

Multi-Source Data Repairing: A Comprehensive Survey

Chen Ye ^{1,2,3} , Haoyang Duan ¹, Hengtong Zhang ⁴, Hua Zhang ^{1,*}, Hongzhi Wang ⁵  and Guojun Dai ^{1,*}¹ Hangzhou Dianzi University, Hangzhou 310018, China² Nanjing University of Aeronautics and Astronautics, Nanjing 210016, China³ Jubang Group Co., Ltd., Yueqing 325600, China⁴ Tencent AI Lab, Shenzhen 518054, China⁵ Harbin Institute of Technology, Harbin 150001, China

* Correspondence: zhangh@hdu.edu.cn (H.Z.); daigj@hdu.edu.cn (G.D.)

Abstract: In the era of Big Data, integrating information from multiple sources has proven valuable in various fields. To ensure a high-quality supply of multi-source data, repairing different types of errors in the multi-source data becomes critical. This paper categorizes errors in multi-source data into entity information overlapping, attribute value conflicts, and attribute value inconsistencies. We first summarize existing repairing methods for these errors and then examine and review the study of the detection and repair of compound-type errors in multi-source data. Finally, we indicate further research directions in multi-source data repair.

Keywords: multiple sources; data quality; data repairing; entity resolution; truth discovery; data dependencies

MSC: 68P20; 68T07



Citation: Ye, C.; Duan, H.; Zhang, H.; Zhang, H.; Wang, H.; Dai, G. Multi-Source Data Repairing: A Comprehensive Survey. *Mathematics* **2023**, *11*, 2314. <https://doi.org/10.3390/math11102314>

Academic Editors: Shih-Wei Lin and José Antonio Sanz

Received: 22 April 2023

Revised: 8 May 2023

Accepted: 9 May 2023

Published: 16 May 2023



Copyright: © 2023 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

In recent years, extensive data sources, including social media, crowdsourcing platforms, and ubiquitous sensor networks, continuously generate vast amounts of data. Conducting data analytics on such massive amounts of data can benefit decision-making and knowledge discovery. Often, data points describing the same object or event originate from multiple sources, resulting in multi-source data. For example, descriptions of a particular item may vary across different commercial platforms. Consequently, multi-source data inevitably have different forms of representation and various errors, which can impact their overall value. To ensure the provision of high-quality multi-source data, it is essential to detect and rectify inconsistencies and mistakes.

Traditional data quality problems are identified according to five dimensions [1]: data consistency [2], information completeness [3,4], data deduplication [5,6], data currency [7,8], and data accuracy [9]. The existing data repairing literature has been mostly focused on data deduplication in multi-source data while addressing other dimensions for the single source/database. Although methods for single-source data can be applied to multi-source data by resolving data quality issues within each source separately, the task of repairing multi-source data differs from that of single-source data. Single-source repairing aims to achieve complete and consistent data through error detection and repair according to the dimensions above. In contrast, multi-source repairing seeks reliable integrated information by repairing conflicts and inconsistencies among data sources, presenting new challenges. Compared to data repairing methods for single sources, techniques for multi-source data face the following data quality problems:

(1) Due to varying representations and formats of entities among multiple data sources, the primary challenge of multi-source repair is to identify overlapping data, particularly by matching records that refer to the same real-world entity. This task involves purging

duplicate information and consolidating and merging complementary information to achieve a consistent view of real-world entities.

(2) For a given entity, each attribute value may have observations provided by a certain number of data sources. However, multi-source conflicts may arise due to recording errors, intentional errors, conflicts, and outdated data across different data sources. To address this issue, it is crucial to determine the correct attribute values for each entity among the conflicted observations.

(3) In addition to multi-source conflicts, inconsistencies may occur among multi-source observations of different entities. Here, data consistency refers to the validity and integrity of attribute values for these entities. To ensure the quality of integrated data, it is necessary to design robust integrity constraints and practical algorithms for detecting and repairing inconsistencies within multi-source data.

To summarize, besides the single-source problems in the multi-source case, the specific data quality issues in multi-source data come from three major categories: entity information overlapping, attribute value conflicts, and attribute value inconsistency [10]. As shown in Figure 1, the existing work usually resolves the entity information overlapping by entity resolution over the multi-source instances. The detection strategy is blocking, and the repair strategy is matching. For multi-source attribute value conflicts, the general solution is truth discovery, which detects inter-source conflicts and resolves them by evaluating the reliability of the data sources. For attribute information inconsistencies among different entities, existing approaches use various integrity constraints to detect and find the “least costly” consistent repair.

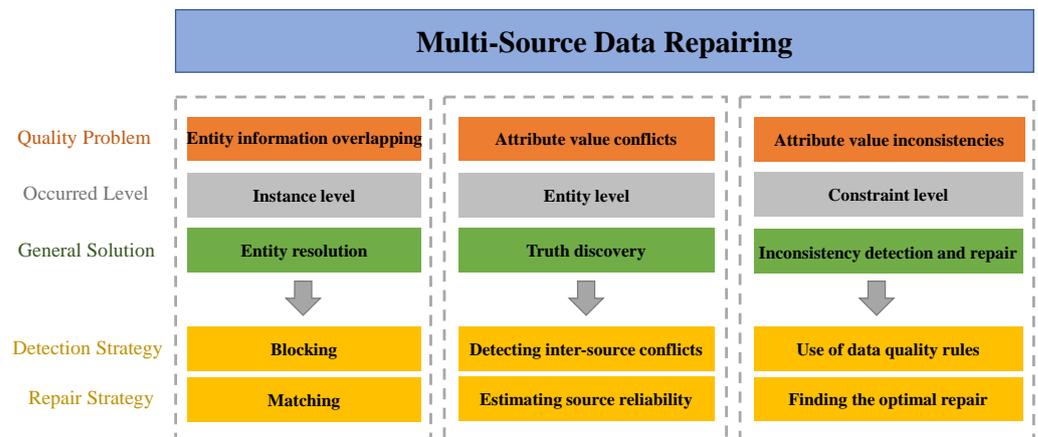


Figure 1. Multi-source data repairing methods.

This paper comprehensively reviews recent research in entity resolution (Section 2), truth discovery (Section 3), and inconsistency detection and repair (Section 4). As the types of errors mentioned above may exist simultaneously and influence each other, we also examine and review the study of compound error detection and repair (Section 5). Finally, we indicate further research directions in multi-source data repairing (Section 6).

2. Entity Resolution

Different data providers may have different descriptions of the same thing (including data formats and representations), and each description of an entity is called a reference to that entity. Entity resolution (ER) is the process of resolving and mapping real-world entities from a “collection of references” [11]. ER is also known as record linkage, object identification, individual identification, and duplicate detection. Intuitively, it is computationally expensive to directly match each pair of records in two data sources by calculating the similarity pair by pair. To avoid the excessive complexity generated by direct matching, the existing work typically divides ER into two steps [12]: (1) blocking and (2) matching.

Specifically, the blocking step divides records that do not represent the same entity into different blocks so that only records within the same block need to be matched, avoiding comparing many records between blocks. Matching refers to determining whether two representations refer to the same real-world entity. For example, whether two records (**Name:** *Kristen Smith*, **Street:** *2 Hurley Pl*, **City:** *South Fork, MN 48503*, **Sex:** *Female*) and (**LastName:** *Smith*, **FirstName:** *Kristen L.*, **Gender:** *F*, **Address:** *2 Hurley Place, South Fork MN, 48503-5998*) refer to the same person *Kristen Smith*. Next, we review recent ER approaches and discuss them in two types: learning-based methods and non-learning-based methods.

2.1. Non-Learning-Based Methods

Most of the non-learning approaches were proposed before artificial intelligence (AI) became widespread. The methods in this category are summarized in Table 1. For a better understanding, we organize them into a novel taxonomy that consists of *Schema-type*, *Step*, and *Knowledge-based*, and detail the related papers according to *Schema-type*.

- *Schema-type*: distinguishes between *schema-aware* or *schema-agnostic* methods. The former indicates that the method selects some specific attribute values for blocking and matching, and these selected attributes are discriminative or contain less noisy data. The latter does not consider pattern information and extracts information from all attribute values.
- *Step*: divides the methods into three categories based on the steps included. *Blocking* means that the method contains only blocking steps, *matching* means that the process consists of only matching, *blocking–matching* implies that the method proposes a complete framework for ER, including both blocking and matching.
- *Knowledge-based*: classifies the methods into *yes*, which means that external knowledge, such as knowledge bases, rules, constraints, etc., are introduced into the techniques, and *no*, which means that they are not.

Table 1. Non-learning-based methods.

Method	Schema Type	Step	Knowledge-Based
Standard blocking [5]	Schema-aware	Blocking	No
Sorted blocking [13]	Schema-aware	Blocking	No
CrowdER [14]	Schema-aware	Blocking–matching	No
Swoosh [15]	Schema-aware	Matching	No
SEMR [16]	Schema-aware	Matching	Yes
Rule-based ER [17]	Schema-aware	Blocking–matching	Yes
DMCE [18]	Schema-aware	Matching	Yes
Token blocking [19]	Schema-agnostic	Blocking	No
Attribute clustering blocking [20]	Schema-agnostic	Blocking	No
SiGMa [21]	Schema-agnostic	Matching	No
MinoanER [22]	Schema-agnostic	Blocking–matching	No

Schema-aware. Methods of this type assume that the input entity profiles adhere to a known schema. Based on this schema and respective domain knowledge, one can select the attributes most suitable for blocking and matching. We then discuss precisely what methods are available in this category [5,13–18].

In terms of blocking, standard blocking [5] is the basis of the blocking algorithm, which represents each entity with one or more key values. Each block corresponds to a

specific key value and contains all the entities represented by that key value. It involves the most straightforward functionality: an expert selects the most suitable attributes, and a transformation function concatenates (parts of) their values to form blocking keys. For every distinct key, a block is created containing all corresponding entities. Comparing records in the same block can reduce the number of record comparisons in the original ER.

