2023 IEEE Sixth International Conference on Artificial Intelligence and Knowledge Engineering (AIKE) | 979-8-3503-3128-8/23/$31.00 ©2023 IEEE | DOI: 10.1109/AIKE59827.2023.00015

Anomaly Detection for Numerical Literals in Knowledge Graphs: A Short Review of Approaches

Farshad Bakhshandegan Moghaddam
University of Bonn
Bonn, Germany
farshad.moghaddam@uni-bonn.de

Jens Lehmann
TU Dresden, Amazon
(work done outside of Amazon)
Dresden, Germany
jens.lehmann@tu-dresden.de

Hajira Jabeen
GESIS-Leibniz Institute for the Social Sciences
Cologne, Germany
hajira.jabeen@gesis.org

***Abstract*—Anomaly Detection is an important problem that has been well-studied within diverse research areas and application domains. However, within the field of Semantic Web and Knowledge Graphs, anomaly detection has been relatively overlooked. Additionally, the existing literature on anomaly detection over Knowledge Graphs lacks proper organization and poses challenges for new researchers seeking a comprehensive understanding. In light of these gaps, this paper aims to offer a well-structured and comprehensive overview of the existing research conducted on anomaly detection over Knowledge Graphs. In this overview, we review the quality metrics of KGs and discuss the possible errors which may occur in different parts of the RDF data. Additionally, we outline a generic conceptual framework for the execution pipeline of Anomaly Detection over KGs. Moreover, we study the anomaly detection techniques, along with their variants, and present key assumptions, to differentiate between normal and anomalous behavior. Finally, we outline open issues in research and challenges encountered while adopting anomaly detection techniques for KGs.**

***Index Terms*—Anomaly Detection, Knowledge Graphs, Outlier Detection, RDF Data, Semantic Web, Linked Open Data**

I. Introduction

With the ever-increasing amount of data available on the Internet, it is becoming vitally important to have a set of tools to extract meaningful and hidden information from the data. The Semantic Web can create a structural view of existing web data and provides machine-readable formats [1]. To facilitate this, the World Wide Web Consortium[1] introduced the Resource Description Framework (RDF)[2] as a standard to model the real world in the form of entities and their relationships. RDF data are a collection of triples `<subject,predicate,object>` with rich relationships that can form a potentially huge and complex RDF graph.

Nowadays, many companies in science, engineering, and business, including bio-informatics, life sciences, business intelligence, and social networks publish their data in the RDF format. Furthermore, the Linked Open Data Project initiative [2] has aided the Semantic Web in gaining traction over the last decade. The Linked Open Data (LOD) cloud currently comprises more than 10,000 datasets available online[3] using the RDF standard.

KGs are being exploited in different real-life use cases such as search engines, industry, medical science, and many more. However, to gain the maximum benefit, the KGs should assure a certain level of quality. This is not a problem per se, because quality typically denotes suitability for a certain use case [3]. KGs are being produced in a variety of ways. Crowdsourcing was used to create some KGs, such as Wikidata [4] and Freebase [5]. Natural language processing techniques were used to create NELL [6], and DBpedia [7] and YAGO [8] were automatically constructed by knowledge extracting tools. Typically, when the entered data lacks restrictions and cross-validation, KGs become vulnerable to different types of errors due to the diverse approaches and freedom in input data insertion. These errors can happen at logical or semantic levels and can occur at `subject`, `predicate` or `object` part of the RDF (check Section II for more information).

One means of finding these errors in KGs is Anomaly Detection (AD). AD is a sub-field of data mining dedicated to the discovery of uncommon events in datasets and has several high-impact applications in sectors such as security, finance, health care, law enforcement, and much more [9]. It is the task of identifying data points and patterns that do not conform to the previously specified behavior of the data. AD is already a well-studied field with the focus specifically on the task of anomaly detection in non-relational datasets [10]. Over the years, numerous techniques have been developed for detecting outliers and anomalies (anomaly and outlier will be used interchangeably in this paper) in unstructured collections of multidimensional points. However, with the current interest in large-scale heterogeneous data in Knowledge Graphs (KGs), most of the traditional algorithms are no longer directly applicable to KGs due to scalability and RDF complex data structure. Furthermore, to the best of our knowledge, there has not been a lot of dedicated research work on anomaly

[1]https://www.w3.org
[2]https://www.w3.org/RDF/
[3]http://lodstats.aksw.org/

979-8-3503-3128-8/23/$31.00 ©2023 IEEE
DOI 10.1109/AIKE59827.2023.00015

detection on KGs. Therefore the aim of this paper is to provide a general, short but comprehensive, and structured overview of the already existing approaches for anomaly detection in data represented as RDF. As a key contribution, our investigation delves into the anomaly detection problem on KGs from various perspectives. We thoroughly examine the different techniques and methods already employed for this purpose.

A. Goal and scope of this survey

As KGs are finding their way in day-to-day usage, producing high-quality KGs and controlling their quality is playing an important role. KGs may contain multiple and various types of errors (Section II), however, the scope of our survey is to provide an overview of AD approaches for finding anomalies in literals. This decision was influenced by the fact that outlier detection predominantly revolves around numeric data, making numeric literals a logical focal point. Moreover, most of the already existing research in this area focuses on outliers in numerical literals.

