applied sciences

MDPI

Article

Dataset Similarity Detection for Reuse Protection in Federated Data Spaces with Privacy Considerations

Christos Panagiotou, Artemios G. Voyiatzis and Kyriakos Stefanidis *

Industrial Systems Institute, Athena Research Center, PSP Building, Stadiou Str., Platani, 26504 Patras, Greece; panagiotou@athenarc.gr (C.P.); bogart@athenarc.gr (A.G.V.)
* Correspondence: stefanidis@athenarc.gr

Abstract

Federated data spaces, established through initiatives such as IDSA and GAIA-X, enable organizations to share and monetize datasets under contractual terms. However, enforcing these contracts—particularly detecting unauthorized reuse or modification of datasets—remains an open challenge. We present the Off-Platform Contract Inspector, a component of the PISTIS framework, that implements a modular similarity-detection pipeline combining path-value Jaccard similarity, field-aware type-specific comparisons, and sentence-embedding-based semantic analysis across structured, semi-structured, and unstructured datasets. This contributes as follows: (i) an Inverse Document Frequency (IDF)-weighted structural similarity mechanism that discounts common domain vocabulary via Inverse Document Frequency weighting over the data space catalog, combined with a schema-evidence-gated fusion that reduces false positives from domain vocabulary overlap; (ii) an adaptive threshold optimization mechanism that learns modality-specific fusion weights and decision thresholds via cross-validated grid search; and (iii) a privacy-preserving similarity layer based on MinHash Locality-Sensitive Hashing signatures, Bloom filters with OR folding alignment, and Laplace noise for differential privacy, enabling cross-organizational dataset comparison without exposing raw data. Further, we contribute a threat taxonomy of seven dataset modification types ordered by detection difficulty, and evaluate the system on dataset pairs derived from real-world datasets across three smart-city application domains (Mobility, Energy, Automotive), with controlled augmentations applied to model adversarial behaviors. The IDF-weighted pipeline achieves high precision on intra-domain hard negatives—pairs of different tables from the same data space that share domain vocabulary—where text-similarity baselines produce false positives. The adaptive scheme learns per-modality fusion weights via cross-validated grid search. The privacy-preserving mode operates without accessing raw data and runs noticeably faster than the full pipeline, enabling screening while preserving data confidentiality.

Keywords: federated data spaces; data sovereignty; threat taxonomy; data reuse detection; data similarity; locality-sensitive hashing; privacy-preserving computation; smart cities

Check for updates

Academic Editor: Luis Javier Garcia Villalba

Received: 24 April 2026
Revised: 30 May 2026
Accepted: 8 June 2026
Published: 11 June 2026

Copyright: © 2026 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.

1. Introduction

The European data economy is evolving rapidly, driven by regulatory frameworks such as the Data Governance Act [1] and the Data Act [2], as well as technical reference architectures such as IDSA [3] and GAIA-X [4]. These initiatives are establishing *data spaces*, i.e., data ecosystems in which organizations can share, exchange, and license datasets under contractual terms [5]. By and large, a data space consists of a set of participants, i.e.,

organizations that produce, consume, or broker data; a catalog of the available datasets; a set of contracts specifying the terms under which each dataset may be accessed and used; and a governance framework defining the rules, roles, and enforcement mechanisms that regulate the data space operation and participants.

Data providers will participate in data spaces and marketplaces only if contractual terms governing their datasets are respected, retaining their data sovereignty. In practice, detecting unauthorized reuse, redistribution, or modification of traded datasets once these leave the perimeter of a provider, including the transfer to the infrastructure of a centralized marketplace, remains an open challenge. Established approaches based on cryptographic hashing of datasets can identify exact copies, but are trivially defeated by even minor content modifications, while provenance tracking systems rely on cooperative reporting that cannot be assumed in adversarial settings.

Federated data spaces aim to address the aforementioned challenges by ensuring that datasets remain at providers' nodes and are only processed by third-party nodes that respect the rules and regulations of the federation. The federated data space governance is addressed at multiple levels. At the policy level, reference architectures such as IDSA [3] and GAIA-X [4] define standardized usage control policies and data sovereignty principles that govern how data may be accessed, processed, and redistributed within a federated data space. At the enforcement level, blockchain-based approaches automate contract execution through smart contracts and provide tamper-proof audit trails of data transactions [6–8].

The PISTIS framework offers a comprehensive platform to realize federated data spaces for monetizing, trading, and safeguarding data transactions through a set of components, among which the *Contract Inspector* [9]. The PISTIS *On-Platform* Contract Inspector enforces usage policies *within* the marketplace boundaries, leveraging the controlled environment to monitor access patterns and ensure compliance with contractual terms [10]. Complementary, the *Off-Platform* Contract Inspector combines structural and semantic analysis to detect unauthorized dataset reuse in the process of introducing a new dataset to the platform [10,11]. In this paper, we enhance and extend the PISTIS Off-Platform Contract Inspector in five dimensions:

1. A threat taxonomy that categorizes the types of dataset modifications an adversary might employ to evade detection (Section 3).
2. An IDF-weighted structural similarity mechanism with schema-evidence-gated fusion that discounts common domain vocabulary and reduces false positives on intra-domain hard negatives (Section 4.3).
3. An adaptive, modality-aware threshold optimization mechanism supported by learned, per-modality fusion weights and decision thresholds (Section 4.8).
4. A privacy-preserving similarity layer based on Locality-Sensitive Hashing (LSH) and Bloom filters, which enables cross-organizational dataset comparison without exposing raw data to any party (Section 4.9).
5. A thorough evaluation of the proposed modular pipeline using a corpus of 966 dataset pairs comprising real-world, processed, and synthetic datasets (Section 6).

The remainder of this paper is organized as follows. Section 2 surveys related work across near-duplicate detection, semantic similarity, and data provenance, positioning our contribution within the existing literature. Section 3 defines the dataset similarity detection problem and introduces the threat taxonomy. Section 4 describes the designed architecture and modular similarity detection processing pipeline. Section 5 reports on our experimental evaluation approach across three application domains and real-world datasets. Section 6 presents the evaluation results and discusses the implications of our findings. Finally, Section 7 concludes with directions for future work.

2. Related Work

2.1. Near-Duplicate Detection

In the context of data spaces, existing systems like MISP [12] and the Eclipse Dataspace Connector (EDC) [13] rely on cryptographic hash functions (e.g., SHA-256) to verify data integrity. While a cryptographic hash provides perfect precision for exact matches, it is brittle: Even a single-byte modification in a dataset produces a completely different cryptographic hash, rendering the approach ineffective against any form of near-duplicate or modified reuse.

The problem of detecting near-duplicate content has been extensively studied in the context of web documents. Broder [14] introduced the *MinHash* sketching technique, which provides an efficient probabilistic approximation of the Jaccard similarity coefficient between two sets. This foundational work was subsequently extended in [15] and applied to web-scale deduplication tasks, where billions of web pages must be efficiently compared [16,17]. Charikar [18] proposed *SimHash*, an alternative fingerprinting technique that enables de-duplication through Hamming distance computation on compact binary signatures. These methods are effective for textual content, as they were designed for unstructured documents. However, they do not account for the hierarchical structure, typed fields, and schema information that characterize structured datasets, such as JSON records, CSV files, and relational database tables.

Our approach differs in two aspects. First, it incorporates the characteristics of *structured* and *semi-structured datasets*, exploiting their schema and field-level organization to provide fine-grained similarity analysis. Second, it produces *interpretable, field-level similarity reports* that explain *which* parts of two datasets overlap and *how*, which is needed for contract enforcement in data marketplaces.

2.2. Semantic Similarity and Embeddings

Sentence and document embedding models enable semantic similarity computation at scale. SBERT adapts the BERT architecture using Siamese and triplet network structures to produce dense vector representations that capture semantic meaning, enabling efficient cosine similarity computation between text passages [19]. More recently, the E5 family of models achieves strong performance on retrieval and similarity benchmarks [20]. Libraries such as FAISS further enable scalable nearest-neighbor retrieval over millions of embedding vectors with sub-linear query time [21].

Embedding-based approaches are less effective when applied to structured dataset comparison. When applied to tabular or hierarchical data, embeddings tend to conflate structural properties (e.g., field names and nesting depth) with content-level semantics (e.g., the actual values stored in fields), making it difficult to distinguish between datasets that share a schema but contain different data, and datasets that are genuine copies with modified field names. Furthermore, embedding similarity produces a single scalar score that offers no insight into *which* fields or records overlap, an interpretability that is needed for reporting contract violations.