Through combining the standard blocking and sort technologies, ref. [13] proposes an entity resolution blocking technology called sorted blocking. Sorted blocking sorts all blocking key values according to dictionary order. The ordered entities are then divided into blocks based on the prefixes of the blocking keys, and the records within one block are compared. In addition, the algorithm uses a windowing technique to avoid losing any matches. Records from different blocks in the window are also involved in the comparison calculation. The sorted blocking algorithm does not limit the block size, which can lead to large blocks taking up a significant amount of processing time.

In terms of matching, swoosh [15] focuses on “pairwise ER”, which matches and merges records operating on two records at a time. Swoosh defines two functions, match and merge, where match identifies duplicates and merge combines two duplicate records into one. There are also methods [14,16–18] for obtaining matching results for record pairs by introducing knowledge of manual annotation information, knowledge bases, rule constraints, etc., through knowledge-based approaches.

SEMR [16] studies how to synthesize matching rules from positive–negative matching examples. It presents a system that introduces real-world associated constraints to improve the accuracy of the synthesized matching rules. This system matches the performance of machine learning (ML) methods and produces concise rules. Rule-based ER [17] considers that records referring to the same entity observed in different periods may be different. The authors use data quality rules, such as matching dependency and data currency, to derive temporal records’ information in the time order and the trend of their attributes’ evolution with time elapsing. Based on the obtained information, a temporal-based clustering method is proposed to improve the accuracy of matching on datasets containing hidden temporal information.

DMCE [18] focuses on entity classification, which is relevant but not identical to matching. They propose a rule-based framework using positive and negative rules to discover miscategorized entities. Positive rules are conservatively used to find disjoint partitions, so the entities within the same partition should be categorized together. The partition with the largest size is called the pivot partition under the realistic assumption that the largest partition is correct. Negative rules are used to compare other partitions with the pivot partition to discover dissimilar entities as miscategorized entities.

In addition to the above work, a more specific type of crowdsourced blocking uses machine-based techniques to remove evident non-repetitive entities while using valuable human resources to examine situations that require human insight. Following this idea, CrowdER [14] proposes a hybrid human–machine ER approach, which first uses machine-based techniques to label the non-similar record pairs and asks the crowd to verify only the remaining pairs.

Schema-agnostic. Methods of this type make no assumptions about schema knowledge, disregarding complete attribute names. We discuss several representatives [19–22] in this category.

In essence, token blocking (TB) [19] is based on the idea that profiles of duplicate entities have at least one value in common, independently of the corresponding attribute names. Therefore, all entities that contain the same token in any attribute of their profile are placed in one block, resulting in redundancy that each entity is contained in multiple blocks. To improve TB, attribute clustering blocking [20] requires the common tokens of two entities to appear in syntactically similar attributes. These attribute names correspond to similar values but are not necessarily semantically matching (unlike schema matching). First, it clusters attributes based on the similarities of their aggregate values. Each attribute is connected to its most similar one, and the transitive closure of the connected attributes

forms disjoint clusters. A $block_{k,t}$ is then defined for every token t in the values of the attributes belonging to cluster k .

Next, we introduce two schema-agnostic methods for matching. SiGMa [21] selects as seed matches the pairs with identical entity names. Then, it propagates the matching decisions on the compatible neighbors of existing matches. Unique mapping clustering is applied for detecting duplicates. That is, for every newly matched pair, the similarities of the neighbors are recomputed, and their position in the priority queue is updated. SiGMa is an iterative propagation algorithm that leverages structural information from the relationship graph and flexible similarity measures between entity properties in a greedy local search, making it scalable. MinoanER [22] also employs unique mapping clustering. However, it differs from SiGMa. MinoanER leverages a token-based similarity of entities to define a new metric that derives the similarity of neighboring entities from the most important relations. A composite blocking method is employed to capture different sources of matching evidence from entities' content, neighbors, or names. The search space of candidate pairs for comparison is compactly abstracted by a novel disjunctive blocking graph and processed by a non-iterative, massively parallel matching algorithm that consists of four generic, schema-agnostic matching rules.

2.2. Learning-Based Methods

With the development of AI, learning methods are used to solve ER problems. We summarize these methods in Table 2, classified according to *step*, *knowledge-based*, *learning type*, and *pre-trained LM*. As *step* and *knowledge-based* are introduced in Section 2.1, we detail *learning type* and *pre-trained LM* as follows:

- *Learning type*: distinguishes these methods into *supervised*, *unsupervised*, *semi-supervised*, and *deep learning*, where we uniform all the learning-based methods belonging to the field of deep learning (DL) to *deep learning*.
- *Pre-trained LM*: Identifying and distinguishing entities requires capturing their semantic similarities, which requires significant language understanding and domain knowledge. Pre-trained transformer-based language models (LMs) perform well on the ER task due to their ability to understand language and their ability to learn where to pay attention. This dimension divides methods into *yes* and *no*. The former indicates introducing a pre-trained LM, while the latter does not.

Table 2. Learning-based methods.

Method	Step	Knowledge-Based	Learning Type	Pre-Trained LM
ApproxDNF [23]	Blocking	No	Supervised	No
BSL [24]	Blocking	No	Supervised	No
BGP [25]	Blocking	No	Supervised	No
CBLOCK [26]	Blocking	No	Supervised	No
KBPearl [27]	Matching	Yes	Supervised	No
CEA [28]	Matching	Yes	Supervised	No
AL-EM [29]	Matching	No	Supervised	No
Semi-supervised de-dup [30]	Matching	No	Semi-supervised	No
Map-Reduce-Based EI [31]	Matching	No	Unsupervised	No
ULEC [32]	Matching	No	Unsupervised	No

Table 2. Cont.

Method	Step	Knowledge-Based	Learning Type	Pre-Trained LM
ZeroER [33]	Matching	No	Unsupervised	No
RER [34]	Matching	Yes	Unsupervised	No
PERC [35]	Matching	Yes	Unsupervised	No
DeepER [36]	Blocking–matching	No	Deep learning	No
DeepMatcher [37]	Matching	No	Deep learning	No
DeepBlocking [38]	Blocking	No	Deep learning	No
DITTO [39]	Blocking–matching	Yes	Deep learning	Yes
Machop [40]	Matching	Yes	Deep learning	Yes
JointMatcher [41]	Matching	No	Deep learning	Yes

Supervised learning methods. Supervised learning is a type of machine learning where the training data usually contain features and labels. A function is then learned by mapping a feature vector to a label. Supervised learning in ER is generally conducted by obtaining feature vectors from matched/unmatched entity pairs and then understanding the mapping relationship between the vector pairs and the matched/unmatched labels. We next describe several representatives [23–29] of supervised learning.

Regarding blocking, ApproxDNF [23] uses small record clusters as samples to train an algorithm. The algorithm learns specific features from the data based on these samples, producing higher-quality results. The training instances for the learning process are pairs of records. These instances are used in the rule selection training analysis, which produces the best record sets in blocking. Then, these selected rules are combined to form expressions in a disjunctive normal form (DNF), which defines how records are blocked.

BSL blocking [24] is similar to ApproxDNF in its use of blocking rules. The main difference is how the training samples are used and how predicates are combined to produce the blocking strategy during the learning process. In practice, BSL blocking is usually used for samples containing a small number of positive training record pairs. As a result, the execution time may be less than the time taken to perform ApproxDNF.

Different from the above methods, BGP [25] uses genetic programming (GP) as the basis for solving adaptive blocking problems. BGP uses a tree representation of supervised blocking schemes, where each leaf node corresponds to a blocking predicate. In each round, a set of genetic programming operators, such as replication, mutation, and crossover, are applied to an initial, random block scheme set. Then, a fitness function infers the performance of the new schemes from the harmonic mean of genuine coverage of pairs (PC) and reduction ratio in the number of candidate pairs (RR), and the best ones are returned as output.

Another tree-based approach is CBLOCK [26], which automatically learns hash functions from attribute domains and a labeled dataset of duplicates. Subsequently, CBLOCK expresses blocking functions using a hierarchical tree structure composed of atomic hash functions. It guides the automated blocking process based on architectural constraints, such as by specifying the maximum size of each block. Each node that exceeds the maximum limit is split into smaller, disjoint blocks.

In terms of matching, KBPearl [27] and CEA [28] are proposed based on knowledge bases. KBPearl [27] constructs a semantic graph based on the knowledge extracted from the canonicalized facts and the side information. Then, KBPearl matches various entities for disambiguating the noun phrases and relation phrases in the facts by determining a dense subgraph from the semantic graph efficiently and effectively. CEA [28] identifies entities that refer to the same real-world object but locate in different knowledge graphs (KGs). It uses structural, semantic, and string signals to capture similarities between entities in heterogeneous knowledge graphs.

AL-EM [29] builds a unified active learning framework containing four essential components: feature extractor, learner, example selector, and evaluator. The feature extractor generates feature vectors for pairs of entities, and the learner learns from a limited amount of initial labeled pairs of entities to develop the initial model. The example selector chooses ambiguous, unlabeled examples that the model finds hard to predict the labels for any queries an Oracle (human or ground truth) for those labels. The newly labeled data is added to the cumulative training data to learn a refined model. The evaluator evaluates the learned model during each active learning iteration.

Unsupervised learning methods. Compared with supervised learning, unsupervised learning does not need labeled training examples and automatically classifies or clusters the input data. We introduce the following unsupervised learning methods [31–35].

In the process of matching, Map-reduce-based EI [31] proposes setting different weights for each attribute to effectively distinguish the effect of each attribute on the degree of entity description. After setting the weights, the similarity between records is calculated from the attribute values and weights, and then entity matching is carried out based on the graph clustering method. Different from Map-reduce-based EI [31], ULEC [32] proposes an unsupervised method to generate groups of value pairs that can be transformed in the same way, i.e., values that are logically the same with different formats share a transformation. Then, the groups are presented to a human for verification and the approved ones are used to standardize the data.

In ZeroER [33], similarity measures are used to create the feature matrix for the two input tables, and a blocking function is selected by the users. ZeroER clusters entity pairs based on the assumption that the similarity values for matched/unmatched entity pairs follow two different distributions. The authors propose a powerful generative model based on Gaussian mixture models for learning the match and unmatched distributions. All parameters can be learned by maximizing the data likelihood via an expectation-maximization algorithm without any labeled data.