B. Article Collection Methodology

The goal of this review is to provide a theoretical framework that can be applied to a wide range of approaches for AD in KGs. In conducting this survey, we used the Keyword Search protocol and considered Google Scholar as an academic database. Keywords such as *outlier detection*, *anomaly detection*, *knowledge graphs*, *Numeric Literals*, etc are considered to fetch the result. Upon careful examination of the findings, we retained the papers that specifically addressed the utilization of AD techniques on numeric literals in KGs.

C. Contributions

More specifically, this article makes the following contributions:

- a short review of existing AD approaches on KGs for numeric literals
- introducing the frequently used AD techniques over KGs
- presenting open issues in research and challenges faced while adopting anomaly detection techniques for KGs

The remainder of the paper is organized as follows: We start with introducing KG quality metrics and possible errors which may occur in KGs in Section II. In Section III we define anomaly and anomaly detection types. In Section IV AD methods over KGs and underlying techniques are introduced. Section V covers existing works in the area of anomaly detection over numeric literals. Finally, Section VI concludes the paper by introducing open challenges and possible future directions.

II. Errors in KGs

KGs are used in a range of applications such as semantic search, question-answering systems, and recommendation systems [11]–[13]. The quality of a KG is essential for its effectiveness in a particular application, so it is important to carefully control the quality of KGs during their construction and maintenance. There are various methods for building KGs, but they can also potentially compromise the quality of the final result. Therefore, it is important to carefully monitor the quality of each step in the construction process and identify the specific quality dimensions that may be impacted. Additionally, KGs must be regularly maintained and updated to remain current and meet changing requirements. Moreover, to uphold the overall quality, it is imperative to rectify any errors that might have been introduced during the construction process of the KG. The quality of the KG can be viewed as a multi-dimensional topic. There is a significant amount of work has been conducted to evaluate the dimensions of KG quality. The most high-level dimensions are: a) Accuracy [14] b) Completeness [15] c) Consistency [16] d) Timeliness [17] e) Trustworthiness [18]

Table I lists the dimensions, definitions, and examples. Although there exists a correlation among these dimensions, however, it is beyond the scope of this paper to analyze the dependency and correlation between each dimension; we refer the reader to previous survey publications for a thorough discussion [19].

Each of the main dimensions can be divided into sub-dimensions and there are many works conducted to mathematically define and evaluate the metrics [14], [20]–[22]. However, this paper solely concentrates on the metric of *Accuracy (Precision)*, as outliers play a crucial role in determining the accuracy of a KG. The accuracy of a KG reflects the extent to which the knowledge it contains aligns with established facts. This is considered the most crucial aspect of KG quality. Factors that can negatively impact the accuracy of a KG include incorrect relations, entities, and attributes within the graph.

RDF data are a collection of triples `<subject,predicate,object>` so the error may happen in the `subject`, `predicate`, or `object` position. `subject` can be an entity or a blank node, `object` can also take an entity, a blank node, or a literal value with different primitive types (String, Integer, Date, ...). In the following, we explain possibilities based on the anomaly position.

A. `subject`

Anomalies can happen in the `subject` position. `subject` can be *blank nodes* or *URIs*. If the anomaly occurs when the subject is *blank node*, detecting is not straightforward because other triples also should be considered. However, if the subject is a *URI*, the correctness of the triple based on the subject can be checked. For example, in `<dbr:Film,dbp:leaderName,dbr:Joe_Biden>`, `dbr:Film` can be considered as an anomaly because it does not conform to the pattern of `<Country,dbp:leaderName,Person>` as `dbr:Film` is not of type `Country`. Ontological rules can check this type of anomaly. Worth to note that, we distinguish here the wrong values and anomalies. For example `dbr:Germany` in `<dbr:Germany,dbp:leaderName,dbr:Joe_Biden>` is a wrong value however, it does not conflict with any obvious pattern of data. These types of errors can be detected and fixed by fact-checking methods [23].

B. `predicate`

In the `predicate` position two types of anomaly may occur. Either the predicate itself can be an anomaly, for example, <dbr:Barack_Obama,dbo:elevation,61> is an anomalous triple, as `dbo:elevation` can not be used for the type `Person` but `City`. Another error happens when an entity has more/less than the usual number of the same predicate. For example, a person should have only one birthplace. However, if he/she has, for example, 5 birthplaces, then this type of (potential) error can be detected.

C. `object`

As `objects` mostly contain literals, they can be a potentially good source for anomalies. In case the `object` is a *URI*, the same approach can be applied as `subject`. However, if the `object` is literal (especially numeric literals), multiple types of anomalies can happen. For example, `dbp:year` can not take negative numbers. Or `dbo:postalCode` can not take big integers. For more examples in this category, we refer the reader to [21], [24], [25].

In this review, we introduce the existing works in the area of anomalous numeric literals. So almost all of the existing works which are mentioned in Section V focus on anomaly detection on `objects` especially when they have numeric literals [24]–[28]. Although some works try to address anomalies in `predicates` as well [24], [29].

III. Definition of Anomaly Detection

The first and most widely used definition of an outlier dates from 1980 and is given in [30]:

"An outlier is an observation that differs so much from other observations as to arouse suspicion that it was generated by a different mechanism."