In our approach, we use embeddings as just one layer within the pipeline to handle free-text and unstructured content. Additionally, dedicated structural and field-level comparators provide the interpretability needed for contract enforcement.

2.3. Data Provenance and Privacy-Preserving Comparison

Data provenance systems track the data lineage and transformation history as the datasets flow through processing pipelines [22,23]. In principle, provenance metadata can reveal whether a dataset was derived from a licensed source, and under which conditions. Provenance-based approaches depend on faithful reporting by all parties participating in

the data processing chain. However, in adversarial settings, a malicious actor in the chain can strip, forge, or omit provenance metadata.

Our content-based approach provides similarity evidence by directly examining the dataset content. It has no dependence on metadata availability and quality, and makes no assumptions about their trustworthiness.

Privacy-preserving computation is relevant in federated data spaces, where organizations must compare datasets without revealing their contents. Homomorphic Encryption allows computation on encrypted data [24], and Secure Multi-Party Computation enables joint computation without any party learning more than the output [25,26]. While both offer strong cryptographic guarantees, they impose substantial computational overhead that limits their practicality for large-scale, routine dataset comparison tasks. On the other hand, Locality-Sensitive Hashing (LSH) provides a practical middle ground: by mapping similar items to the same hash buckets with high probability, LSH enables approximate similarity estimation from compact fingerprints without exposing the original data [27,28].

In our work, we adopt LSH as the foundation of our privacy-preserving layer, augmented by Bloom filters [29] for structural (schema-level) comparison and calibrated Differential Privacy noise [30] for score reporting. Our approach ensures that the reported similarity values do not leak information about individual records.

3. Data Reuse: Problem Formulation and Threat Model

It is necessary to protect a federated data space from data reuse threats. In this context, data reuse is defined as any attempt to inject into the data space a reprocessed dataset that is similar to one(s) already existing in the federated catalog, making the former appear as if it were from a different origin or with different characteristics than the originals.

There are many practical challenges associated with reuse detection. First, *dataset heterogeneity*: datasets within a data space span different modalities (e.g., structured JSON, flat CSV, or free-text documents) and widely vary in size, schema complexity, and content density, requiring the detection function to adapt its strategy accordingly. Second, *adversarial modifications*: an adversary may alter a dataset to evade detection, applying transformations such as field renaming, value perturbation, or partial extraction. Third, *partial reuse*: a dataset may incorporate only a subset of records or fields from a licensed source, requiring the system to detect overlap even when the reused portion constitutes a small fraction of the submitted dataset. Fourth, *interpretability*: for a detection result to be actionable, the system must provide an explanation of *which* fields and records overlap rather than merely a (vague) scalar similarity score. Fifth, *privacy constraints*: in cross-organizational settings, the dataset comparison must be performed without exposing the raw data of any dataset. Sixth, *scalability*: the system must handle catalogs with thousands of datasets, requiring efficient indexing and comparison strategies.

To provide a more concrete basis for analysis, we define in the following a taxonomy of seven threat types, T1 to T7, that an adversary might apply to disguise a dataset's origin.

T1 (Exact Replication) represents the simplest case, in which a dataset is redistributed without any modification. While trivially detectable via cryptographic hashing, this scenario serves as a baseline and occurs in practice when actors are unaware of data space monitoring capabilities.

T2 (Format Transformation) involves converting the dataset to a different serialization format (e.g., from JSON to CSV), while preserving the underlying content. Such transformations alter the byte-level representation but do not change the logical structure or values.

T3 (Structural Modification) encompasses changes to the dataset schema, including renaming fields (T3a) and injecting additional spurious fields that carry no meaningful data (T3b). These modifications are designed to reduce the apparent structural overlap between the original and the modified copy.

T4 (Value Perturbation) involves modifying the actual data values through operations such as rounding numerical fields, adding random noise, or substituting synonyms in text fields. The severity ranges from mild (e.g., rounding to fewer decimal places) to aggressive (e.g., adding 10–20% Gaussian noise to all numerical fields).

T5 (Partial Extraction) represents scenarios where only a subset of the original dataset is reused, either by selecting a subset of records (row extraction) or a subset of fields (column extraction). This is common in practice, as a consumer may only need a portion of a licensed dataset for their needs.

T6 (Semantic Obfuscation) involves substituting field names and categorical values with synonyms or near-synonyms to alter surface-level expression while preserving meaning. This can be achieved using dictionary-based synonym substitution, where field names and string values are replaced according to a curated domain-aware synonym mapping (e.g., the field "name" becomes "identifier", "city" becomes "municipality", and"status" becomes "state"). This type of modification targets structural detection by reducing lexical overlap at the schema level without changing the underlying semantics.

T7 (Composite Attack) combines two or more of the above threat types. For example, an attacker might extract a subset of records (T5), rename several fields (T3a), and add noise to some numerical fields (T4), aiming to maximize the difficulty of detection.

4. Similarity-Detection Pipeline: Architecture and Design

4.1. Preliminaries

Once a dataset is *inside* the PISTIS perimeter, the PISTIS framework ensures that unauthorized data processing, i.e., a contract violation, cannot occur [9]. Hence, it defends by design against all the threats mentioned in Section 3. However, it does not protect during the *initial introduction* of a dataset in the platform, i.e., during dataset injection.

In the next section, we present the architecture and implementation of a modular similarity-detection pipeline, the core part of the PISTIS Off-Platform Contract Inspector [10,11], designed specifically to address this security challenge. For the reader's convenience, Table 1 summarizes how our proposed approach is positioned relative to existing methods presented in Section 2 across the key dimensions of near-duplicate detection, partial reuse handling, structured data support, interpretability, and privacy preservation.

Table 1. Comparison with related approaches (✓: Yes, ×: No).

Approach	Near-Dup.	Partial	Structured	Interpretable	Privacy
Cryptographic hashing	Exact only	×	×	×	✓
MinHash/SimHash	✓	×	×	×	Partial
SBERT/E5 embeddings	✓	×	×	×	×
Provenance tracking	×	×	✓	✓	Varies
This work	✓	✓	✓	✓	✓

Respectively, Table 2 maps each threat type presented in Section 3 to a suggested detection mechanism in our design, along with a qualitative assessment of the associated detection difficulty.

Table 2. Threat types and detection mechanisms.

Type	Description	Primary Detection	Detection Difficulty
T1	Exact replication	Hash fingerprint	Low
T2	Format transformation	Path-value normalization	Low
T3	Structural modification	Jaccard path similarity	Medium
T4	Value perturbation	Field-aware value sim.	Medium
T5	Partial extraction	Sliding-window embedding	High
T6	Semantic obfuscation	SBERT/E5 embedding sim.	High
T7	Composite attack	Multi-layer fusion	Very High

4.2. Architecture

The PISTIS Off-Platform Inspector is designed as an asynchronous microservice that can be deployed independently or integrated within the broader PISTIS infrastructure. Requests to the Inspector are processed via a task queue that supports horizontal scaling across multiple worker nodes. As depicted in Figure 1, the enhanced system architecture has six components, each addressing a stage of the detection pipeline.

