] IEEE. [n.d.]. IEEE LOM: IEEE Standard for Learning Object Metadata.
[14] William H Inmon, Bonnie O'Neil, and Lowell Fryman. 2010. *Business metadata: Capturing enterprise knowledge*. Morgan Kaufmann.
[15] S Jha and Mahbub Hassan. 2002. Building agents for rule-based intrusion detection system. *Computer Communications* 25, 15 (2002), 1366–1373.
[16] Phokion G Kolaitis. 2005. Schema mappings, data exchange, and metadata management. In *Proceedings of the twenty-fourth ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems*. 61–75.
[17] Brian F Lavoie and Richard Gartner. 2005. *Preservation metadata*. OCLC.
[18] Zhen Li and Manish Parashar. 2004. Rudder: A rule-based multi-agent infrastructure for supporting autonomic grid applications. In *International Conference on Autonomic Computing, 2004. Proceedings.* IEEE, 278–279.
[19] Marilyn McClelland. 2003. Metadata standards for educational resources. *Computer* 36, 11 (2003), 107–109.
[20] Renée J Miller. 2018. Open data integration. *Proceedings of the VLDB Endowment* 11, 12 (2018), 2130–2139.
[21] Felix Naumann. 2014. Data profiling revisited. *ACM SIGMOD Record* 42, 4 (2014), 40–49.
[22] Nils J Nilsson and Nils Johan Nilsson. 1998. *Artificial intelligence: a new synthesis*. Morgan Kaufmann.
[23] NISO. [n.d.]. Understanding metadata. National Information Standards Organization. NISO Press, 2004.
[24] Thorsten Papenbrock, Tanja Bergmann, Moritz Finke, Jakob Zwiener, and Felix Naumann. 2015. Data profiling with metanome. *Proceedings of the VLDB Endowment* 8, 12, 1860–1863.
[25] David Peto and Stef Lewandowski. 2015. Metadata tagging of moving and still image content. US Patent 8,935,204.
[26] IBM Redbooks. [n.d.]. http://www.redbooks.ibm.com/.
[27] Arun Sen. 2004. Metadata management: past, present and future. *Decision Support Systems* 37, 1 (2004), 151–173.
[28] John R Smith and Peter Schirling. 2006. Metadata standards roundup. *IEEE MultiMedia* 13, 2 (2006), 84–88.
[29] Alessandra Toninelli, Jeffrey Bradshaw, Lalana Kagal, Rebecca Montanari, et al. 2005. Rule-based and ontology-based policies: Toward a hybrid approach to control agents in pervasive environments. In *proceedings of the Semantic Web and Policy Workshop*.
[30] Jason Tsay, Alan Braz, Martin Hirzel, Avraham Shinnar, and Todd Mummert. 2020. AIMMX: Artificial Intelligence Model Metadata Extractor. In *Proceedings of the 17th International Conference on Mining Software Repositories*. 81–92.
[31] John W Tukey et al. 1977. *Exploratory data analysis*. Vol. 2. Reading, Mass.
[32] Jovan Varga, Oscar Romero, Torben Bach Pedersen, and Christian Thomsen. 2014. Towards next generation BI systems: the analytical metadata challenge. In *International conference on data warehousing and knowledge discovery*. Springer, 89–101.
[33] Marcia Lei Zeng. 2008. *Metadata*. Neal-Schuman Publishers, Inc.