As the definition indicates, anomalies are not necessarily wrong values but values that do not conform with normal data behavior. Outliers are also referred to as abnormalities, discordants, deviants, or anomalies in the data mining and statistics literature [31]. Anomaly Detection is a well-studied area and found its way to many real-world scenarios such as Intrusion Detection Systems, Financial Fraud Detection, IoT and sensor data, Medical Diagnosis, Law Enforcement, Earth Science, and many more [31].

We need to distinguish the difference between *outlier* and *natural outliers*. *Natural outliers* are the values that are not wrong. For instance, if we compare the height of a building (α) with the height of animals (β) which $\alpha >> \beta$ then the height of the building will be considered as an anomaly however, if we only compare the height of different building together, most probably the α value will not be detected as an anomaly. These types of anomalies are called *natural outliers*. Thus, when using outlier detection to find errors in data, special care must be taken to distinguish natural outliers from outliers caused by actual data errors.

Anomaly Detection techniques can be categorized from different aspects. Generally, they can be grouped to *supervised*, *semi-supervised*, and *unsupervised*. Moreover, the methods can be categorized as *neighbor-based*, *subspace-based*, and *ensemble-based* detection methods. Based on the number of features they are being applied, the methods can be classified as *univariate* or *multi-variate*. Although a comprehensive review of anomaly detection techniques is beyond the scope of this paper; we refer the reader to previous survey publications for a thorough discussion of such approaches [9], [10], [32].

Supervised and *semi-supervised* approaches require training data in which outlier/normal values are labeled. In contrast, *unsupervised* approaches do not rely on any labeled training data. As the creation of labeled training data would be rather expensive and labor extensive, the mostly used outlier detection methods are unsupervised.

The primary objective of *Neighbor-Based Detection* methods is to identify outliers using neighborhood data. For example, the anomaly score of a data point can be defined as the average distance or weighted distance to its k nearest neighbors [33], [34]. Another approach is to consider the Local Outlier Factor (LOF) [35] as the measurement of anomaly degree, in which the anomaly score was measured relative to its neighborhood. In contrast, methods in *Subspace-Based Detection* attempt to project high-dimension data to lower dimensions and then search for anomalies. The reason for this is that anomalies frequently exhibit abnormal behavior in one or more low-dimensional sub-spaces. The full-dimensional analysis would obscure low-dimensional abnormal behaviors [36]. For example [37] demonstrated that for an object in a high-dimensional space, only a subset of relevant features provides useful information, while the rest is irrelevant to the task. In the literature, subspace learning is a popular technique for dealing with high-dimensional problems [38]–[41]. Aside from that, the *Ensemble-Based Detection* method detects anomalies by utilizing various learning techniques or even multiple subspaces at the same time. Because of the complexity of the data, none of the outlier detection methods can detect all anomalies in a low-dimensional subspace. One ensemble strategy is, for example, summarizing the anomaly scores and selecting the best one after ranking [42].

If the AD approach considers multiple dimensions of data at once (for example considering longitude and latitude together to detect a geo-coordinate as an anomaly) it is called *multi-variate* and if it just utilizes a single dimension (for example only checking the age of people to detect anomalies) it is called *univariate*.

IV. Anomaly Detection over KGs

As explained in Section II, different types of errors can be hidden in the different dimensions of Knowledge Graphs. Most of the already existing research in this area, focuses on outliers in numerical literals [24]–[28]. These methods try to find anomalies on single literal value. That is, given one property, such as `dbo:elevation`, representing the elevation of a place, these methods want to detect anomalous values that are

TABLE I: Definitions of evaluation dimensions

Dimension	Definition	Example (anomaly)
Accuracy	Correctness of facts	`(Barack_Obama, birthPlace, Germany)`
Completeness	Coverage of Knowledge by the KG	-
Consistency	Degree of self-contradiction in the KG	`(John, spouseOf, Mary)` `(Mary, sisterOf, John)`
Timeliness	Degree to which knowledge is up-to-date	`(Barack_Obama, presidentOf, USA)`
Trustworthiness	Degree of objectivity, authority, and verifiability of a KG	-

used as literal objects of that property. Furthermore, collecting all values of a specific attribute, such as weight, and attempting to perform anomaly detection for this attribute is conceptually incorrect in KGs. The reason is that the same predicate can be used for different types of entities. For example, the weight of vehicles can not be compared to the weight of animals. Therefore, to overcome this problem, the existing works use a mechanism to cluster entities before applying anomaly detection techniques. Moreover, there should be a mechanism to first extract features from KG before applying any anomaly detection techniques.

Figure 1 depicts the standard pipeline of anomaly detection over KGs and Table II summarizes the existing approaches and their main characteristics which are explained in Section V. Below, we provide a brief introduction to the commonly used feature extractors, clustering methods, and anomaly detection algorithms.

A. Feature Extractor

Before being able to run anomaly detection on KGs, the KG should be featurized (also referred to as vectorization or prepositionalization). This step generates features from KGs and prepares machine learning-friendly features for subsequent processes. There are plenty of works in the area of prepositionalization [43]–[46], however, in this paper we only focus on those which have been used for the anomaly detection purposes.

1) FeGeLOD: FeGeLOD [27] is an open-source and unsupervised approach for enriching data with features derived from LOD. This approach uses six unsupervised feature generation techniques to explore the data and fetches the features. It uses 6 predefined SPARQL queries to extract information from KGs and comprises three subsequent steps: entity recognition, the actual feature generation, and the optional selection of a subset of the generated features.