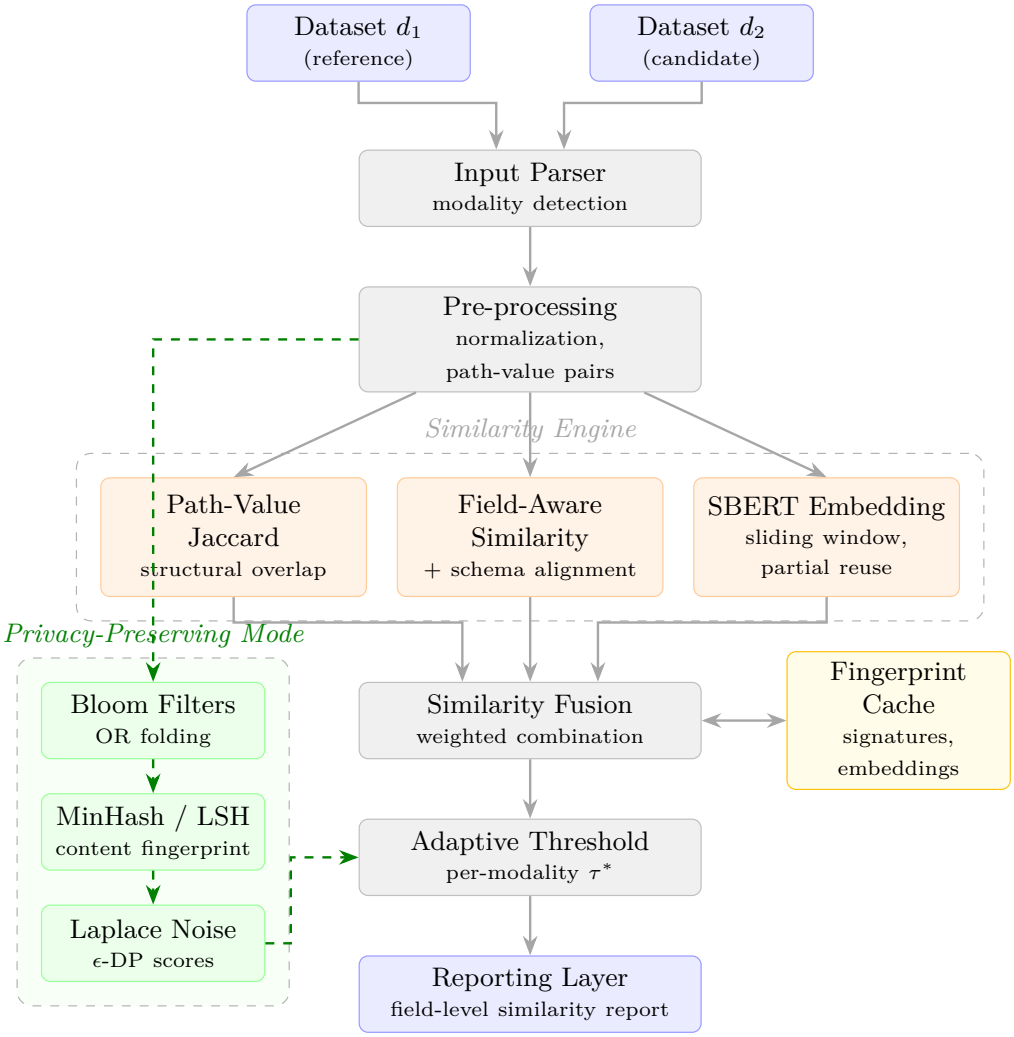


Figure 1. Architecture of the PISTIS Off-Platform Contract Inspector.

The *Input Parser* serves as the entry point, accepting datasets in multiple formats and automatically detecting their modality (JSON, CSV, XML, or unstructured text). This is needed because datasets within a data space may use different serialization formats.

The *Pre-processing Module* transforms the parsed input into a canonical internal representation. For structured and semi-structured data, this involves flattening hierarchical structures into path-value pairs, normalizing whitespace and encoding, and standardizing numerical representations. This ensures that superficial formatting differences (such as different JSON indentation styles or varying decimal precision) do not artificially inflate dissimilarity.

The *Similarity Engine* implements the multi-layer comparison strategy described in Sections 4.3–4.7 below. Its flexible design allows different similarity modules to be activated or deactivated, depending on the dataset modality and the desired trade-off between thoroughness and computational cost. Further, its extensible design allows new similarity modules to be added in the future.

The *Fingerprint Cache* stores precomputed fingerprints (e.g., MinHash signatures, Bloom filters, and embedding vectors) for datasets already indexed in the catalog. When a new dataset is submitted for comparison, the cache eliminates the need to reprocess the entire catalog, reducing the comparison to a lookup-and-compare operation against cached fingerprints.

The *Privacy-Preserving Module* is a new layer, purposely designed to enable cross-organizational dataset comparison without exposing raw data to any party. This module can be activated in federated data spaces with strong privacy considerations.

Finally, the *Reporting Layer* aggregates the outputs of the similarity engine into structured, field-level similarity reports. These reports identify not only whether a match was found but also which specific fields and records overlap, the type of similarity detected (structural, semantic, or value-level), and the confidence level of the match. For example, a report for a field renaming violation (T3a) might contain entries such as: "`flight_id` ↔ `fl_identifier`: *value match 98.5% (Levenshtein), 40/40 records aligned;* `departure_time` ↔ `dep_time`: *value match 100% (exact), field names aligned via semantic similarity (cosine = 0.91); overall structural overlap: 0.35 (Jaccard on original paths), value overlap: 0.97 (after schema alignment), combined score: 0.87.*" This field-level detail is needed for contract dispute resolution, where a scalar score alone is not sufficient. We note that the dataset similarity report provides *indications* of dataset reuse, not a proof of contract violation. It is up to the PISTIS governance layer to engage in the contract dispute resolution and enforce the sanctions specified for contract breaches.

4.3. Path-Value Extraction and Path Similarity

The first processing layer of the Similarity Engine pipeline addresses structural comparison. Structured and semi-structured datasets are initially flattened by the Preprocessing Module into a set of canonical path-value pairs, where each pair consists of a root-to-leaf path through the data hierarchy and the corresponding leaf value. For example, a JSON object `{"user": {"name": "Alice"}}` would yield the path-value pair `user.name: Alice`.

The flattened representation provides a uniform basis for comparison across different serialization formats. The structural overlap between two flattened datasets, say d_1 and d_2, is then measured using the Jaccard similarity coefficient computed over their path-value (PV) sets as

$$S_{\text{Jaccard}}(d_1, d_2) = \frac{|\text{PV}(d_1) \cap \text{PV}(d_2)|}{|\text{PV}(d_1) \cup \text{PV}(d_2)|} \tag{1}$$

In a data space catalog containing many datasets from the same application domain, certain schema paths (e.g., `timestamp`, `id`, and `status`) appear across unrelated datasets simply because they are standard field names in that domain. The standard Jaccard coefficient treats all paths equally, which inflates similarity scores for *intra-domain hard*

negatives, i.e., domain-related but genuinely distinct datasets. To address this inefficiency, we apply Inverse Document Frequency (IDF) weighting over the catalog corpus. Given a collection of N datasets, the IDF weight of a schema path p is defined as

$$w_{\text{IDF}}(p) = \log \frac{N}{df(p)}, \tag{2}$$

where $df(p)$ is the number of datasets in the catalog that contain path p. Paths that appear in every dataset get an IDF weight of $\log(1) = 0$, effectively eliminating their contribution to the similarity score, while paths unique to a single dataset get the maximum IDF weight of $\log(N)$. The IDF-weighted Jaccard similarity replaces, hence, the standard set intersection with an IDF-weighted sum

$$S^{\text{IDF}}_{\text{Jaccard}}(d_1, d_2) = \frac{\sum_{p \in P_1 \cap P_2} w_{\text{IDF}}(p)}{\sum_{p \in P_1 \cup P_2} w_{\text{IDF}}(p)}, \tag{3}$$

where P_1 and P_2 denote the sets of schema-level paths (with row indices stripped) extracted from datasets d_1 and d_2, respectively. We note that the IDF weights are computed only once over the full catalog and held fixed, ensuring that the weighting reflects the global distribution of paths across the data space.

4.4. Field-Aware Value Similarity

While $S^{\text{IDF}}_{\text{Jaccard}}$ only captures structural overlap, S_{Jaccard} treats path-value pairs as atomic tokens and assigns zero similarity to pairs that share a path but have slightly different values. To address these limitations, the second layer of the pipeline performs fine-grained, type-aware comparison of the actual data values associated with shared paths. For each path p that appears in both datasets, $P_{\text{shared}} = P_1 \cap P_2$, a type-specific similarity function is applied: numerical field values are compared using a normalized absolute difference (yielding a score of 1.0 for identical values and decreasing proportionally with the magnitude of the difference); short string fields (such as names, identifiers, and categorical labels) are compared using the Levenshtein ratio [31], which captures character-level edit distance; and longer free-text fields are compared using SBERT cosine similarity [19], which captures semantic equivalence even when the surface phrasing differs. The field-wise value similarity scores $s_p(v_1^p, v_2^p)$ are then averaged across all shared paths to produce an aggregate value similarity

$$S_{\text{values}}(d_1, d_2) = \sum_{p \in P_{\text{shared}}} \frac{1}{|P_{\text{shared}}|} s_p(v_1^p, v_2^p) \tag{4}$$

4.5. Path-Value Similarity Fusion

The structural similarity (Equation (3)) and the value similarity (Equation (4)) capture complementary aspects of dataset overlap: the former measures how much of the data structure is shared, while the latter how closely the actual data values match within that shared structure. To derive a single overall similarity score, the two signals are fused via a weighted linear combination

$$S_{\text{combined}}(d_1, d_2) = \alpha \cdot S^{\text{IDF}}_{\text{Jaccard}}(d_1, d_2) + (1 - \alpha) \cdot S_{\text{values}}(d_1, d_2) \tag{5}$$

For the sake of clarity, we defer the discussion of the optimal α to Section 4.8.