Incorporating annotated information into matching is a good idea because humans can naturally recognize information about real-world entities represented in different ways. Considering the fact that workers' answers may not be accurate due to a lack of domain expertise, fatigue, malicious behavior, etc., RER [34] corrects the responses of an oracle through indirect "control queries" and finally obtains the correct annotated information. PERC [35] adopts an uncertain graph model to address the entity matching problem with noisy crowd answers. This model sets the problem of ER equivalent to finding the maximum-likelihood clustering.

Semi-supervised learning methods. Semi-supervised learning involves a small number of labeled samples and a large number of unlabeled samples, which is not commonly used in ER. Semi-supervised de-dup [30] regards matching as a clustering task, where the goal is to put records corresponding to the same physical entity in the same cluster while separating the records corresponding to different entities into different clusters. Based on restricted correlation clustering (RCC) [42], the authors develop a semi-supervised approach that leverages a small labeled dataset, which is carefully selected via an efficient sampling procedure based on locality-sensitive hashing (LSH).

Deep Learning. As seen from the above approaches, structured data are usually featured on entity attributes and compared for similarity to determine whether they match. However, with the increasing need for matching textual data, for example, matching the information on a company homepage with a Wikipedia page describing the company, traditional learning-based ER solutions (e.g., SVMs) may have difficulty matching text instances by featuring the attribute values of textual descriptions. Considering the fact that DL-based models are sensitive to each word in a textual description, they can automatically extract the semantic features of the entire record or the corresponding attributes for matching.

DeepER [36] proposes an LSH-based blocking approach that considers all attributes of a tuple and produces smaller blocks, compared with traditional methods that consider only a few attributes. For matching, it adopts an LSTM-based RNN with the Siamese

architecture [43], a neural network architecture that has a pair of neural networks with the same architecture and shared parameters. DeepER first tokenizes each pair of entities, which are converted into distributed representations using a word embedding model. The model then aggregates token-level distributed representations into an entity representation for each entity. Next, the two entity representations are fed into a dense layer that calculates the similarity between the entities, followed by an output layer that predicts the matching.

DeepMatcher [37] extends DeepER by introducing an architecture template for deep learning ER methods with three main modules: (1) attribute embedding, which converts sequences of words used in the attribute values of an entity description to word embedding vectors; (2) attribute similarity representation, which applies a similarity function on the attribute embeddings of two descriptions to obtain a final similarity value of those descriptions (i.e., it learns the similarity function); and (3) a classifier, which uses the similarities between descriptions as features for a classifier that decides whether a pair of descriptions is a match (i.e., it learns the match function).

After DeepMatcher was proposed, the authors proposed DeepBlocking [38] to apply deep learning to blocking. Specifically, each tuple in two input tables is converted into a string by concatenating all the attribute values. The resulting strings are fed into word embedding, tuple embedding, and vector pairing. Word embedding converts the words in each string into a high-dimensional vector. Tuple embedding then combines these vectors into a single-valued vector representing the entire string (the original tuple). Finally, the vector-pairing module looks for similar vector pairs.

With the development of pre-trained LMs, DITTO [39] proposes an ER method, which serializes two records in a record pair and inputs them into a BERT (bidirectional encoder representation from transformers) model [44], while features are extracted by the BERT model. In this way, records with different structures can be uniformly processed as input to a classification model that determines whether to match. Based on modeling matching as a sentence pair classification problem and introducing pre-trained language models for fine-tuning, DITTO also proposes three optimization techniques: injecting domain knowledge, summarizing long texts so that their length meets the input requirements of the pre-trained models, and data augmentation of the training data. Machop [40] extends the work of Ditto by allowing for matching between data entries in different data formats (structured, semi-structured, and unstructured). JointMatcher [41] extends the work of Ditto by adding a relevance-aware segments encoder and a numerically aware segments encoder. Consequently, JointMatcher focuses more on similar and number-contained segments to capture more high-contextualized and fine-grained features, resulting in better performance than DITTO on small datasets with the same pre-trained LMs.

3. Truth Discovery

As different types of errors can exist in various data sources, the attribute values of an entity from multiple sources often contradict each other. To ensure data quality, we need to resolve such conflicts. An intuitive idea for eliminating conflicts is majority voting, which assumes that all sources are equally reliable. However, this assumption may not hold in most cases. To compensate for the lack of majority voting, the concept of truth discovery [45] has been introduced. Truth discovery, also known as data integration, aims to find the truths from multi-source conflicting data by simultaneously estimating the reliability of the data sources and inferring the true information.

Truth discovery was first proposed in the literature [46], and the basic idea is that the more reliable the data source, the more likely it is to provide correct data. The more correct data are provided, the more reliable the data source is. Since the reliability of a data source is usually not known in advance, truth discovery methods [46–49] mostly use iterative algorithms to perform reliability assessment of data sources and inference of truth in an alternating update mode. Usually, the algorithm starts with an initialization of source weights and then iteratively conducts the truth computation step and source weight estimation step. Some stopping criteria are adopted in practice to control the

number of iterations. One commonly adopted criterion is verifying the change of identified truths or source weights and terminating the algorithm when the change is smaller than a pre-defined threshold.

We compare recent truth discovery approaches in Table 3. Here, we summarize them under different features, i.e., *labeled truths*, *entity dependency*, *source dependency*, *non-single truth*, and *truth discovery in crowdsourcing*.

Table 3. Truth discovery methods.

Method	Labeled Truths	Entity Dependency	Source Dependency	Non-Single Truth	Crowdsourcing
SSTF [50]	✓				
SLiMFast [51]	✓				
OpSTD [52]	✓				
Investment [53]		✓		✓	
GFFTD [54]		✓			
MTM [55]		✓			
DynaTD [56]		✓			
TD-corr [57]		✓			
OTD [58]		✓			
CRDI [59]		✓			
CTD [60]		✓			
ACCU [61]			✓		
MSS [62]			✓		✓
TDCD [63]			✓		
PrecRecCorr [64]			✓	✓	
LTM [65]				✓	
DART [66]				✓	
TEM(15) [67]				✓	
FairTD [68]					✓
TDSSA [69]					✓
MTD-CC [70]			✓	✓	✓
IMCC [71]			✓		✓

3.1. Labeled Truths

Most truth discovery methods are unsupervised due to the difficulty of collecting data with truth labels, while in [50–52,72], the authors argue that a small set of truths is available and thus the proposed algorithms which work in semi-supervised settings.

SSTF [50] focuses on the problem of truth discovery with semi-supervised graph learning by using a small set of ground truth data to help distinguish true facts from false ones and identify trustworthy data sources. SSTF requires that the ground truths be among the observations sources provided. This setting is impractical for many real-world applications, especially when ground truth and observations are real numbers. Thus, it performs poorly on datasets when object truths are continuous data.

To compensate for the above shortcomings, SLiMFast [51] uses a small amount of ground truth to obtain an initial estimate of the source accuracy and then uses an iterative process to obtain a final estimate of the source accuracy and the potential true value of the object. The work in [52] studies the semi-supervised truth discovery problem for continu-

ous object truths. They propose an optimization-based semi-supervised truth discovery (OpSTD) method for discovering continuous object truths, in which the truth discovery problem is formulated as an optimization task where both object truths and source reliabilities are modeled as variables, and the ground truth is modeled as a regularization term to propagate its trustworthiness to the estimated truths.

3.2. Entity Dependency

In the general truth discovery framework, it is assumed that entities are independent and uncorrelated from each other. However, in real-world applications, different entities and attributes usually have certain relationships with each other and may influence each other. For example, there is a strong connection between the “city” and “province” where the same person lives, and in the same region, “A pays more than B” may mean “A pays more taxes than B”. This dependency relationship between entities and attributes provides more clues, which can improve the accuracy of truth discovery.

In Investment [53] and GFFTD [54], the authors represent the relationships between entities as a priori knowledge or common sense. This external knowledge is translated into propositional constraints integrated into each round of truth discovery. Specifically, each truth (an entity and its corresponding attribute values) is represented as a $[0, 1]$ variable, and then the associated truths are combined into propositional constraints based on external knowledge. The cost function is the distance between initial results based on truth discovery and new results satisfying the propositional constraints. By minimizing the cost function, the probability of each candidate truth being true is “corrected” during each iteration based on external knowledge. To ensure that this optimization problem is solvable, the truth of each entity is allowed to be “unknown” in order to lower the constraints and avoid possible constraint conflicts.

In comparison, refs. [53–55] combines information extraction and truth discovery to address the slot-filling validation task, to determine the trustworthiness of the output information extracted from different systems in the diverse corpus. The authors construct a multidimensional truth discovery model (MTM), a heterogeneous method comprising the system, the corpus, the extracted information, and the weight matrix between them. Similar to the iterative truth discovery process, the reliability of the information propagated is used to infer the reliability of the system and the corpus. In turn, the system’s reliability and the corpus’s reliability reassess the trustworthiness of the information it extracts. The authors construct knowledge graphs to establish dependencies between different time slots (entities) as a clue to initialize the trustworthiness of the information.

In another line of research, the works in [56,57] propose to exploit the temporal and spatial relationships between entities to improve the performance of the truth discovery algorithm. For example, today’s maximum temperature is correlated with yesterday’s; neighboring several streets may have similar traffic conditions. This correlation is measured as similarity is added to the optimization model, leading to better results. DynaTD [56] constructs a model of temporal relationships between entities on continuous data. TD-corr [57] can address both temporal and spatial relationships on continuous data, and their experiments show that capturing the relationships between entities can improve the performance of truth discovery since, in many practical applications, entity relationships can be viewed in large numbers by sensor-observation results. Building on [56,57], OTD [58] further models and integrates temporal trends in the evolution of truth guides the truth discovery.

Unlike all the above methods, ref. [59] proposes an approach, CRDI, to enhance the accuracy of truth discovery. CRDI uses the information in the relationships between entities to find more evidence for the correctness or incorrectness of the values generated by different data sources. The authors then propose an alternative approach [73], which uses relational machine learning methods to estimate the relations between entities and then uses these relations to estimate the true value using some fusion functions.