2) Literal2Feature: Literal2Feature [47] is a generic, distributed, and a scalable software framework implemented over Apache Spark[4] (which is a scalable, in-memory, general-purpose cluster computing framework) that can automatically transform a given RDF dataset to a standard feature matrix by deep traversing the RDF graph and extracting literals to a given depth. It uses a scalable Breadth-First Search (BFS [48]) to traverse the KG and returns a SPARQL query that extracts the features. This option allows the user to extract features that are not in the immediate vicinity of an entity for the purpose of outlier detection.

3) Pivoting/Grouping: Pivoting is a data reshaping mechanism that produces a “pivot” table based on predicate values. The example below shows how pivoting works on a sample RDF dataset if one wants to pivot the Listing 1 based on “Predicate” and aggregate over “Object”. This approach only generates features that are in the direct vicinity of an entity.

B. Clustering

To be able to achieve precise anomaly detection results and avoid natural outliers, one needs to perform clustering over entities to avoid comparing, for example, the height of animals with the height of buildings. In this section, we briefly explain the clustering approaches used in anomaly detection over KGs.

1) Clustering by `rdf:type`: For this approach, all types of an entity one-hot encoded to a vector of boolean values, representing whether or not the entity is of a certain type. Afterward, any traditional clustering techniques such as Estimation Maximization (EM) [49] or K-Means [50] can be applied to the vectors to cluster the entities. This approach has been used in [26] however, it is only applicable to the RDF dataset which contains `rdf:type` information.

2) Clustering with Constraints: This approach is introduced in [28] that generates sub-populations of data based on classes and properties, and subsequently applies outlier detection to these sub-populations. For example, when considering a complete dataset, the populations of continents would be regarded as outliers due to their significantly higher magnitude compared to the predominant population values observed in cities or countries. A lattice is used to overcome this issue. Each node of the lattice is given a set of constraints that determine which instances are considered at that node. Because the root node has an empty constraint set, it represents all instances and corresponding values of the currently considered property. Each child node has one more constraint than its parent (a constraint can be an extra property, an extra class type, etc.). This allows the data to be divided into subpopulations. In this lattice, a leaf node refers to a node that adding additional constraints does not alter the number of instances contained within it.

3) Clustering by LHD: As for some entities, the `rdf:type` information could be missing, [25] introduced a new way of clustering based on Linked Hypernyms Dataset

[4]https://spark.apache.org/

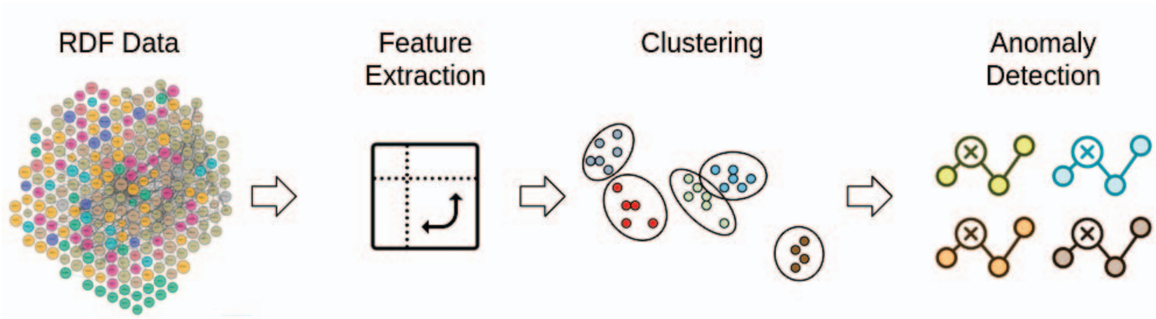

Fig 1: Standard Anomaly Detection pipeline over KGs

```
+-----------------+-------------+-----------------+
|Subject          |Predicate    |Object           |
+-----------------+-------------+-----------------+
|dbr:Donald_Trump |dbo:spouse   | dbr:Melania_Trump|
|dbr:Donald_Trump |dbo:birthDate| 1946-06-14       |
|dbr:Olaf_Scholz  |dbo:spouse   | dbr:Britta_Ernst |
|dbr:Olaf_Scholz  |dbo:birthDate| 1958-06-14       |
+-----------------+-------------+-----------------+
```

```
+-----------------+------------+-----------------+
|Subject          |dbo:birthDate|dbo:spouse       |
+-----------------+------------+-----------------+
|dbr:Donald_Trump |1946-06-14  |dbr:Melania_Trump|
|dbr:Olaf_Scholz  |1958-06-14  |dbr:Britta_Ernst |
+-----------------+------------+-----------------+
```

Listing 1: RDF data pivoting

(LHD) [51]. LHD contains types from the DBpedia namespace that have been extracted from the opening sentences of Wikipedia articles written in various languages. The identification of these types was accomplished by employing Hearst pattern matching on the text annotated with part-of-speech tags, and then disambiguating them to align with DBpedia concepts. Moreover, [25] used Locality Sensitive Hashing (LSH) [52] for creating clusters (they called the clusters cohorts because the same data can appear in more than one cluster). LSH is an important class of hashing techniques that hashes data points into buckets, so that the data points which are close to each other are in the same buckets with high probability, while data points that are far away from each other lie in different buckets. [25] used `rdf:type` and LHD information to create vectors and then hashed it based on LSH and further generated cohorts.