4.6. Schema Alignment for Renamed Fields

Structural comparison is susceptible to field renaming (threat T3a), where one manipulates field names to reduce path overlap. The IDF-weighted Jaccard similarity in Equation (3) would indeed suffer a sharp drop should equivalent field names be changed. The same holds for the aggregate value similarity in Equation (4), should the number of shared paths decrease. To maintain robustness against field renaming, the pipeline uses two approaches.

First, when the path overlap is low, but the value one is high, the system performs *value-distribution matching*: for each unmatched path in d_1, it computes a distributional fingerprint of the associated values (type, cardinality, numerical range, or string-length histogram) and compares it against unmatched paths in d_2. Paths whose value distributions are statistically compatible above a configurable similarity threshold are tentatively aligned, and only then are their values compared using the field-aware similarity described in Section 4.4.

Second, for textual field names, the system computes the SBERT cosine similarity of the field names (e.g., "departure_time" vs. "dep_time"), aligning fields whose names are semantically equivalent, even when lexically different. The aligned field pairs are then included in the value similarity computation, recovering the signal that would have been lost due to strict path matching.

4.7. Embedding-Based Comparison

The structural and value-based comparison layers are effective for datasets with parseable schemata, but they cannot handle unstructured data (e.g., free-text documents and system logs) or cases where the structural parsing fails due to irregular formatting. To address this, the pipeline includes an embedding-based comparison layer that operates on dense vector representations of the dataset content. Specifically, we use the SBERT model `all-MiniLM-L6-v2` (with the option to substitute with E5 or other models), which encodes the textual representation of a dataset into a fixed-dimensional embedding vector. The cosine similarity between two such embedding vectors captures their overall semantic relatedness, including shared topics, vocabulary, and content patterns.

A manipulated dataset may contain only a fragment of the original. In such cases, the overall embedding of the smaller dataset may not match well the embedding of the original, since the latter contains substantial additional content that dilutes the matching signal. We apply a *sliding-window comparison strategy* to address this. For tabular data, a "window" consists of k consecutive records (rows); for semi-structured data (e.g., JSON arrays), a "window" comprises k consecutive top-level objects. Each window's records are serialized to text (concatenating field names and values) and independently embedded using the same SBERT model. The window size k is set to match the number of records in the smaller dataset d_s. The original (larger) dataset d_l is traversed with a stride of $k/2$ (50% overlap) to ensure that the relevant segment is not split across window boundaries. To avoid false positives from boilerplate content (e.g., repeated header rows or schema preambles), the serialization excludes field names and includes only the data values. Hence, the similarity is driven by content overlap rather than shared schema vocabulary. The embedding-based partial reuse score is then computed as the maximum cosine similarity between the smaller dataset's embedding $\mathbf{e}_{d_s}$ and that of any window $\mathbf{e}_w$:

$$S_{\text{partial}}(d_s, d_l) = \max_{w \in \text{windows}(d_l, k)} \cos(\mathbf{e}_{d_s}, \mathbf{e}_w) \tag{6}$$

This approach ensures that even if a small subset of records was extracted from a large dataset, the window containing the corresponding records will produce a high similarity

score. Equation (6) captures semantic similarity at the content level, which is valuable for detecting paraphrased or reformatted copies.

Datasets from the same application domain can produce elevated embedding similarity scores even when they share no actual data provenance, because sentence embeddings encode general semantic content. To account for this, we gate the embedding contribution by the degree of schema evidence. Specifically, we compute the *schema recall*, i.e., the fraction of the smaller dataset's schema paths that are shared with the larger dataset

$$r_{\text{schema}}(P_1, P_2) = \frac{|P_1 \cap P_2|}{\min(|P_1|, |P_2|)} \tag{7}$$

The gated-partial-reuse score is then

$$S^{\text{gated}}_{\text{partial}} = r_{\text{schema}} \cdot S_{\text{partial}} \tag{8}$$

and the final detection score for a dataset pair is computed as

$$\max(S_{\text{combined}}, S^{\text{gated}}_{\text{partial}}) \tag{9}$$

When two datasets share substantial schema structure, i.e., $r_{\text{schema}} \to 1^-$, the embedding score $S^{\text{gated}}_{\text{partial}}$ passes through at full strength; when they have disjoint schemata, i.e., $r_{\text{schema}} \to 0^+$, the embedding contribution is suppressed, and the decision relies solely on the structural and value-based components S_{combined}. This ensures that embedding-based similarity is used as corroborating evidence for structural relatedness, rather than as an independent indicator that can override structural dissimilarity. Thus, it prevents false positives due to domain vocabulary overlap.

4.8. Adaptive Fusion Threshold Optimization

The fusion weight $\alpha = 0.5$ for Equation (5), as well as a decision threshold $\tau = 0.7$, were set as fixed constants in [11], based on empirical observation over a limited set of test cases. These values indeed provide a reasonable default performance. However, they do not account for the significant differences in how structural and value-level similarities behave across different dataset modalities. For example, JSON datasets with rich hierarchical schemata, where the path structure carries discriminative information, benefit from a higher structural weight. In contrast, a CSV file has a flat, single-level schema (column headers) that is far less informative for structural comparison and, thus, benefits from emphasizing value-level comparison. A fixed fusion weight that works well for one modality may be suboptimal for another.

To address this limitation, we introduce here an *adaptive threshold optimization mechanism* that learns per-modality parameters from labeled training data. The optimization procedure operates as follows. For each dataset modality (structured JSON, structured CSV, unstructured text), we perform a grid search over the fusion weight $0 \leq \alpha \leq 1$ with a step size of 0.05, evaluating each candidate value α using five-fold stratified cross-validation. The stratification ensures that each fold maintains the same proportion of similar and dissimilar pairs as the full dataset, preventing biased evaluation. For each value and each fold, we compute the full Receiver Operating Characteristic (ROC) curve and select the optimal decision threshold by maximizing Youden's index $J = \text{TPR} - \text{FPR}$, which identifies the threshold that best balances sensitivity (TPR, True Positive Rate) and specificity (FPR, False Positive Rate). The value α that achieves the highest mean F_1-score across the five folds is selected as the optimal fusion weight α^* and the corresponding threshold τ^* is derived from the Youden-optimal point on the aggregate ROC curve. The resulting modality

profiles specify optimal (α^*, τ^*) pairs that are stored and applied automatically based on the detected modality of future datasets.

4.9. Privacy-Preserving Similarity Detection

Dataset comparison in a federated data space often involves multiple parties that are not willing or, due to regulatory constraints, not permitted to share their raw data with each other. For instance, a PISTIS-based data marketplace operator may need to verify that a newly-submitted dataset does not infringe on the intellectual property of an existing provider, but neither party wishes to expose their datasets to the other or to the marketplace operator itself. This cross-organizational comparison scenario imposes three privacy requirements: (i) *content confidentiality*, ensuring that the raw records and values of each dataset remain accessible only to their owner; (ii) *schema privacy*, protecting the structural organization of the datasets (field names, hierarchies) from disclosure; and (iii) *result minimality*, ensuring that the comparison output reveals only the minimum information necessary for a violation decision (i.e., a similarity score and a binary verdict), without enabling reconstruction of the underlying data.