CTD [60] introduces denial constraints into the truth discovery problem and formulates it as a constrained optimization problem. To address the problem, they propose algorithms to partition the entities into disjoint groups and generate arithmetic constraints for each disjoint group separately. The true attribute values of the entities in each disjoint group are then derived by minimizing the objective function under the corresponding arithmetic constraints.

3.3. Source Dependency

Since copying sometimes occurs between multiple data sources (e.g., different websites on the Internet), detecting copying between data sources is essential to prevent erroneous information from being copied continuously, leading to a high proportion of error messages when discovering the truth. The main principle behind copy detection is that, if some sources make many common mistakes, they are not likely to be independent of each other. However, this principle becomes ineffective when some sources copy information from a good source.

In ACCU [61], the authors use a Bayesian model to infer the existence of copy relationships between data sources and confirm their dependence. This copy-detection process is combined with the truth discovery for iterative updates. MSS [62] mitigates the source relationship dependency problem by revealing the underlying group structure among data sources and performing truth discovery at the group level. This solution reduces the risk of overusing information from dependent data sources, especially when these data sources are unreliable.

In contrast to [61,62], which detect copy relationships between data sources from static data, TDCD [63] detects copy relationships between data sources from dynamic data sources, where the information provided by the data sources is changing. Given the historical update records of the data sources, the authors apply a hidden Markov model (HMM) to detect the copy relationships. The model iteratively evaluates source quality and updated information simultaneously, outputting changing copy relationships and truth.

In a separate line of research, PrecRecCorr [64] considers different correlations between data sources that are more general than copying. Sources may have positive correlations, such as copying information from each other, or negative correlations, such as providing data from complementary domains or focusing on different information domains. In general, observations from positively correlated sources should not increase the confidence that they are true values, and observations supported by only a few negatively correlated sources should not decrease the confidence that they are true values. The method models inter-source correlation with joint precision and joint recall and employs Bayesian analysis to infer whether an observation is true.

3.4. Non-Single Truth

Most methods usually assume that each entity has only one correct value for each entity. Based on this assumption, these methods aim to select the information with the highest confidence as the correct value. However, this “single truth” assumption does not always hold. For example, there are multiple authors for a book and also multiple actors/actresses for a movie.

In the literature [64,65], the authors propose the probabilistic graphical model to find multiple truths for each entity. In this case, considering only the credibility of the source does not distinguish between sources with low precision and those with low recall. Therefore, in [64], the authors calculate the precision and recall of the data source for finding multiple truths. Similarly, LTM [65] considers both false positive and negative aspects to find multiple truths for the same entity. DART [66] proposes an integrated Bayesian approach that exploits the unique features of the multi-truth problem to assign more reasonable confidence scores to candidate truth sets, combining data source domain knowledge and confidence scores to achieve unsupervised multiple truth discovery.

In addition to the above case of multiple true values, there is a particular case of no correct value [53,67]. For example, it is suitable to give an output of “unknown” for the questions (objects) about the death date of someone still alive. Investment [53] uses “unknown” to augment their data and provides an “unknown” answer when there is insufficient information or conflicting constraint. TEM [67] considers the has-truth questions, i.e., when the truth of an entity is not provided by any data source, the truth returned by truth discovery for that entity should be non-existent. The study evaluated three quality indicators of the data source in the search for truth, silent rate, false spoken rate, and true spoken rate, thus effectively filtering out information about entities for which there is no truth.

3.5. Truth Discovery in Crowdsourcing

In the past decade, crowdsourcing has emerged as a popular internet-based collaborative computing paradigm. In crowdsourcing systems, requesters can ask workers (“sources”) for true values (“truths”) of objects or events. Generally, sources could provide inconsistent or conflicting answers (“candidates”) about the object. Researchers have studied truth discovery in crowdsourcing from different perspectives.

Crowdsourcing collects shared information from a large number of people and is often negatively impacted by unreliable sources with low-quality data. To address this problem, MSS [62] proposes a probabilistic model to jointly assess the reliability of sources and find true data. It also explicitly reveals the reliability of groups and minimizes the negative impacts of unreliable groups.

FairTD [68] argues from a fairness perspective that simply aggregating feedback from all crowdsourced workers will inevitably lead to race, gender, and political interference in the outcome. Thus, FairTD proposes a notion of difference in truth discovery, three theories of fairness enhancement, and an iterative method of estimating bias and truth to address the fairness problem in truth discovery. TDSSA [69] considers the truth discovery problem under Sybil attack and proposes a method to defend against strategic Sybil attack. The method assigns the same standard set of tasks to each worker in the cluster. It filters malicious worker feedback out by classifying each cluster as normal or malicious based on worker feedback.

FairTD and TDSSA ignore the correlation between candidates, so the inferred truth may differ from the ground truth. To solve this problem, ref. [70] proposes MTD-CC, multi-truth discovery with candidate correlations. Specifically, the authors first design a metric of potential function to measure the correlation between each pair of candidates based on the sources’ votes and reliabilities. Then, they construct a Markov random field (MRF) to represent these correlations. Finally, they transform the MRF into a directed graph and cut it based on the Min-cut theorem to infer which candidates are truths.

Unlike the copy detection in Section 3.3, IMCC [71] considers the copying of workers in crowdsourcing. Some workers copy some (or all) of the data from other workers, making truth discovery more complicated if the wrong answer is copied. IMCC aims to design a crowdsourcing incentive mechanism for the truth discovery of textual answers with copiers. They formulate the problem of maximizing social welfare such that all tasks can be completed with the least confidence for truth discovery and design a three-stage incentive mechanism. In the contextual embedding and clustering stage, they construct and cluster the content vector representations of crowdsourced textual answers at the semantic level. In the truth discovery stage, they estimate the truth for each task based on the dependence and accuracy of workers. In the reverse auction stage, they design a greedy algorithm to select the winners and determine the payment.

4. Inconsistency Detection and Repair

Data consistency refers to data collection that does not contain semantically incorrect or conflicting data. For data consistency detection, existing studies [74–76] usually establish a set of data quality rules such that once inconsistent data appear in the database, the corresponding rules are violated so that the tuple violating the rules is discovered and

repaired. Data quality rules describing consistency include function dependencies (FDs), conditional functional dependencies (CFDs), denial constraints (DCs), etc. The followings are brief descriptions of these three common data rules, respectively.

Consider a relational schema R with attributes $attr(R)$. An FD φ is defined as $X \rightarrow Y$, where $X \subseteq attr(R)$ and $Y \subseteq attr(R)$. An instance I of R satisfies FD φ , denoted as $I \models \varphi$ if for any two tuples t_α, t_β in I , such that $t_\alpha[X] = t_\beta[X]$, then $t_\alpha[Y] = t_\beta[Y]$. For example, an FD $[Country\ Code, Area\ Code] \rightarrow City$ is presented in Table 4, stating that the values of *Country Code* and the *Area Code* can determine the value of *City*.

A CFD φ on R is a pair $(R : X \rightarrow Y, T_p)$, where (1) X, Y are sets of attributes from $attr(R)$, (2) $R : X \rightarrow Y$ is a standard FD, referred to as the FD *embedded in* φ ; and (3) T_p is a tableau with all attributes in X and Y , referred to as the *pattern tableau* of φ , where for each A in X or Y and each tuple $t \in T_p$, $t[A]$ is either a constant ‘a’ in the domain $dom(A)$ of A , or an unnamed variable ‘_’. If A appears in both X and Y , we use $t[A_L]$ and $t[A_R]$ to indicate the A field of t corresponding to A in X and Y , respectively [77]. For instance, given a CFD: $([Country\ Code, Zip] \rightarrow Street, (44, _ , _))$, $[Country\ Code, Zip] \rightarrow Street$ refers to an FD and $(44, _ , _)$ specifies the condition, i.e., when *Country Code* is 44, *Zip* uniquely determines *Street*.

A DC φ on R is defined as: $\forall t_\alpha, t_\beta, t_\gamma, \dots \in R, \neg(P_1 \wedge P_2 \dots \wedge P_m)$, where each predicate P_i is of the form $v_1 \theta v_2$ or $v_1 \theta c$ with $v_1, v_2 \in t_x[A]$, $x \in \{\alpha, \beta, \gamma, \dots\}$, $A \in R$, c is a constant in the domain of A , and $\theta \in \{=, <, >, \neq, \leq, \geq\}$. For example, $\forall t_\alpha \in R, \neg(t_\alpha[Country\ Code] = '01' \wedge t_\alpha[Area\ Code] = '08' \wedge t_\alpha[City] \neq 'MH')$, i.e., there is no tuple where *Country Code* is ‘01’, *Area Code* is ‘08’, and *City* is not ‘MH’.

Table 4. An example of database information.

Tid	Country Code	Area Code	Phone	Name	Street	City	Zip
t_1	01	908	5219527	Mike	Tree Ave.	MH	07974
t_2	01	908	5219527	Rick	Tree Ave.	MH	07974
t_3	01	212	2018967	Joe	5th Ave	NYC	01202
t_4	01	908	2018967	Jim	Elm Str.	MH	07974
t_5	44	131	3314823	Ben	High St.	EDI	EH4 1DI
t_6	44	131	4675378	Ian	High St.	EDI	EH4 1DT
t_7	44	908	4675378	Ian	Port PI	MH	W1B 1JH
t_8	01	131	2018967	Sean	3rd Str.	UN	01202

The problem of repairing inconsistent data is making “minimal” changes to data collection that do not satisfy a given set of rules. In Table 5, we summarize the inconsistency detection and repair methods in two dimensions, *Data* and *Constraints*. *Data* divides the methods into *Table Data* and *Graph Data*, which represent the type of data handled by the method. *Constraints* refers to *FDs*, *CFDs*, *DCs*, and *Other Constraints*, which represent the data quality rules used in the method. Next, we discuss these papers according to *Data*.

Table 5. Inconsistency detection and repair methods.