4) Clustering based on Semantic Features: Most clustering algorithms require a mechanism to calculate the similarity between different data points. In [24], authors used a mechanism to incorporate semantic features for calculating similarity called DistSim [53]. DistSim has different modes for calculating semantic similarity. For instance considering outgoing links, incoming links, a combination of predicates and objects, and many more. One intuitive way of calculating similarity for clustering entities could be using the predicates as main features (Outgoing Relation mode in [53]). In short, in this mode, two entities will be similar if they share many common predicates. This aids the clustering algorithm in grouping similar entities.

C. Anomaly Detection Algorithms

As already mentioned, most of the existing works focus on univariant anomaly detection methods (except [24], [29]). So in this section, we briefly summarize the already in-use anomaly detection algorithms in [24]–[29].

1) Interquartile Range: The IQR [54] technique is a statistical metric that is based on calculating the first quartile ($Q1$), the median ($Q2$), and the third quartile ($Q3$) of a numerical dataset. The difference between $Q3$ and $Q1$ is called IQR. Outliers are data points that are less than $Q1 - 1.5 \times IQR$ and more than $Q3 + 1.5 \times IQR$.

2) Median Absolute Deviation: MAD [55] is a measure of the variability of a univariate sample of numeric data. The MAD for a data collection $X = \{x_1, x_2, ..., x_n\}$ is defined as the median of the absolute deviations from the median of the data. So if $\widetilde{x} = median(X)$ then: $MAD = b \times median(|x_i - \widetilde{x}|)$ where b is a constant that changes with the distribution. The values in X that are more than $\widetilde{x} + 2.5 \times MAD$ and less than $\widetilde{x} - 2.5 \times MAD$ are outliers. MAD is more resistant to data set outliers than the standard deviation technique.

3) Z-Score: Z-Score is the number of standard deviations by which the value of a raw score (i.e., an observed value or data point) is above or below the mean value of what is being observed or measured. Positive standard scores are assigned to raw scores that are greater than the mean, while negative standard scores are assigned to raw scores that are less than the mean. For example, a Z-Score of 1.5 indicates that the data point is 1.5 units away from the mean, indicating that it can be an outlier. The Z-Score is defined as: $z - score = \frac{x-\mu}{\sigma}$ where μ is the mean of the data and σ is the standard deviation.

4) Kernel Density Estimation: KDE [56] is a non-parametric approach for estimating the probability density function of a random variable. The kernel density estimator of density f is: $\hat{f}_h(x) = \frac{1}{n \times h} \sum_{i=1}^{n} K(\frac{x-x_i}{h})$ where K is a non-negative function, and $h > 0$ is a smoothing parameter. To obtain outlier scores for a given dataset, firstly a KDE should be constructed from the data and then the resultant probability at each point should be calculated. To put this probability into context, it should be compared to the mean probability across all points. Then a predefined threshold can be used to produce a binary classification.

5) Local Outlier Factor: LOF [35] is one of the density-based anomaly detection approaches. The fundamental idea behind the local outlier factor revolves around local density. In this context, locality is determined by considering the k nearest neighbors, and their inter-distance is utilized to approximate the density. Regions with similar densities and spots with much lower densities than their neighbors can be detected by comparing an object's local density to the local densities of its neighbors. These are known as outliers.

6) Global Anomaly Score: GAS is one of the most frequently used nearest-neighbor algorithms. The anomaly score is either set to the average distance of the k nearest neighbors, as recommended in [34], or to the distance to the k^{th} neighbor, as proposed in [33]. It is worth noting that the first strategy is far more resistant to statistical variations.

7) One-Class SVM: One-class SVM is a multi-variant anomaly detection algorithm which unlike traditional SVM aims to develop a decision boundary that produces the greatest separation between the points and the origin [57]. One-class SVM projects data into a higher dimensional space using the kernel's implicit transformation function, $\varphi(\cdot)$. The algorithm then learns the decision boundary (a hyperplane) that separates the bulk of the data from the origin. Only a few data points are allowed to fall on the opposite side of the decision boundary; these data points are known as outliers.

8) Isolation Forest: IF [58] is also a multi-variant anomaly detection algorithm that identifies anomalies through isolation. Same as Random Forests, Isolation Forest is built upon decision trees and generates an ensemble of isolation trees from the training data. The Isolation Forest approach is based on the fact that anomalous examples in a dataset are easier to isolate (separate) from the rest of the regular points. In IF, randomly sub-sampled data is processed in a tree structure based on randomly selected features. Anomalies are less likely to arise in greater-depth samples because they require more cuts to be separated. Similarly, samples that end up in shorter branches indicate anomalies.

V. Existing Works

So far the Anomaly Detection workflow, techniques, and approaches described. Now in this section, we cover the existing works which have been carried on the area of AD over KGs.

One of the early works in the area of detecting incorrect numerical data in DBpedia is [26]. The authors argued that the traditional outlier detection approaches are limited by the existence of natural outliers and performed the process of finding numerical outliers in two steps. In the first step, all types of an entity are considered as a vector of boolean values (one-hot encode), representing whether or not the entity is of a certain *type*. The authors used the FeGeLOD framework for vectorizing the entities and did the clustering with the Estimation Maximization (EM) algorithm [49], using the implementation in WEKA [59]. In the next step, the outliers are detected. The authors have compared different outlier detection techniques, such as IQR, KDE, and dispersion estimators, and reported that IQR performs the best. In addition, they reported that the run-time on datasets containing only two properties-*DBpedia-owl:populationTotal* and *DBpedia-owl:elevation* is over 24 hours due to the slow clustering algorithm.