The privacy-preserving layer we propose for the similarity-detection pipeline satisfies these requirements by employing a three-stage protocol entirely operating on compact, non-invertible fingerprints:

1. MinHash Signature Computation (Content Fingerprinting). Each party locally computes a MinHash signature over their dataset's path-value pairs. The MinHash technique generates a fixed-length vector of hash values that serves as a compact fingerprint of the set, with the property that the fraction of matching positions between two signatures provides an unbiased estimate of the Jaccard similarity between the underlying sets. The original path-value pairs cannot be reconstructed from the signature alone. The signature length n controls the trade-off between estimation accuracy and privacy: longer signatures provide more precise similarity estimates but also reveal more information about the dataset's characteristics.
2. Bloom Filter Encoding (Structural Fingerprinting). To enable structural (schema-level) comparison without revealing field names, each party encodes their dataset's paths into a Bloom filter—a space-efficient probabilistic data structure that supports set membership queries with a controlled false-positive rate. When two parties have datasets with different schema sizes, their Bloom filters may have different lengths. To enable consistent bitwise comparison, we apply *OR folding*: the larger filter is reduced to the size of the smaller filter by partitioning the former into equally-sized segments and combining them via bitwise OR operation. OR folding preserves set membership: if a bit was set in any segment of the original filter, it remains set in the folded filter. This is equivalent to inserting all original elements into a smaller filter. The false-positive rate of the folded filter is bounded by $p' \approx (1 - e^{-kn/m'})^k$, where k is the number of hash functions, n is the element count, and m' is the folded size. This procedure preserves the approximate set intersection properties of the Bloom filters while ensuring that the comparison operates on representations of equal size.
3. Noise-Based Score Obfuscation. Even after computing similarity from fingerprints rather than raw data, the reported similarity score itself could potentially leak information—for example, a score very close to 1.0 reveals that the datasets are nearly identical. To mitigate this, we add calibrated Laplace noise to the similarity score before reporting. We note that this provides *score-level* privacy (obscuring the precise similarity value) rather than *record-level* differential privacy over the underlying datasets—the latter would require a different mechanism. The similarity score is bounded in the range of $[0, 1]$, giving a global sensitivity of $\Delta f = 1$ (changing one

dataset can shift the score by at most 1). We draw noise from Laplace$(0, 1/\epsilon)$, and clip the result to $[0, 1]$. Under the bounded query model, this provides ϵ-differential privacy for the reported score. The noise scale is inversely proportional to ϵ: smaller ϵ values provide stronger obfuscation at the cost of greater score perturbation.

In this protocol, only compact fingerprints (MinHash signatures and Bloom filters) are exchanged between parties; raw data never leaves the provider's perimeter. We note that fingerprint-based approaches provide *computational* rather than *information-theoretic* privacy.

While the MinHash signatures and Bloom filters are not directly invertible, they are still susceptible to dictionary attacks: an adversary who can enumerate candidate path-value pairs can check whether a specific pair is consistent with the observed fingerprint. To mitigate this risk, both parties agree on a secret "salt" via a secure key exchange protocol at protocol initiation. During the protocol execution, each party applies the secret, locally held random salt, when computing hash functions for both MinHash and Bloom filter construction. Since both parties use the same salt, the calculated fingerprints remain comparable, but an external observer cannot mount dictionary attacks.

The theoretical properties of the underlying techniques provide guarantees on the accuracy of the privacy-preserving comparison. A MinHash signature of length n estimates the true Jaccard similarity with an expected error bounded by $O(1/\sqrt{n})$, meaning that doubling the signature length reduces the estimation error by a factor of approximately $\sqrt{2}$. The Bloom filter comparison introduces a false-positive rate that depends on the filter's saturation level (the fraction of bits set to 1), which in turn depends on the ratio of inserted elements to filter size. The false-positive rate of the folded filter follows the standard Bloom filter formula at the reduced size, $p' \approx (1 - e^{-k \cdot n/m'})^k$, which monotonically increases as m' decreases. For filters with moderate saturation (below 50%), the increase in false-positive rate is modest. We validate this empirically: the privacy pipeline's detection performance (Section 6.5) shows that the approximation is sufficient for screening.

Differential privacy introduces a bias to the reported score that is proportional to $1/\epsilon$: at $\epsilon = 10.0$, the added noise is negligible and has minimal impact on decision accuracy; at $\epsilon = 1.0$, individual scores are noticeably perturbed but aggregate decisions across multiple comparisons remain meaningful; at $\epsilon = 0.1$ (a strong privacy regime), individual score perturbation is significant, making the system more suitable for screening rather than precise similarity quantification. Section 6.5 quantifies these trade-offs empirically across the full range of ϵ values and signature lengths.

5. Experimental Evaluation

5.1. Datasets

5.1.1. Reference Datasets

We evaluate the proposed similarity-detection pipeline using as a reference eight real-world datasets provided by the PISTIS consortium in their native format as semicolon-delimited CSV files. These datasets span smart city data hubs across three application domains. The data hubs represent typical IoT-intensive smart city verticals where multiple stakeholders share operational data under contractual terms.

The Mobility Data Hub contributes flight operations data from Athens International Airport (AIA) in Greece (arrival and departure passenger information, flight scheduling, baggage reconciliation records); transportation datasets from the Athens public transit authority (e.g., aggregated metro ticket validations); geospatial datasets from municipal authorities (business registry, administrative district boundaries, street network data); and meteorological datasets.

The Energy Data Hub contributes datasets related to grid resilience and flexibility markets, including consumption forecasts, generation profiles, flexibility market data, and contextual weather information. Data is exchanged among energy distribution system operators, market operators, and aggregators to support coordination of distributed energy resources in both long- and short-term flexibility markets.

The Automotive Data Hub contributes connected vehicle data, including trip recordings and sensor measurements collected from smartphone applications and vehicle sensors. These datasets capture driving patterns in relation to temporal, spatial, and environmental conditions (e.g., weather and air quality), and support urban analytics such as emissions modeling, traffic quality assessment, and driving behavior classification.

5.1.2. Synthetic Datasets

To emulate threat scenarios where data providers may attempt to republish partially modified or enriched versions of existing datasets, we apply a controlled augmentation process to each reference dataset following the threat taxonomy defined in Section 3. This covers all threat types at varying severity levels. For each reference dataset, we produce the following synthetic datasets: exact copies (T1); field-reordered versions where the column or field ordering is shuffled (T2); datasets with renamed fields using plausible alternative names (T3a) and datasets with injected spurious fields containing random data (T3b); variants with mild value perturbation such as rounding to fewer decimal places (T4 mild) and aggressive perturbation with 10–20% Gaussian noise added to numerical fields (T4 aggressive); partial extractions retaining 50% and 30% of the original records (T5); semantically obfuscated variants where field names and categorical values are replaced with dictionary-based synonyms (T6); and composite attacks that combine multiple modification types simultaneously (T7).

In the augmentation process for T6, we use a curated synonym dictionary covering 20 common field names and categorical values across the evaluation domains. While this represents a rudimentary level of adversarial sophistication, it serves as a baseline for validating the system's robustness in detecting similarity despite schema-level lexical changes.

5.1.3. Split Datasets

In addition to the full-size synthetic datasets, we consider the case of *partial reuse*, where only a part of the original dataset is used. For this "split dataset" scenario, we consider a subset of the AIA data described in Section 5.1.1 comprising: (i) a dataset of flight turnaround records (121K rows of 188 columns each) covering arrivals, departures, scheduling, and ground handling for a full year; and (ii) four baggage handling datasets (arrival, departure, transfer, and breakdown records; 12–54K rows of 14–21 columns each).

For each of the five datasets, we created multiple variants through random row sampling and temporal windowing (splitting by date into non-overlapping time periods), yielding 35 more derivative datasets that undergo the same T1–T7 augmentation process. We further constructed three categories of real-data evaluation pairs: *temporal-split pairs* (adjacent time windows from the same source), *cross-table pairs* (data from different source tables), and *real-vs-synthetic pairs* (real data paired with synthetic ones).

The cross-table pairs comprise two source types: real and synthetic. The former ones originate from the five AIA datasets. These five sources yield ten pair combinations. We sample five distinct (variant, variant) pairings from each of the combinations, mixing random-subset and temporal-window variants on each side. This approach results in 50 cross-table hard negatives. The latter ones pair two different synthetic table types belonging to the *same* application domain (e.g., flight-operations vs. public-transport ticket validations, both within the Mobility Hub). The five structured synthetic table-pair com-

binations across the three domains each contribute 10 distinct variant pairings, drawn from the cross product of available base variants. This contributes some 50 additional cross-table hard negatives. The count per type (50) supports a more demanding evaluation of the system's robustness against domain-related false positives. It was chosen as the smallest sample size at which the Wilson 95% confidence intervals (defined in [32]) on the per-category false-positive rate are statistically disjoint between the Inspector and every text-similarity baseline.

5.1.4. Evaluation Corpus

The resulting evaluation corpus comprises 966 dataset pairs in total: 765 similar pairs, comprising 750 from augmented datasets covering T1–T7 and 15 temporal-split pairs from real airport data, and 201 dissimilar control pairs: 66 cross-domain, 100 cross-table, equally split between the two types, and 35 real-vs-synthetic.