	Data		Constraints			
	Table Data	Graph Data	FDs	CFDs	DCs	Other Constraints
FD-based value modification [78]	✓		✓			✓
FindVRepair [79]	✓		✓	✓		

Table 5. Cont.

	Data		Constraints			
	Table Data	Graph Data	FDs	CFDs	DCs	Other Constraints
CFD-based Detection [77]	✓			✓		
CFD-based value modification [80]	✓			✓		
Distributed Detection [81]	✓			✓		
Incremental Detection [82]	✓			✓		
C-Repair [83]	✓					✓
Holistic Repair [84]	✓				✓	
HoloClean [85]	✓				✓	✓
GFD-based Detection [86]		✓				✓
IncDect [87]		✓				✓
StarRepair [88]		✓				✓
Logic-based Graph Repair [89]		✓				✓

4.1. Table Data

Traditional repairing methods [2,90,91] directly delete the non-compliant records from the database so that the remaining records can comply with all the given universal integrity constraints (ICs). Since the errors in the remaining dataset are not increased when the records are deleted, it is always possible to iterate through the data repair, i.e., first finding the set of records that violate the constraint, then deleting a random record, and then testing the remaining dataset. This strategy is relatively simple, but a significant amount of information is lost. To address this issue, an alternative strategy [92] is proposed. Instead of adding or deleting any records, only certain fields are modified so that the entire collection can meet the pre-defined consistency requirements. This strategy allows for maximum preservation of the original information of the data collection.

In [78], the authors propose a repair model which detects tuples that violate FDs and inclusion dependencies (INDs) and modifies attribute values to ensure consistency in the database. The model describes the cost-based minimum repair measured by the tuple weights and the similarity of the values for each modification, which was not considered by any previous work. The tuple weights represent the confidence of each tuple placed by the users. The similarity of the values is defined as the minimum number of individual character insertions, deletions, and substitutions required to convert the original value to the repaired value.

In [77], the authors argue that FDs have been developed primarily for schema design and are often insufficient to capture the semantics of the data. They introduce CFDs, an extension of traditional FDs, which can capture data consistency by enforcing bindings of semantically related values. They provide an inference system analogous to Armstrong's axioms for FDs and consistency analysis. Since CFDs allow for data bindings, many individual constraints may hold on a table, complicating the detection of constraint violations. They first provide an SQL technique for finding violations of a single CFD, e.g., using group by statements and then extend it to validate multiple CFDs.

Following [78], FindVRepair [79] proposes a method for repairing the inconsistency errors in the database that violate FDs by minimizing the distance metric. The distance metric depends on the number of value modifications and the weight/confidence associated with the modified tuples. In addition, the authors show how to use CFDs for data repair. In [80], the authors propose two algorithms: one for automatically computing the repaired database that satisfies a given CFD set and the other for incrementally finding repairs in response to updates to a clean database. The former extends the FD-based cost model of [78] to ensure quality repairs are found when processing CFDs. The latter is an effective heuristic algorithm for finding fixes that respond to updates, i.e., the deletion or insertion of a set of tuples, and can also be used to find fixes for dirty databases.

In another line of research, ref. [81] describes the detection problem in various distributed environments. The authors formulate CFD violation detection for data partitioned horizontally or vertically as optimization problems measured by either response time or data shipment (i.e., the amount of data shipped from one site to another). They study CFD violation detection in horizontal partitioning but consider neither incremental detection nor algorithms for detecting errors in vertical partitions. The works in [81,82] provide such incremental algorithms for vertically partitioned and horizontally partitioned data and further propose optimization techniques for the incremental algorithm over vertical partitions.

Compared with the above CFD-based repairing methods, ref. [83] no longer focuses solely on databases containing CFDs but specifically on data containing conditional functional dependencies (CFDPs) with built-in predicates. The authors propose a heuristic algorithm for inconsistent data detection and repairing, which can repair inconsistent data violating the CFDPs in datasets under unsupervised circumstances. For data detection, the maximum dependency set is used to detect them. For data repairing, they use unsupervised machine learning to learn the correlation among attributes in datasets and integrate the minimum cost idea and information theory to repair, which makes the repair results most relevant to the initial values with minimum repair times.

The above techniques do not consider the interaction among different class constraint violations. Therefore, the researchers introduce the DCs for repairing [84,85], which subsumes several types of DCs, such as FDs and CFDs. The authors in [84] let users specify quality rules using DCs with ad hoc predicates. They introduce two data structures: the Conflict Hypergraph, which encodes all violations into a common graph structure, and the Repair Context, which encodes all necessary information on how to fix violations holistically.

The authors also introduce HoloClean [85], a framework for holistic data repairing driven by probabilistic inference. HoloClean unifies existing qualitative data repairing approaches, which rely on DCs or external data sources, with quantitative data repairing methods, which leverage statistical properties of the input data. With an inconsistent dataset as input, HoloClean automatically generates a probabilistic program to fix data.

Data quality rules by themselves are not sufficient to correctly resolve conflicts, or worse, may even introduce new errors when attempting to repair data. Thus, some studies [76,93] use master data or users to guide the repair process. In [76], the authors propose a method to find certain fixes based on master data, the concept of certain regions, and a class of editing rules. A certain region is a set of attributes for which the user is guaranteed to be correct. Given a certain region and master data, edit rules tell which attributes to fix and how to update them. Techniques for reasoning about edit rules are then developed to determine whether they lead to unique fixes and whether they can fix all attributes in the tuple relative to the master data and a certain region. A framework and algorithms are also proposed to find certain fixes by interacting with the user to ensure that a region is correct.

In [93], the authors introduce GDR, a guided data repair framework that incorporates user feedback in the repairing process to enhance and accelerate existing automated repair techniques while minimizing user involvement. Specifically, GDR consults the user on the updates that are most likely to be beneficial in improving data quality. Then, GDR uses

existing machine learning methods to identify and apply the correct updates directly to the database without the actual involvement of the user on these specific updates.

4.2. Graph Data

Graph data inconsistency refers to semantic information errors or contradictions among the relevant data information in a graph. Recent methods capture inconsistency errors in graph data and repair them in terms of rules [86,87], logic-based incremental methods [89], etc.

In [86], the authors propose a class of FDs for graphs, referred to as GFDs. Capturing the attribute-value dependencies and topology of entities through GFDs provides an effective way to detect inconsistencies in knowledge graphs. The same team later proposed an arithmetic and comparison expression, NGD, to capture semantic inconsistencies in the graphs, and developed an incremental algorithm, IncDect [87], to detect errors in graph data and provide provable performance guarantees. Compared to GFDs, NGDs focus more on numerical inconsistency in real-life situations, and NGDs extend GFDs by supporting linear arithmetic expressions and built-in comparison predicates $\{=, \neq, >, \geq, <, \leq\}$.

In [88], the authors propose a class of constraints called Star Function Dependencies (StarFDs), which enforce value dependencies conditional on entities and their associated neighbors, unlike traditional ICs. These neighbors are identified by a star schema containing a join rule path query. This type of star function dependency strikes a balance between expressiveness and complexity. Given a set of StarFDs and graphs, entity repair detects and resolves inconsistencies differently using optimal and cost-bounded solutions, referred to as the StarRepair method. StarRepair computes the repairing process under the minimum cost by enforcing StarFDs, where the cost is determined by the size of the inconsistency.

In another line of research, ref. [89] proposes a logic-based approach to incremental graph repair, a method that does minimal change repair. The authors formalize consistency using a so-called graph condition equivalent to a first-order logic on the graph. Inconsistency is then resolved according to two repair algorithms: state-based repair independent of graph update history, and incremental repair that considers graph update history to restore consistency.

5. Detection and Repair of Compound-Type Errors

In practice, multi-source data often contain multiple error types, and fixing one type of error may introduce other types of errors. Thus, several methods focus on fixing compound errors. We summarize these methods in Table 6, where the considered types of errors are *Duplication* (i.e., entity information overlapping), *Conflict* (i.e., attribute value conflicts), *Inconsistency* (i.e., attribute value inconsistency), and *Incomplete*. As several papers consider the incomplete case in their compound-type error study, we also added “*Incomplete*” as an aspect of classification.

In [94], the authors design graph repair rules (GRRs) that capture incompleteness, conflict, and redundancy in a graph and indicate how to repair these errors. Three fundamental issues defined semantically in GRRs, namely implication, consistency, and termination, are investigated to verify whether a GRR is meaningful. The work in [94,95] proposes to deduce certain fixes for graphs based on data quality rules and truth values. They define data quality rule (GQRs), to support CFDs, recursively defined keys, and negative rules on graphs so that repairs can be deduced by combining data repair and entity resolution. In addition, they prove that deducing certain fixes is Church–Rosser and propose online and offline modes to perform fixes, respectively.

The authors of [95] also propose another class of graph association rules [96], represented by GARs, to describe regular relationships between entities in a graph. GARs are combinations of graph schemas and dependencies that can be used as predicates for relationship prediction by machine learning classifiers, capable of capturing incomplete information in schema-free graphs, predicting links in social graphs, identifying leads in digital marketing, and extending graph function dependencies to capture missing links and inconsistencies. They demonstrate that the satisfiability, implication, and association

derivation problems for GARs are coNP-complete, NP-complete, and NP-complete, respectively, as well as that they maintain the same complexity bounds as the graph function dependency problem despite the improved expressiveness of GARs.

Instead of relying on various data quality rules, ref. [97] proposes a knowledge fragment cleaning method (KFCM) for cleaning up faulty knowledge fragments in a genealogical knowledge graph, consisting of three phases. The first stage detects and analyzes faulty knowledge fragments by inferring common problem patterns; the second stage supplements the entity relationship for different error patterns and conducts entity resolution; the third stage aligns entities with missing attributes based on parent–child relationships and connects isolated knowledge fragments.

In [98], the authors propose combining the idea of truth discovery methods and rule-based data repair methods to resolve the conflicts and inconsistencies. They present an automatic multi-source data repair method, AutoRepair, that uses FDS to resolve inconsistencies in multi-source data. The source reliability is used as evidence to discover and repair the errors among these violations. At the same time, the corrected results are used to estimate the source reliability. As the source reliability is unknown, the process is modeled as an iterative framework to ensure better performance.