In another work [28], which is close to [26], an outlier detection method is introduced that cross-checks the results of outliers by exploiting the *"sameAs"* properties in the knowledge graph. Outlier detection is accomplished through dataset inspection using specialized SPARQL queries against the knowledge graph. The authors begin by selecting the interesting properties for outlier detection. The sub-population is generated in the second step by applying a set of constraints (top-down ILP algorithms) to classes, properties, and property values. This exploration is laid out as a lattice, with the root node consisting of a property and the number of instances that correspond to it. After the lattice has been generated, the outliers on all unpruned nodes of the lattice must be found. The outlier score results are saved as a set of constraints that returns the corresponding instance set. Outliers are classified as natural or real using the data interlinking property and comparison with different datasets. This procedure improves the handling of natural outliers, lowering the false positive rate.

CONOD [25] is a scalable and generic algorithm for numeric outlier detection for DBpedia. It utilized *rdf:type* and Linked Hypernyms Dataset (LHD) [51] for creating cohorts. Cohorts, unlike clusters, could overlap with each other. For cohorts, [25] used a scalable clustering approach based on Locality Sensitive Hashing (LSH) [52]. This approach has been developed over Apache Spark therefore it can be applied to very large KGs, however, as the authors used *rdf:type* and LHD, this approach is only applicable to DBpedia. Moreover, this approach has been integrated into the Scalable Semantic Analytics Stack (SANSA) [60], which is a framework built on top of Apache Spark. It offers fault-tolerant, highly available, and scalable methods for efficiently processing RDF data while supporting semantic technology standards.

DistAD [24] is another generic, scalable, and distributed framework for anomaly detection on large RDF knowledge graphs which exploits Apache Spark. DistAD has been designed to handle very large KGs.Additionally, it offers end-users a range of options to choose from, including various algorithms, methods, and hyperparameters, for detecting outliers on KGs. This approach supports multiple feature extraction methods such as Literal2Feature [47] and Pivoting, multiple clustering algorithms such as BiSecting Kmeans [61] and MinHashLSH[5]. Moreover, it provides univariant anomaly detection methods such as IQR, MAD, and Z-Score, and multi-

[5]https://spark.apache.org/docs/latest/ml-features#minhash-for-jaccard-distance

variant Isolation Forest [58]. Moreover, this approach has been also integrated into the SANSA Stack.

Due to the lack of explainability in previous works, ExPAD [62] has been introduced to bridge this gap. It represents an improvement over DistAD, as it can generate human-readable explanations for why a specific numerical result is considered an outlier. ExPAD utilizes decision trees and Apache Spark to partition and handles large knowledge graphs, respectively. By applying an anomaly detection method, such as IQR, to the partitioned data and identifying anomalies, ExPAD generates explanations for why a given value of the target variable may be considered an outlier, by considering the decision tree branches and their associated variables.

Table II summarizes and compares the existing works and their main characteristics.

VI. Discussion, Challenges, and Conclusion

Although there is plenty of research in different areas of anomaly detection, in this paper we focused on providing a concise overview of existing methodologies and techniques specifically tailored for identifying anomalies in Knowledge Graphs involving numeric literals. In the concluding section, we highlight the challenges encountered in this field and discuss potential avenues for future research.

a) Comprehensiveness: As [19], [21] introduced, there are many types of errors that may occur in KGs however, the existing works tried to address mostly the errors in numerical literals. Thus one of the working areas could be handling other types of errors such as temporal anomalies, anomalies on `subject`, anomalies on `predicate`, semantic anomalies, ...

b) Graph-based Approaches: Most of the exciting works focused on statistical univariant anomaly detection techniques. However, KGs are heterogeneous complex graphs with labeled edges (predicates). Therefore, applying graph-based anomaly detection techniques [25], [63] to KGs could be an interesting research area.

c) Scalability: The size of the RDF dataset is growing substantially nowadays. However, most of the exciting works (except [24], [25], [62]) did not consider this fact and they fail on large KGs such as DBpedia. So the need of having scalable anomaly detection methods for large KGs is a necessity.

d) Explainability: Explainability is an important factor for a robust anomaly detection technique. Having human-readable explanations for why a given value of a variable in an observation is an outlier could be beneficial. Unfortunately, most of the existing works (except [62]) do not offer explainability for the detected anomalies.

e) Lack of Benchmarks: One of the main reasons why anomaly detection over KGs has not gained adequate attention is the lack of benchmarks and ground truths. To the best of our knowledge, there is no publicly available labeled dataset for anomaly detection in the RDF format. Introducing such a public benchmark can boost this research field.

f) Streaming: None of the existing methods support data streaming, which is frequently utilized in industrial scenarios.[6] Therefore, having an anomaly detection approach that can handle RDF data streams would provide significant benefits.