5.2. Experimental Setup

All experiments were conducted on a computer equipped with an Intel Core i9-10885H CPU (8 cores, 16 threads, 2.4 GHz base / 5.3 GHz boost), 128 GB DDR4 RAM, and integrated Intel UHD Graphics (no dedicated GPU). The benchmarking is implemented in Python 3.12 using `PyTorch` 2.10 for embedding computation (CPU-only inference for SBERT), `scikit-learn` 1.8 for evaluation metrics, and standard libraries for hashing and Bloom filter operations. SBERT embeddings are computed once per dataset and cached. All reported timings reflect warm-cache conditions, i.e., the first embedding computation is excluded from per-pair timing.

The IDF weights for the weighted Jaccard component are computed over all base (reference) datasets in the evaluation corpus and held fixed before any pairwise comparisons, matching the intended deployment model where IDF statistics are derived from the full catalog of registered datasets. Per-pair timings are measured as single-threaded execution; in a production deployment with parallel workers, throughput would scale approximately linearly with the number of workers. All experiments use a batch size of one, i.e., one dataset pair per comparison call, to reflect the expected online-submission use case.

5.3. Evaluation Methodology

The evaluation methodology is designed to provide a rigorous and fair comparison across all methods. Adaptive thresholds are learned via five-fold stratified *group* cross-validation, where the group key per pair is the underlying base dataset shared by its two sides (i.e., the same base dataset stripped of any augmentation suffix). This guarantees that augmented variants of a single base dataset never straddle the train/test split: no base dataset can appear in more than one fold. The 966-pair evaluation corpus contains 291 distinct group keys, well above the five folds, and each fold maintains the same ratio of similar-to-dissimilar pairs as the overall corpus through stratification; the two splitting schemes recover nearly-identical (α^*, τ^*) profiles, providing direct empirical evidence that the original procedure was not optimistic.

We report seven standard classification metrics: precision (P), which measures the fraction of flagged pairs that are truly similar; recall (R), which measures the fraction of truly similar pairs that are correctly flagged; F_1-score, the harmonic mean of precision and recall; specificity, the fraction of truly dissimilar pairs correctly rejected; balanced accuracy, the arithmetic mean of recall and specificity; Matthews Correlation Coefficient (MCC) [33], which summarizes the full confusion matrix into a single value bounded in $[-1, 1]$; and Area Under Curve (AUC)-ROC, which measures overall discriminative ability across all possible thresholds. Optimal classification thresholds are selected by maximizing Youden's index J on the training folds and then applied without modification to the held-out fold,

preventing threshold overfitting. For baseline comparisons, we report results at the default threshold ($\tau = 0.5$), which provides a uniform comparison point, and discuss optimized thresholds in the text.

The evaluation corpus is class-imbalanced (79.2% positive pairs), which may inflate the F_1-score for methods with high recall. A trivial classifier that labels all pairs as similar would achieve F_1-score of 0.884. To control for this inflation, we additionally report specificity, balanced accuracy, and MCC, all of which are insensitive to the positive/negative ratio: a trivial all-positive classifier obtains MCC of zero and specificity of zero, so a similarity-detection pipeline that outperforms baselines on these metrics is doing so on the basis of actual discriminative power rather than corpus skew. We also explicitly report the absolute false-positive count, since in a contract-enforcement deployment, each false positive triggers a dispute or investigation and is therefore operationally meaningful.

5.4. Baseline Methods

We compare our pipeline against six baseline methods that represent the primary alternative approaches for dataset comparison. SHA-256 method computes a cryptographic hash of the entire dataset content and declares a match only when the hashes are identical, representing the standard integrity verification approach used in most data platforms today. MinHash-128 and MinHash-256 compute MinHash signatures of length 128 and 256, respectively, over the tokenized dataset content, providing approximate Jaccard similarity estimation with different accuracy–efficiency trade-offs. TF-IDF cosine treats each dataset as a bag of tokens, computes Term Frequency–Inverse Document Frequency weights, and measures similarity via cosine distance between the resulting sparse vectors. BM25 applies the Okapi BM25 relevance scoring function, which extends TF-IDF with document length normalization and term frequency saturation, treating one dataset as the query and the other as the document. Finally, pure SBERT (`all-MiniLM-L6-v2`) encodes the textual representation of each dataset into a dense embedding vector and computes cosine similarity, representing the state of the art in neural text similarity. All baseline methods are applied on the same dataset pairs, ensuring a fair comparison under identical conditions.

6. Results and Discussion

6.1. Overall Performance

Table 3 summarizes the performance of all methods at the default threshold $\tau = 0.5$ for the whole evaluation corpus described in Section 5.1. Our core pipeline (Inspector) achieves the highest F_1-score 0.992 at the default threshold, with near-perfect precision (0.987) and recall (0.996). The class-imbalance-robust headline numbers for both specificity and MCC are substantially above the closest baseline, namely BM25. Against the trivial all-positive classifier, which would obtain MCC = 0 and specificity = 0, the Inspector exhibits genuine discriminative power rather than artifactual F_1 inflation. The Inspector produces only 10 false positives out of 201 negative pairs, outperforming BM25 ($F_1 = 0.965$, 55 FP), SBERT ($F_1 = 0.903$, 165 FP), and TF-IDF ($F_1 = 0.888$, 193 FP). Its advantage stems almost entirely from precision on hard negatives: while all baselines achieve perfect or near-perfect recall, they suffer from elevated false-positive rates on domain-related dissimilar pairs, as discussed in Section 6.2. SHA-256 detects only exact copies (T1); it achieves perfect precision but exhibits low recall (0.222). MinHash variants also achieve perfect precision but limited recall (resp. 0.567 and 0.563). Due to token-level hashing without schema-aware comparison, they produce near-zero scores for temporal-split pairs, although the structural similarity provides some useful signal.

Table 3. Overall performance at threshold $\tau = 0.5$ on the full 966-pair evaluation corpus.

Method	Precision	Recall	F1	Spec.	MCC	FP	AUC-ROC	Time (s/pair)
SHA-256	1.000	0.222	0.364	1.000	0.237	0	0.611	0.001
MinHash-128	1.000	0.567	0.724	1.000	0.463	0	0.937	0.061
MinHash-256	1.000	0.563	0.721	1.000	0.460	0	0.943	0.120
TF-IDF Cosine	0.799	1.000	0.888	0.040	0.178	193	1.000	0.005
BM25	0.933	1.000	0.965	0.726	0.823	55	0.976	0.007
SBERT	0.823	1.000	0.903	0.179	0.384	165	0.999	0.054
Inspector (base)	0.987	0.996	0.992	0.950	0.959	10	0.997	0.178
Inspector (privacy)	0.862	0.529	0.656	0.677	0.167	65	0.608	0.066

6.2. Per-Threat Similarity Score Breakdown

Figure 2 depicts the average similarity scores per method broken down by threat type. Inspector, our proposed pipeline achieves perfect or near-perfect scores (0.94–1.00) across all augmentation-based threat types T1–T7. The lowest average score is observed again for T4 aggressive, confirming that heavy value perturbation is the most challenging single-threat scenario for structural and value-based comparison. The perfect T6 (synonym replacement) score demonstrates that the field-aware value comparison layer described in Section 4.4 successfully identifies semantically-equivalent substitutions.

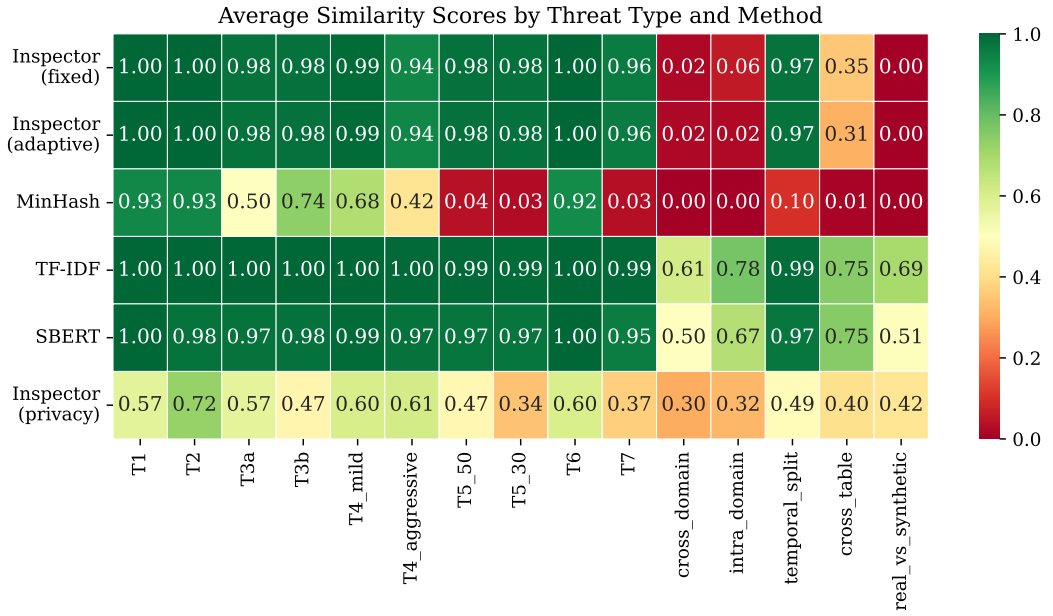


Figure 2. Average similarity scores by threat type and method.