The authors of [98] then propose incorporating DCs into the truth discovery process to resolve the conflicts together with dependencies [99]. They formulate the data repairing problem as an optimization problem and create constraints for the DCs. Compared with [98], this approach can repair the errors among related entities in different data types (e.g., categorical data and continuous data). They also give four concrete cases using different classes of DCs as examples.

Table 6. Compound-type error detection and repair methods.

Method	Duplication	Conflict	Inconsistency	Incomplete
GRRs [94]	✓		✓	✓
GQRs [95]	✓		✓	
GARs [96]			✓	✓
KFCM [97]	✓		✓	✓
AutoRepair [98]		✓	✓	
Conflicts Together with Dependencies [99]		✓	✓	

6. Future Research Directions

In summary, many studies have been carried out by domestic and international scholars in multi-source data error detection and repair research. These research efforts have focused on entity resolution and truth discovery, and some methods that have emerged in recent years have considered the inconsistency of multi-source data. However, several unresolved issues remain.

The detection and repair of compound-type errors in multi-source data need to be further studied. As compound-type errors commonly exist in multi-source data, error detection, and repair methods for multi-source data need to consider multiple types of errors simultaneously to obtain optimal repair solutions. However, the repair of existing compound types of errors is limited to considering only a few types of errors. Thus, there needs to be more research work that integrates the consideration of conflicting, duplicated, and inconsistent entity information on multi-source data.

The error detection based on semantic association in multi-source data needs to be studied in depth. Existing inconsistency detection methods typically use different types of data quality rules [100] to express dependencies between attributes or attribute values and treat information that violates the corresponding type of data quality rule as an error and

repair it. However, with multiple data sources, information from different sources may be expressed differently. Let us suppose only exact matches of data values are considered. In that case, this can affect the judgment of the reliability of the data source and the accuracy of error detection and subsequent repair. Therefore, it is necessary to design data quality rules based on semantic association to detect various errors accurately.

Efficient repair in multi-source data needs to be further studied. Existing entity matching methods and inconsistency detection and repair methods tend to have high time complexity and can suffer from more severe efficiency problems in massive data scenarios [101]. In the present era of Big Data, modern data mining and analysis methods often include multiple iterations, thus making the efficiency disadvantage of traditional methods even more apparent. Therefore, error repair methods for multi-source data need to maximize efficiency while ensuring accuracy.

Author Contributions: Resources, H.W.; writing—original draft preparation, C.Y., H.D. and H.Z. (Hengtong Zhang); writing—review and editing, H.Z. (Hua Zhang); supervision, H.W. and G.D.; funding acquisition, G.D. All authors have read and agreed to the published version of the manuscript.

Funding: This paper was partially supported by National Natural Science Foundation of China (No. 62202132), Natural Science Foundation of Zhejiang Province (No. LQ22F020032), and National Key Research and Development Program of China (No. 2022YFE0199300).

Data Availability Statement: Not applicable.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Fan, W. Data Quality: From Theory to Practice. *SIGMOD Rec.* **2015**, *44*, 7–18. [[CrossRef](#)]
2. Arenas, M.; Bertossi, L.E.; Chomicki, J. Consistent Query Answers in Inconsistent Databases. In Proceedings of the Eighteenth ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems, Philadelphia, PA, USA, 31 May–2 June 1999; pp. 68–79.
3. Cao, Y.; Deng, T.; Fan, W.; Geerts, F. On the data complexity of relative information completeness. *Inf. Syst.* **2014**, *45*, 18–34. [[CrossRef](#)]
4. Fan, W.; Geerts, F. Relative Information Completeness. *ACM Trans. Database Syst.* **2010**, *35*, 97–106. [[CrossRef](#)]
5. Fellegi, I.P.; Sunter, A.B. A theory for record linkage. *J. Am. Stat. Assoc.* **1969**, *64*, 1183–1210. [[CrossRef](#)]
6. Elmagarmid, A.K.; Ipeirotis, P.G.; Verykios, V.S. Duplicate Record Detection: A Survey. *IEEE Trans. Knowl. Data Eng.* **2007**, *19*, 1–16. [[CrossRef](#)]
7. Zhang, H.; Diao, Y.; Immerman, N. Recognizing Patterns in Streams with Imprecise Timestamps. *Inf. Syst.* **2013**, *38*, 1187–1211. [[CrossRef](#)]
8. Clifford, J.; Dyreson, C.; Isakowitz, T.; Jensen, C.S.; Snodgrass, R.T. On the Semantics of “Now” in Databases. *ACM Trans. Database Syst.* **1997**, *22*, 171–214. [[CrossRef](#)]
9. Widom, J. Trio: A System for Integrated Management of Data, Accuracy, and Lineage. In Proceedings of the Second Biennial Conference on Innovative Data Systems Research, Asilomar, CA, USA, 4–7 January 2005; Volume 5, pp. 262–276.
10. Rahm, E.; Do, H.H. Data Cleaning: Problems and Current Approaches. *IEEE Data Eng. Bull.* **2000**, *23*, 3–13.
11. Papadakis, G.; Ioannou, E.; Palpanas, T. Entity Resolution: Past, Present and Yet-to-Come. In Proceedings of the 23rd International Conference on Extending Database Technology, Copenhagen, Denmark, 30 March–2 April 2020; pp. 647–650.
12. Konda, P.; Das, S.; Doan, A.; Ardalan, A.; Ballard, J.R.; Li, H.; Panahi, F.; Zhang, H.; Naughton, J.; Prasad, S.; et al. Magellan: Toward Building Entity Matching Management Systems. *Proc. VLDB Endow.* **2016**, *9*, 1197–1208. [[CrossRef](#)]
13. Draisbach, U.; Naumann, F. A generalization of blocking and windowing algorithms for duplicate detection. In Proceedings of the 2011 International Conference on Data and Knowledge Engineering, Milano, Italy, 6 September 2011; pp. 18–24.
14. Wang, J.; Kraska, T.; Franklin, M.J.; Feng, J. CrowdER: Crowdsourcing Entity Resolution. *Proc. VLDB Endow.* **2012**, *5*, 1483–1494. [[CrossRef](#)]
15. Benjelloun, O.; Garcia-Molina, H.; Menestrina, D.; Su, Q.; Whang, S.E.; Widom, J. Swoosh: A generic approach to entity resolution. *VLDB J.* **2009**, *18*, 255–276. [[CrossRef](#)]
16. Singh, R.; Meduri, V.V.; Elmagarmid, A.; Madden, S.; Papotti, P.; Quiané-Ruiz, J.A.; Solar-Lezama, A.; Tang, N. Synthesizing Entity Matching Rules by Examples. *Proc. VLDB Endow.* **2017**, *11*, 189–202. [[CrossRef](#)]
17. Wang, H.; Ding, X.; Li, J.; Gao, H. Rule-Based Entity Resolution on Database with Hidden Temporal Information. *IEEE Trans. Knowl. Data Eng.* **2018**, *30*, 2199–2212. [[CrossRef](#)]
18. Hao, S.; Tang, N.; Li, G.; Feng, J. Discovering Mis-Categorized Entities. In Proceedings of the 34th IEEE International Conference on Data Engineering, Paris, France, 16–19 April 2018; pp. 413–424.