[6]https://platoon-project.eu

References

[1] T. Berners-Lee, "A roadmap to the semantic web," 1998.
[2] C. Bizer, M.-E. Vidal, and et al, *Linked Open Data*, 2018.
[3] J. M. Juran, *Juran's Quality Control Handbook*, 4th ed. Mcgraw-Hill (Tx), 1974.
[4] D. Vrandecic, "Wikidata: a new platform for collaborative data collection," in *WWW*, 2012.
[5] K. Bollacker, C. Evans, and et al, "Freebase: A collaboratively created graph database for structuring human knowledge," in *SIGMOD*, 2008.
[6] T. Mitchell, W. Cohen, and et al, "Never-ending learning," in *AAAI-15*, 2015.
[7] S. Auer, C. Bizer, G. Kobilarov, J. Lehmann, R. Cyganiak, and Z. G. Ives, "Dbpedia: A nucleus for a web of open data," in *The Semantic Web, 6th International Semantic Web Conference*, ser. Lecture Notes in Computer Science, vol. 4825. Springer, 2007, pp. 722–735.
[8] F. M. Suchanek, G. Kasneci, and G. Weikum, "Yago: a core of semantic knowledge," in *Proceedings of the 16th International Conference on World Wide Web*. ACM, 2007, pp. 697–706.
[9] V. Chandola and et al, "Anomaly detection: A survey," *ACM Comput. Surv.*, 2009.
[10] C. C. Aggarwal, "Outlier ensembles: Position paper," *SIGKDD Explor. Newsl.*, 2013.
[11] C. Xiong, R. Power, and J. Callan, "Explicit semantic ranking for academic search via knowledge graph embedding," in *WWW*, 2017.
[12] S. Ji and et al, "A survey on knowledge graphs: Representation, acquisition, and applications," *IEEE Transactions on Neural Networks and Learning Systems*, 2022.
[13] F.-L. Li, H. Chen, and et al, "Alimekg: Domain knowledge graph construction and application in e-commerce," in *CIKM*, 2020.
[14] A. Zaveri, A. Rula, and et al, "Quality assessment methodologies for linked open data," *Semantic Web Journal*, 2013.
[15] B. Stvilia, L. Gasser, M. B. Twidale, and L. C. Smith, "A framework for information quality assessment," *J. Assoc. Inf. Sci. Technol.*, 2007.
[16] A. Zaveri, A. Rula, and et al, "Quality assessment for linked data: A survey," *Semantic Web*, 2016.
[17] M. Färber, F. Bartscherer, and et al, "Linked data quality of dbpedia, freebase, opencyc, wikidata, and YAGO," *Semantic Web*, 2018.
[18] V. Jayawardene, S. Sadiq, and M. Indulska, "An analysis of data quality dimensions," *ITEE Tech*, 2015.
[19] X. Wang, L. Chen, T. Ban, M. Usman, Y. Guan, S. Liu, T. Wu, and H. Chen, "Knowledge graph quality control: A survey," *Fundamental Research*, 2021.
[20] R. Y. Wang and D. M. Strong, "Beyond accuracy: What data quality means to data consumers," *J. Manag. Inf. Syst.*, 1996.
[21] A. Zaveri, D. Kontokostas, and et al, "User-driven quality evaluation of dbpedia." in *I-SEMANTICS*, 2013.
[22] F. Naumann, *Quality-Driven Query Answering for Integrated Information Systems*. Springer-Verlag, 2002.
[23] Z. H. Syed, M. Röder, and A.-C. Ngonga Ngomo, "Factcheck: Validating rdf triples using textual evidence," in *CIKM*, 2018.
[24] F. B. Moghaddam, J. Lehmann, and H. Jabeen, "Distad: A distributed generic anomaly detection framework over large kgs," in *ICSC*, 2022.
[25] H. Jabeen, R. Dadwal, and et al, "Divided we stand out! forging cohorts for numeric outlier detection in large scale knowledge graphs (CONOD)," in *EKAW*, 2018.
[26] D. Wienand and H. Paulheim, "Detecting incorrect numerical data in dbpedia," in *The Semantic Web: Trends and Challenges*, 2014.
[27] H. Paulheim and J. Fürnkranz, "Unsupervised generation of data mining features from linked open data," in *WIMS*, 2012.
[28] D. Fleischhacker, H. Paulheim, and et al, "Detecting errors in numerical linked data using cross-checked outlier detection," in *ISWC*, 2014.
[29] H. Paulheim, "Identifying wrong links between datasets by multi-dimensional outlier detection." in *WoDOOM*, ser. CEUR Workshop Proceedings. CEUR-WS.org, 2014.
[30] D. Hawkins, *Identification of Outliers*. Chapman and Hall, 1980.
[31] A. Charu C., *Outlier Analysis.* Springer, 2013.
[32] G. Pang, C. Shen, and et al, "Deep learning for anomaly detection: A review," *ACM Comput. Surv.*, 2021.

TABLE II: Existing works of AD on KGs and their characteristics

		[26]	[28]	[25]	[24]	[62]
AD Level		Numeric Literals	Numeric Literals	Numeric Literals	Numeric Literals + Number of Predicates	Numeric Literals
AD Algorithm	Univariant	✓	✓	✓	✓	✓
	Multi-variant	×	×	×	✓	×
	Graph-based	×	×	×	×	×
	Statistical	✓	✓	✓	✓	✓
	Supervised	×	×	×	×	×
	Semi-supervised	×	×	×	×	×
	Unsupervised	✓	✓	✓	✓	✓
	Algorithm	IQR MAD KDE	IQR LOF	IQR	IQR MAD Z-Score Isolation Forest	IQR MAD Z-Score
Feature Extractor		FeGeLOD	SPARQL	-	Literal2Feature Pivoting	Literal2Feature Pivoting
Clustering		rdf:type + EM	constraint + lattice	LHD + Cohorting	DistSim + Bisecting Kmeans	Decision Tree
Scalability		×	×	✓	✓	✓
Explainability		×	×	×	×	✓
Streaming		×	×	×	×	×

[33] S. Ramaswamy, R. Rastogi, and K. Shim, "Efficient algorithms for mining outliers from large data sets." in *SIGMOD Conference*. ACM, 2000.