Cross-domain pairs score 0.01 on average while real-vs-synthetic pairs score 0.00, correctly identified as dissimilar with no exceptions. Intra-domain pairs (e.g., synthetic flight operations vs. public transport tables from the same Mobility domain) produce a mean score of 0.06, demonstrating near-perfect separation. Temporal-split pairs achieve an average score of 0.97 with perfect detection ($F_1 = 1.0$), confirming that our pipeline correctly identifies those datasets sharing the same schema even when drawn from non-overlapping time periods. Cross-table pairs (different baggage/flight tables from the same Mobility domain) produce a mean score of 0.35, well below the detection threshold, despite sharing domain-specific vocabulary (airport codes, flight identifiers, handler names). For comparison, TF-IDF, SBERT, and BM25 assign these same pairs a mean score of 0.75, 0.74, and 0.63, respectively—all above the default threshold, resulting in false positives for every baseline. The Inspector's separation on these hard negatives is analyzed in Section 6.6.

6.3. Synthetic vs. Reference Data

To assess whether the pipeline generalizes from synthetic to real-world data, Figure 3 compares the score distributions for synthetic and real dataset pairs. For similar pairs (left panel), both distributions tightly cluster near 1.0, with real data showing a slightly wider spread due to the natural noise, missing values, and schema irregularities present in actual airport data. For dissimilar pairs (right panel), the IDF-weighted Jaccard mechanism ensures that both synthetic and real dissimilar pairs produce low scores, with cross-domain pairs averaging 0.01 and real-vs-synthetic pairs averaging 0.00. Real dissimilar pairs exhibit higher scores due to shared domain vocabulary (hard negatives). The cross-table pairs (mean 0.35) are the hardest negatives but remain below the default threshold of $\tau = 0.5$, confirming that the proposed pipeline generalizes from synthetic to real data.

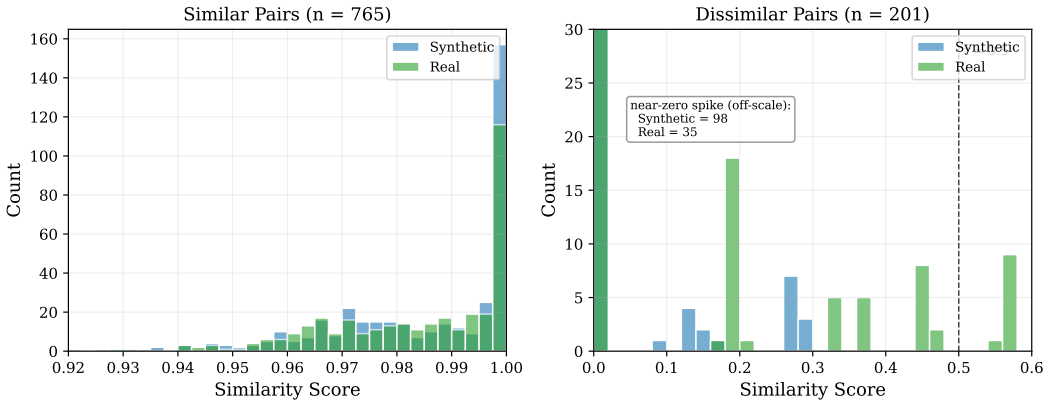


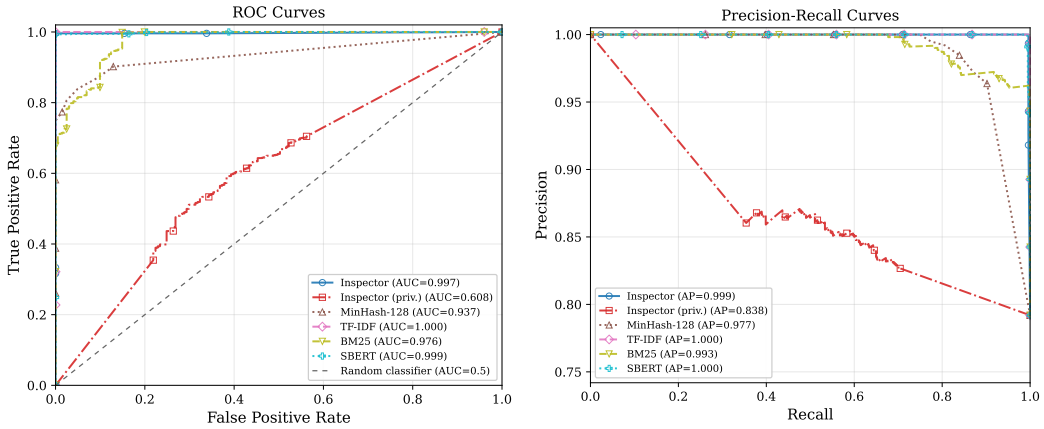
Figure 3. Score distributions for synthetic and real similar (**left**) and dissimilar (**right**) dataset pairs.

Only three similar pairs out of the 765 ones were missed by our Inspector—all involving T4 aggressive perturbation, where 50% of numerical fields were altered with 10% Gaussian noise, representing the most extreme modification in our corpus. Our proposed pipeline produced 10 false positives out of 201 negatives, all of them on cross-table (real) hard negatives where different airport tables share domain vocabulary. The Inspector's cross-table (real) false-positive rate is 10/50 (20.0%) with a 95% Wilson confidence interval of [11.2%, 33.0%]; on the other three dissimilar-pair categories (cross-table (synthetic), cross-domain, and real-vs-synthetic), the Inspector commits zero false positives. We address the cross-table (real) 20% rate explicitly in Section 6.6 as the residual failure mode of the pipeline and discuss its mitigation through threshold adjustment.

In contrast, TF-IDF produced 193 false positives out of 201, flagging as similar essentially every dissimilar pair (cross-table (real) FPR 100%, cross-table (synthetic) FPR 100%). The TF-IDF bag-of-tokens representation inflates similarity scores whenever datasets share domain vocabulary. Similarly, SBERT produced 165 false positives, including all cross-table (real and synthetic) pairs (cross-table (real) FPR 100%). BM25 proved more selective (55 false positives overall, cross-table (real) FPR 74%) but still misclassified the majority of cross-table (real) pairs at the default threshold. Importantly, on the cross-table (real) category, the four methods' 95% Wilson confidence intervals for false-positive rate are pairwise disjoint—Inspector [11.2%, 33.0%] vs. BM25 [60.4%, 84.1%] vs. SBERT and TF-IDF both [92.9%, 100%]—which establishes the Inspector's dominance on hard negatives at statistical significance at the chosen $n = 50$ per category.

Figure 4 depicts the ROC and Precision-Recall curves for all evaluated methods. TF-IDF and SBERT achieve perfect AUC-ROC (1.000), indicating that their score distributions are separable with sufficient threshold tuning (TF-IDF requires $\tau = 0.93$ and SBERT $\tau = 0.87$). However, such aggressive thresholds are impractical without labeled calibration data, highlighting the robustness of the proposed pipeline at the default threshold. Min-

Hash variants show lower overall discriminative power (AUC-ROC ≈ 0.95), due to their binary token-matching nature.



(**a**) ROC curves. (**b**) Precision-Recall curves.

Figure 4. ROC (**a**) and Precision-Recall (**b**) curves for evaluated methods.

6.4. Adaptive Thresholds

The adaptive threshold optimization presented in Section 4.8 learns per-modality fusion weights and decision thresholds via cross-validated grid search. Table 4 presents the learned profiles for each modality in our evaluation corpus, together with the F_1 of the fixed configuration ($\alpha = 0.5, \tau = 0.5$), the held-out F_1 obtained by applying the learned (α^*, τ^*) to out-of-fold predictions on the same modality slice for reference, and the spread of the per-fold Youden-optimal threshold across the five cross-validation folds. The tiny spread (at most 1.5×10^{-4}) confirms that τ^* is a stable property of the score distribution rather than a knife-edge calibration choice. The proposed adaptive scheme applies a different (α^*, τ^*) per modality, unlike the baseline methods of Section 5.4, which apply a single global threshold to the entire corpus. As such, figures in Table 4 are not directly comparable to the ones in Table 3.