19. Papadakis, G.; Ioannou, E.; Niederée, C.; Fankhauser, P. Efficient entity resolution for large heterogeneous information spaces. In Proceedings of the Forth International Conference on Web Search and Web Data Mining, Hong Kong, China, 9–12 February 2011; pp. 535–544.
20. Papadakis, G.; Ioannou, E.; Palpanas, T.; Niederée, C.; Nejdil, W. A blocking framework for entity resolution in highly heterogeneous information spaces. *IEEE Trans. Knowl. Data Eng.* **2012**, *25*, 2665–2682. [[CrossRef](#)]
21. Lacoste-Julien, S.; Palla, K.; Davies, A.; Kasneci, G.; Graepel, T.; Ghahramani, Z. Sigma: Simple greedy matching for aligning large knowledge bases. In Proceedings of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Chicago, IL, USA, 11–14 August 2013; pp. 572–580.
22. Efthymiou, V.; Papadakis, G.; Stefanidis, K.; Christophides, V. MinoanER: Schema-Agnostic, Non-Iterative, Massively Parallel Resolution of Web Entities. In Proceedings of the Advances in Database Technology—22nd International Conference on Extending Database Technology, Lisbon, Portugal, 26–29 March 2019; pp. 373–384.
23. Bilenko, M.; Kamath, B.; Mooney, R.J. Adaptive Blocking: Learning to Scale Up Record Linkage. In Proceedings of the 6th IEEE International Conference on Data Mining, Hong Kong, China, 18–22 December 2006; pp. 87–96.
24. Michelson, M.; Knoblock, C.A. Learning Blocking Schemes for Record Linkage. In Proceedings of the AAAI’06: Proceedings of the 21st National Conference on Artificial Intelligence, Boston, MA, USA, 16–20 July 2006; pp. 440–445.
25. Evangelista, L.O.; Cortez, E.; da Silva, A.S.; Meira, W., Jr. Adaptive and Flexible Blocking for Record Linkage Tasks. *J. Inf. Data Manag.* **2010**, *1*, 167.
26. Das Sarma, A.; Jain, A.; Machanavajjhala, A.; Bohannon, P. An Automatic Blocking Mechanism for Large-Scale de-Duplication Tasks. In Proceedings of the 21st ACM International Conference on Information and Knowledge Management, Maui, HI, USA, 29 October–2 November 2012; pp. 1055–1064.
27. Lin, X.; Li, H.; Xin, H.; Li, Z.; Chen, L. KBPearl: A Knowledge Base Population System Supported by Joint Entity and Relation Linking. *Proc. VLDB Endow.* **2020**, *13*, 1035–1049. [[CrossRef](#)]
28. Zeng, W.; Zhao, X.; Tang, J.; Lin, X. Collective Entity Alignment via Adaptive Features. In Proceedings of the 36th IEEE International Conference on Data Engineering, Dallas, TX, USA, 20–24 April 2020; pp. 1870–1873.
29. Meduri, V.V.; Popa, L.; Sen, P.; Sarwat, M. A Comprehensive Benchmark Framework for Active Learning Methods in Entity Matching. In Proceedings of the 2020 International Conference on Management of Data, Portland, OR, USA, 14–19 June 2020; pp. 1133–1147.
30. Kushagra, S.; Saxena, H.; Ilyas, I.F.; Ben-David, S. A Semi-Supervised Framework of Clustering Selection for De-Duplication. In Proceedings of the 35th IEEE International Conference on Data Engineering, Macao, China, 8–11 April 2019; pp. 208–219.
31. Ran, H.; Wang, H.; Rong, Z.; Li, J.; Hong, G. Map-Reduce Based Entity Identification in Big Data. *J. Comput. Res. Dev.* **2013**, *50*, 170–179.
32. Deng, D.; Tao, W.; Abedjan, Z.; Elmagarmid, A.K.; Ilyas, I.F.; Li, G.; Madden, S.; Ouzzani, M.; Stonebraker, M.; Tang, N. Unsupervised String Transformation Learning for Entity Consolidation. In Proceedings of the 35th IEEE International Conference on Data Engineering, Macao, China, 8–11 April 2019; pp. 196–207.
33. Wu, R.; Chaba, S.; Sawlani, S.; Chu, X.; Thirumuruganathan, S. ZeroER: Entity Resolution using Zero Labeled Examples. In Proceedings of the 2020 International Conference on Management of Data, Portland, OR, USA, 14–19 June 2020; pp. 1149–1164.
34. Galhotra, S.; Firmani, D.; Saha, B.; Srivastava, D. Robust entity resolution using random graphs. In Proceedings of the 2018 International Conference on Management of Data, Houston, TX, USA, 10–15 June 2018; pp. 3–18.
35. Ke, X.; Teo, M.; Khan, A.; Yalavarthi, V.K. A Demonstration of PERC: Probabilistic Entity Resolution with Crowd Errors. *Proc. VLDB Endow.* **2018**, *11*, 1922–1925. [[CrossRef](#)]
36. Ebraheem, M.; Thirumuruganathan, S.; Joty, S.R.; Ouzzani, M.; Tang, N. Distributed Representations of Tuples for Entity Resolution. *Proc. VLDB Endow.* **2018**, *11*, 1454–1467. [[CrossRef](#)]
37. Mudgal, S.; Li, H.; Rekatsinas, T.; Doan, A.; Park, Y.; Krishnan, G.; Deep, R.; Arcaute, E.; Raghavendra, V. Deep learning for entity matching: A design space exploration. In Proceedings of the 2018 International Conference on Management of Data, Houston, TX, USA, 10–15 June 2018; pp. 19–34.
38. Thirumuruganathan, S.; Li, H.; Tang, N.; Ouzzani, M.; Govind, Y.; Paulsen, D.; Fung, G.; Doan, A. Deep Learning for Blocking in Entity Matching: A Design Space Exploration. *Proc. VLDB Endow.* **2021**, *14*, 2459–2472. [[CrossRef](#)]
39. Li, Y.; Li, J.; Sahara, Y.; Doan, A.; Tan, W.C. Deep Entity Matching with Pre-Trained Language Models. *Proc. VLDB Endow.* **2020**, *14*, 50–60. [[CrossRef](#)]
40. Wang, J.; Li, Y.; Hirota, W.; Kandogan, E. Machop: An end-to-end generalized entity matching framework. In Proceedings of the Fifth International Workshop on Exploiting Artificial Intelligence Techniques for Data Management, Philadelphia, PA, USA, 17 June 2022; pp. 2:1–2:10.
41. Ye, C.; Jiang, S.; Zhang, H.; Wu, Y.; Shi, J.; Wang, H.; Dai, G. JointMatcher: Numerically-aware entity matching using pre-trained language models with attention concentration. *Knowl. Based Syst.* **2022**, *251*, 109033. [[CrossRef](#)]
42. Kushagra, S.; Ben-David, S.; Ilyas, I.F. Semi-supervised clustering for de-duplication. In Proceedings of the 22nd International Conference on Artificial Intelligence and Statistics, Naha, Japan, 16–18 April 2019; Volume 89, pp. 1659–1667.
43. Mueller, J.; Thyagarajan, A. Siamese Recurrent Architectures for Learning Sentence Similarity. In Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence, Phoenix, AZ, USA, 12–17 February 2016; pp. 2786–2792.

44. Devlin, J.; Chang, M.; Lee, K.; Toutanova, K. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Minneapolis, MN, USA, 2–7 June 2019; pp. 4171–4186.
45. Li, Y.; Gao, J.; Meng, C.; Li, Q.; Su, L.; Zhao, B.; Fan, W.; Han, J. A survey on truth discovery. *SIGKDD Explor.* **2016**, *17*, 1–16. [[CrossRef](#)]
46. Yin, X.; Han, J.; Yu, P.S. Truth discovery with multiple conflicting information providers on the web. In Proceedings of the 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Jose, CA, USA, 12–15 August 2007; pp. 1048–1052.
47. Xiao, H.; Gao, J.; Li, Q.; Ma, F.; Su, L.; Feng, Y.; Zhang, A. Towards confidence interval estimation in truth discovery. *IEEE Trans. Knowl. Data Eng.* **2018**, *31*, 575–588. [[CrossRef](#)]
48. Li, Q.; Li, Y.; Gao, J.; Su, L.; Zhao, B.; Demirbas, M.; Fan, W.; Han, J. A confidence-aware approach for truth discovery on long-tail data. *Proc. VLDB Endow.* **2014**, *8*, 425–436. [[CrossRef](#)]
49. Li, Q.; Li, Y.; Gao, J.; Zhao, B.; Fan, W.; Han, J. Resolving conflicts in heterogeneous data by truth discovery and source reliability estimation. In Proceedings of the International Conference on Management of Data, Snowbird, UT, USA, 22–27 June 2014; pp. 1187–1198.
50. Yin, X.; Tan, W. Semi-supervised truth discovery. In Proceedings of the 20th International Conference on World Wide Web, Hyderabad, India, 28 March–1 April 2011; pp. 217–226.
51. Rekatsinas, T.; Joglekar, M.; Garcia-Molina, H.; Parameswaran, A.G.; Ré, C. SLIMFast: Guaranteed Results for Data Fusion and Source Reliability. In Proceedings of the 2017 ACM International Conference on Management of Data, Chicago, IL, USA, 14–19 May 2017; pp. 1399–1414.
52. Yang, Y.; Bai, Q.; Liu, Q. On the Discovery of Continuous Truth: A Semi-supervised Approach with Partial Ground Truths. In Proceedings of the Web Information Systems Engineering—WISE 2018—19th International Conference, Dubai, United Arab Emirates, 12–15 November 2018; Volume 11233, pp. 424–438.
53. Pasternack, J.; Roth, D. Knowing What to Believe (when you already know something). In Proceedings of the 23rd International Conference on Computational Linguistics, Beijing, China, 23–27 August 2010; pp. 877–885.
54. Pasternack, J.; Roth, D. Making Better Informed Trust Decisions with Generalized Fact-Finding. In Proceedings of the 22nd International Joint Conference on Artificial Intelligence, Barcelona, Spain, 16–22 July 2011; pp. 2324–2329.
55. Yu, D.; Huang, H.; Cassidy, T.; Ji, H.; Wang, C.; Zhi, S.; Han, J.; Voss, C.R.; Magdon-Ismael, M. The Wisdom of Minority: Unsupervised Slot Filling Validation based on Multi-dimensional Truth-Finding. In Proceedings of the 25th International Conference on Computational Linguistics, Technical Papers, Dublin, Ireland, 23–29 August 2014; pp. 1567–1578.
56. Li, Y.; Li, Q.; Gao, J.; Su, L.; Zhao, B.; Fan, W.; Han, J. On the Discovery of Evolving Truth. In Proceedings of the 21st ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Sydney, NSW, Australia, 10–13 August 2015; pp. 675–684.
57. Meng, C.; Jiang, W.; Li, Y.; Gao, J.; Su, L.; Ding, H.; Cheng, Y. Truth Discovery on Crowd Sensing of Correlated Entities. In Proceedings of the 13th ACM Conference on Embedded Networked Sensor Systems, Seoul, Republic of Korea, 1–4 November 2015; pp. 169–182.
58. Yao, L.; Su, L.; Li, Q.; Li, Y.; Ma, F.; Gao, J.; Zhang, A. Online Truth Discovery on Time Series Data. In Proceedings of the 2018 SIAM International Conference on Data Mining, San Diego Marriott Mission Valley, San Diego, CA, USA, 3–5 May 2018; pp. 162–170.
59. Nakhaei, Z.; Ahmadi, A.; Sharifi, A.; Badie, K. Conflict resolution in data integration using the relationship between entities. *Int. J. Inf. Commun. Technol. Res.* **2019**, *11*, 38–49.
60. Ye, C.; Wang, H.; Zheng, K.; Kong, Y.; Zhu, R.; Gao, J.; Li, J. Constrained Truth Discovery. *IEEE Trans. Knowl. Data Eng.* **2022**, *34*, 205–218. [[CrossRef](#)]
61. Dong, X.L.; Berti-Équille, L.; Srivastava, D. Integrating Conflicting Data: The Role of Source Dependence. *Proc. VLDB Endow.* **2009**, *2*, 550–561. [[CrossRef](#)]
62. Qi, G.; Aggarwal, C.C.; Han, J.; Huang, T.S. Mining collective intelligence in diverse groups. In Proceedings of the 22nd International World Wide Web Conference, Rio de Janeiro, Brazil, 13–17 May 2013; pp. 1041–1052.
63. Dong, X.L.; Berti-Équille, L.; Srivastava, D. Truth Discovery and Copying Detection in a Dynamic World. *Proc. VLDB Endow.* **2009**, *2*, 562–573. [[CrossRef](#)]
64. Pochampally, R.; Das Sarma, A.; Dong, X.L.; Meliou, A.; Srivastava, D. Fusing data with correlations. In Proceedings of the International Conference on Management of Data, Snowbird, UT, USA, 22–27 June 2014; pp. 433–444.
65. Zhao, B.; Rubinstein, B.I.P.; Gemmell, J.; Han, J. A Bayesian Approach to Discovering Truth from Conflicting Sources for Data Integration. *Proc. VLDB Endow.* **2012**, *5*, 550–561. [[CrossRef](#)]
66. Lin, X.; Chen, L. Domain-aware multi-truth discovery from conflicting sources. *Proc. VLDB Endow.* **2018**, *11*, 635–647. [[CrossRef](#)]
67. Zhi, S.; Zhao, B.; Tong, W.; Gao, J.; Yu, D.; Ji, H.; Han, J. Modeling truth existence in truth discovery. In Proceedings of the 21st ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Sydney, NSW, Australia, 10–13 August 2015; pp. 1543–1552.
68. Li, Y.; Sun, H.; Wang, W.H. Towards fair truth discovery from biased crowdsourced answers. In Proceedings of the 26th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, California, CA, USA, 23–27 August 2020; pp. 599–607.