[34] F. Angiulli and C. Pizzuti, "Fast outlier detection in high dimensional spaces." in *PKDD*, 2002.

[35] M. M. Breunig, H.-P. Kriegel, and et al, "Lof: Identifying density-based local outliers." in *SIGMOD*, 2000.

[36] C. C. Aggarwal and P. S. Yu, "Outlier detection for high dimensional data," *SIGMOD Rec.*, 2001.

[37] A. Zimek, E. Schubert, and H.-P. Kriegel, "A survey on unsupervised outlier detection in high-dimensional numerical data." *Stat. Anal. Data Min.*, 2012.

[38] J. Zhang, Y. Jiang, and et al, "A concept lattice based outlier mining method in low-dimensional subspaces," *Pattern Recognit. Lett.*, 2009.

[39] J. K. Dutta, B. Banerjee, and C. K. Reddy, "Rods: Rarity based outlier detection in a sparse coding framework." *IEEE Trans. Knowl. Data Eng.*, 2016.

[40] J. Zhang, S. Zhang, K. H. Chang, and X. Qin, "An outlier mining algorithm based on constrained concept lattice." *Int. J. Systems Science*, 2014.

[41] C. C. Aggarwal and P. S. Yu, "An effective and efficient algorithm for high-dimensional outlier detection." *VLDB J.*, 2005.

[42] A. Lazarevic and V. Kumar, "Feature bagging for outlier detection." in *KDD*, 2005.

[43] V. N. P. Kappara, R. Ichise, and O. P. Vyas, "Liddm: A data mining system for linked data," in *WWW*, ser. CEUR Workshop Proceedings. CEUR-WS.org, 2011.

[44] W. Cheng, G. Kasneci, and et al, "Automated feature generation from structured knowledge," in *CIKM*, 2011.

[45] M. Khan, G. Grimnes, and A. Dengel, "Two pre-processing operators for improved learning from semanticweb data," in *RCOMM*, 2010.

[46] P. Ristoski, C. Bizer, and H. Paulheim, "Mining the web of linked data with rapidminer," *J. Web Semant.*, 2015.

[47] F. B. Moghaddam, C. F. Draschner, J. Lehmann, and H. Jabeen, "Literal2feature: An automatic scalable rdf graph feature extractor," in *SEMANTICS*, 2021.

[48] E. Moore, *The Shortest Path Through a Maze*, ser. Bell Telephone System. Technical publications. monograph. Bell Telephone System., 1959.

[49] A. P. Dempster, N. M. Laird, and D. B. Rubin, "Maximum likelihood from incomplete data via the em algorithm," *Journal of the Royal Statistical Society, Series B*, 1977.

[50] G. Ball and D. Hall, "Isodata: A novel method of data analysis and pattern classification," Stanford Research Institute, Tech. Rep., 1965.

[51] T. Kliegr, "Linked hypernyms: Enriching dbpedia with targeted hypernym discovery," *Journal of Web Semantics*, 2015.

[52] M. Datar, N. Immorlica, and et al, "Locality-sensitive hashing scheme based on p-stable distributions," in *SCG*, 2004.

[53] C. F. Draschner, J. Lehmann, and H. Jabeen, "Distsim-scalable distributed in-memory semantic similarity estimation for rdf knowledge graphs," in *ICSC*, 2021.

[54] R. McGill, J. W. Tukey, and W. A. Larsen, "Variations of box plots," *The American Statistician*, 1978.

[55] C. Leys, C. Ley, and et al, "Detecting outliers: Do not use standard deviation around the mean, use absolute deviation around the median," *Journal of Experimental Social Psychology*, 2013.

[56] E. Parzen, "On estimation of a probability density function and mode," *The Annals of Mathematical Statistics*, 1962.

[57] B. Schölkopf, J. C. Platt, and et al, "Estimating the support of a high-dimensional distribution," *Neural Computation*, 2001.

[58] T. Zemicheal and T. G. Dietterich, "Anomaly detection in the presence of missing values for weather data quality control," in *SIGCAS*, 2019.

[59] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, and I. H. Witten, "The weka data mining software: an update," *SIGKDD Explor. Newsl.*, 2009.

[60] J. Lehmann, G. Sejdiu, and et al, "Distributed semantic analytics using the sansa stack," in *ISWC*, 2017.

[61] M. Steinbach, G. Karypis, and V. Kumar, "A comparison of document clustering techniques," in *In KDD Workshop on Text Mining*, 2000.

[62] F. B. Moghaddam, J. Lehmann, and H. Jabeen, "Expad: An explainable distributed automatic anomaly detection framework over large kgs," in *ICSC*, 2023.

[63] L. Akoglu, H. Tong, and D. Koutra, "Graph based anomaly detection and description: a survey." *Data Min. Knowl. Discov.*, 2015.