69. Wang, Y.; Wang, K.; Miao, C. Truth discovery against strategic sybil attack in crowdsourcing. In Proceedings of the 26th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, California, CA, USA, 23–27 August 2020; pp. 95–104.
70. Huang, H.; Fan, G.; Li, Y.; Mu, N. Multi-truth Discovery with Correlations of Candidates in Crowdsourcing Systems. In Proceedings of the Collaborative Computing: Networking, Applications and Worksharing—17th EAI International Conference, Virtual Event, 16–18 October 2021; pp. 18–32.
71. Jiang, L.; Niu, X.; Xu, J.; Yang, D.; Xu, L. Incentive mechanism design for truth discovery in crowdsourcing with copiers. *IEEE Trans. Serv. Comput.* **2021**, *5*, 2838–2853. [\[CrossRef\]](#)
72. Dong, X.L.; Saha, B.; Srivastava, D. Less is More: Selecting Sources Wisely for Integration. *Proc. VLDB Endow.* **2012**, *6*, 37–48. [\[CrossRef\]](#)
73. Nakhaei, Z.; Ahmadi, A.; Sharifi, A.; Badie, K. Conflict resolution using relation classification: High-level data fusion in data integration. *Comput. Sci. Inf. Syst.* **2021**, *18*, 1101–1138. [\[CrossRef\]](#)
74. Yu, Z.; Chu, X. Piclean: A probabilistic and interactive data cleaning system. In Proceedings of the 2019 International Conference on Management of Data, Amsterdam, The Netherlands, 30 June–5 July 2019; pp. 2021–2024.
75. Wang, J.; Tang, N. Towards Dependable Data Repairing with Fixing Rules. In Proceedings of the International Conference on Management of Data, Snowbird, UT, USA, 22–27 June 2014; pp. 457–468.
76. Fan, W.; Li, J.; Ma, S.; Tang, N.; Yu, W. Towards Certain Fixes with Editing Rules and Master Data. *VLDB J.* **2012**, *21*, 213–238. [\[CrossRef\]](#)
77. Bohannon, P.; Fan, W.; Geerts, F.; Jia, X.; Kementsietsidis, A. Conditional Functional Dependencies for Data Cleaning. In Proceedings of the 23rd International Conference on Data Engineering, Istanbul, Turkey, 15–20 April 2007; pp. 746–755.
78. Bohannon, P.; Flaster, M.; Fan, W.; Rastogi, R. A Cost-Based Model and Effective Heuristic for Repairing Constraints by Value Modification. In Proceedings of the ACM SIGMOD International Conference on Management of Data, Baltimore, MD, USA, 14–16 June 2005; Özcan, F., Ed.; ACM: Baltimore, MD, USA, 2005; pp. 143–154.
79. Kolahi, S.; Lakshmanan, L.V.S. On approximating optimum repairs for functional dependency violations. In Proceedings of the Database Theory—ICDT 2009, 12th International Conference, St. Petersburg, Russia, 23–25 March 2009; Volume 361, pp. 53–62.
80. Cong, G.; Fan, W.; Geerts, F.; Jia, X.; Ma, S. Improving Data Quality: Consistency and Accuracy. In Proceedings of the 33rd International Conference on Very Large Data Bases, Vienna, Austria, 23–27 September 2007; University of Vienna: Vienna, Austria, 2007; pp. 315–326.
81. Fan, W.; Geerts, F.; Ma, S.; Müller, H. Detecting inconsistencies in distributed data. In Proceedings of the 26th International Conference on Data Engineering, Long Beach, CA, USA, 1–6 March 2010; pp. 64–75.
82. Fan, W.; Li, J.; Tang, N.; Yu, W. Incremental Detection of Inconsistencies in Distributed Data. In Proceedings of the IEEE 28th International Conference on Data Engineering, Washington, DC, USA, 1–5 April 2012; pp. 318–329.
83. Li, P.; Dai, C.; Wang, W. Inconsistent Data Cleaning Based on the Maximum Dependency Set and Attribute Correlation. *Symmetry* **2018**, *10*, 516. [\[CrossRef\]](#)
84. Chu, X.; Ilyas, I.F.; Papotti, P. Holistic data cleaning: Putting violations into context. In Proceedings of the 29th IEEE International Conference on Data Engineering, Brisbane, Australia, 8–12 April 2013; pp. 458–469.
85. Rekatsinas, T.; Chu, X.; Ilyas, I.F.; Ré, C. HoloClean: Holistic Data Repairs with Probabilistic Inference. *Proc. VLDB Endow.* **2017**, *10*, 1190–1201. [\[CrossRef\]](#)
86. Fan, W.; Wu, Y.; Xu, J. Functional Dependencies for Graphs. In Proceedings of the 2016 International Conference on Management of Data, San Francisco, CA, USA, 26 June–1 July 2016; pp. 1843–1857.
87. Fan, W.; Liu, X.; Lu, P.; Tian, C. Catching Numeric Inconsistencies in Graphs. In Proceedings of the 2018 International Conference on Management of Data, Houston, TX, USA, 10–15 June 2018; pp. 381–393.
88. Lin, P.; Song, Q.; Wu, Y.; Pi, J. Repairing Entities using Star Constraints in Multirelational Graphs. In Proceedings of the 36th IEEE International Conference on Data Engineering, Dallas, TX, USA, 20–24 April 2020; pp. 229–240.
89. Schneider, S.; Lambers, L.; Orejas, F. A Logic-Based Incremental Approach to Graph Repair. In Proceedings of the Fundamental Approaches to Software Engineering—22nd International Conference, Held as Part of the European Joint Conferences on Theory and Practice of Software, Prague, Czech Republic, 6–11 April 2019; Volume 11424, pp. 151–167.
90. Bravo, L.; Bertossi, L.E. Logic Programs for Consistently Querying Data Integration Systems. In Proceedings of the Eighteenth International Joint Conference on Artificial Intelligence, Acapulco, Mexico, 9–15 August 2003; pp. 10–15.
91. Cali, A.; Lembo, D.; Rosati, R. On the decidability and complexity of query answering over inconsistent and incomplete databases. In Proceedings of the Twenty-Second ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems, San Diego, CA, USA, 9–12 June 2003; pp. 260–271.
92. Wijzen, J. Condensed Representation of Database Repairs for Consistent Query Answering. In Proceedings of the Database Theory—ICDT 2003, 9th International Conference, Siena, Italy, 8–10 January 2003; Calvanese, D., Lenzerini, M., Motwani, R., Eds.; Springer: Siena, Italy, 2003; Volume 2572, pp. 375–390.
93. Yakout, M.; Elmagarmid, A.K.; Neville, J.; Ouzzani, M.; Ilyas, I.F. Guided data repair. *Proc. VLDB Endow.* **2011**, *4*, 279–289. [\[CrossRef\]](#)
94. Cheng, Y.; Chen, L.; Yuan, Y.; Wang, G. Rule-Based Graph Repairing: Semantic and Efficient Repairing Methods. In Proceedings of the 34th IEEE International Conference on Data Engineering, Paris, France, 16–19 April 2018; pp. 773–784.
95. Fan, W.; Lu, P.; Tian, C.; Zhou, J. Deducing Certain Fixes to Graphs. *Proc. VLDB Endow.* **2019**, *12*, 752–765. [\[CrossRef\]](#)

96. Fan, W.; Jin, R.; Liu, M.; Lu, P.; Tian, C.; Zhou, J. Capturing Associations in Graphs. *Proc. VLDB Endow.* **2020**, *13*, 1863–1876. [[CrossRef](#)]
97. Liu, G.; Li, L. Knowledge Fragment Cleaning in a Genealogy Knowledge Graph. In Proceedings of the 2020 IEEE International Conference on Knowledge Graph, Online, 9–11 August 2020; pp. 521–528.
98. Ye, C.; Li, Q.; Zhang, H.; Wang, H.; Gao, J.; Li, J. AutoRepair: An automatic repairing approach over multi-source data. *Knowl. Inf. Syst.* **2019**, *61*, 227–257. [[CrossRef](#)]
99. Ye, C.; Wang, H.; Zheng, K.; Gao, J.; Li, J. Multi-source data repairing powered by integrity constraints and source reliability. *Inf. Sci.* **2020**, *507*, 386–403. [[CrossRef](#)]
100. Fan, W.; Geerts, F.; Li, J.; Xiong, M. Discovering conditional functional dependencies. *IEEE Trans. Knowl. Data Eng.* **2010**, *23*, 683–698. [[CrossRef](#)]
101. Rezig, E.K.; Ouzzani, M.; Aref, W.G.; Elmagarmid, A.K.; Mahmood, A.R.; Stonebraker, M. Horizon: Scalable dependency-driven data cleaning. *Proc. VLDB Endow.* **2021**, *14*, 2546–2554. [[CrossRef](#)]

Disclaimer/Publisher’s Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.