We emphasize that on this evaluation corpus, the adaptive scheme does not improve corpus-wide F_1 over the fixed defaults. Rather, its contribution is the per-modality calibration with documented per-fold stability, as described in the next paragraphs. The negative ΔF_1 entries in Table 4 reflect this trade-off explicitly: the adaptive scheme trades a small F_1 reduction for the empirical calibration stability documented by the per-fold τ^* range, which is at most 1.5×10^{-4} across all three modalities—an operating point that is data-driven rather than fixed at the conventional default.

Table 4. Learned adaptive profiles per modality, after dataset-level (group) cross-validation.

Modality	F1 Fixed	α^*	τ^*	F1 Adaptive	AUC-ROC	ΔF_1	Per-Fold τ^* Range
Structured JSON	1.000	0.95	0.0325	0.980	1.000	−0.020	[0.03247, 0.03258]
Structured CSV	0.986	0.90	0.0678	0.951	0.994	−0.035	[0.06768, 0.06807]
Unstructured text	1.000	0.00	1.0000	0.958	0.910	−0.042	[1.0000, 1.0000]

For structured JSON datasets, the optimizer selects $\alpha^* = 0.95$, assigning a very high weight to IDF-weighted structural overlap. This reflects the fact that with IDF weighting, the path-level comparison becomes highly discriminative: nested JSON field hierarchies with rare path combinations receive high IDF weights, making the structural similarity strongly indicative of genuine relatedness. The corresponding threshold $\tau^* = 0.0325$ is very low and is recovered within $\pm 6 \times 10^{-5}$ across the five cross-validation (CV) folds (Table 4). This is explained by the score distribution for JSON pairs exhibited in Figure 5: dissimilar

cross-domain pairs produce near-zero IDF-weighted Jaccard scores (since they share few paths, and any shared paths receive low IDF weights), while even heavily modified positive pairs (e.g., T3 with renamed fields, or T7 composites) retain sufficient structural similarity to exceed this low threshold. The AUC-ROC of 1.000 confirms perfect separability across all operating points.

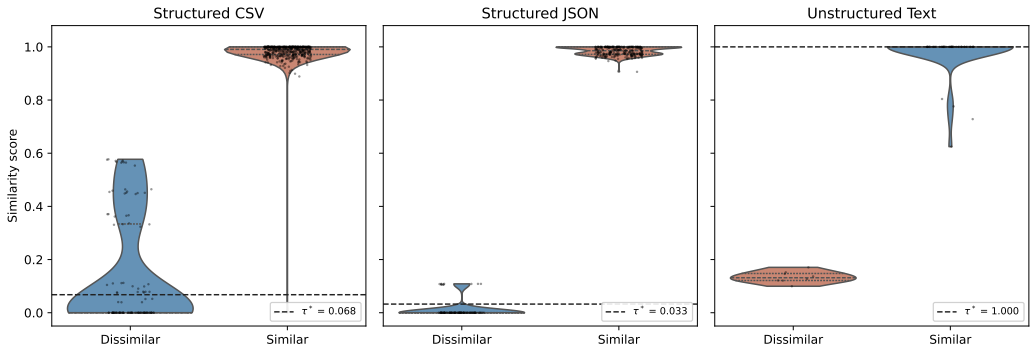


Figure 5. Per-modality score distributions of the adaptive pipeline under group cross-validation. The dashed line marks the learned τ^*.

For CSV data, the optimizer selects $\alpha^* = 0.90$, assigning most of the weight to IDF-weighted structural overlap. With IDF weighting, column headers provide a strong discriminative signal: common domain-specific headers shared across many datasets (e.g., "date", "id") are down-weighted, while distinctive column combinations that appear in only a few datasets receive high IDF weights. A CSV pair from two distinct airport tables shares many surface-level tokens but few rare path–value combinations, so the optimizer correctly increased the structural weight to suppress the resulting false positives. The threshold $\tau^* = 0.0678$ is recovered within $\pm 4 \times 10^{-4}$ across the five CV folds.

For unstructured text, the optimizer selects $\alpha^* = 0.0$, meaning the system solely relies on embedding-based similarity. The threshold $\tau^* = 1.000$ requires particular explanation. In this modality, the similarity score is the raw SBERT cosine similarity, which operates in the range $[-1, 1]$ but in practice produces values in $[0.3, 1.0]$ for our corpus. The positive pairs (copies with synonym substitution or mild paraphrasing) produce cosine similarities very close to 1.0 (typically greater than 0.98), because the SBERT model used (`all-MiniLM-L6-v2`) produces highly similar embeddings for semantically equivalent text, even after dictionary-based synonym substitution. The dissimilar pairs produce lower but still non-trivial cosine scores between 0.4 and 0.7, as application descriptions share general vocabulary. All five CV folds independently selected $\tau^* = 1.0000$ to four decimal places after the boundary-candidate exclusion described in Section 4.8, confirming that this value is the genuine Youden-optimum on the evaluation corpus rather than an artifact of the threshold-search procedure.

All structured modalities achieve high AUC-ROC, confirming that the similarity scores provide strong separability between similar and dissimilar pairs once the fusion weight is properly tuned. The evaluation includes both cross-domain negatives (e.g., Mobility vs. Energy) and same-domain cross-table negatives (different tables from the same application domain, whether sourced from the real airport catalog or from the synthetic generators), with the IDF-weighted Jaccard and schema-evidence-gated fusion mechanisms specifically designed to handle this harder same-domain case. The unstructured text modality reaches a lower AUC-ROC, which is expected, given the inherent difficulty of distinguishing semantically related but independently authored text documents from genuine copies.

6.4.1. Fixed and Adaptive Configuration Relationship

At the global default $\tau = 0.5$, the fixed and adaptive configurations differ only in the value of the fusion weight α: fixed applies $\alpha = 0.5$ uniformly, whereas adaptive applies the per-modality ($\alpha^* = 0.95$ for JSON, 0.90 for CSV, 0.00 for text). The two configurations therefore produce *different per-pair scores*, but on this corpus, the score shifts induced by α never cross the threshold $\tau = 0.5$: every pair classified as similar by fixed is also classified as similar by adaptive, and vice versa, so the two configurations are distinguishable at the score level yet indistinguishable at the prediction level when both are evaluated at $\tau = 0.5$. The reason is structural: the IDF-weighted Jaccard combined with schema-evidence-gated fusion induces a strongly bimodal score distribution, as depicted in Figure 5, with positive pairs clustered between 0.95 and 1.00 and dissimilar pairs below 0.35, leaving an empty interval of width approximately 0.60 that contains $\tau = 0.5$. Because the fused score is a convex combination of two bounded non-negative components, varying α shifts any individual score by at most $|S^{\text{IDF}}_{\text{Jaccard}} - S_{\text{values}}|$, empirically, a few hundredths in either direction. We verified by sweeping α across $[0, 1]$ in steps of 0.05 that no pair changes its predicted class at $\tau = 0.5$ under any choice of α. Consequently, we do not claim that the adaptive scheme improves classification accuracy over the fixed defaults on the evaluation corpus.

The contribution of the adaptive scheme is the per-modality calibration rather than a classification gain: the learned τ^* defines the actual location of the score-distribution gap under each modality's specific score statistics, sitting at the lower edge of the empty interval (closer to the dissimilar cluster). This admits a small number of marginal false positives and slightly lowers held-out F_1 (-0.020 for JSON, -0.035 for CSV, and -0.042 for text), but the fold-level stability in Table 4 indicates that the calibration is a property of the data rather than of any individual training sample. We verified the robustness of each τ^* to small perturbations by recomputing held-out F_1 at $\tau^* \pm 0.05$ and $\tau^* \pm 0.10$: F_1 changes by less than 0.05 over a ± 0.10 window for every modality, confirming that τ^* is a gap-located threshold rather than a knife-edge choice. In a deployment whose score distribution lacks such a wide empty interval, e.g., a catalog containing many domain-related but distinct datasets that compress the gap, the fixed default $\tau = 0.5$ would no longer be safely within the gap, and the adaptive profiles would be required to recover the correct operating point.

The operational value of this stability is twofold. First, it gives the deploying operator confidence that the learned τ^* reflects a genuine property of the score distribution rather than the particular labeled sample used for training: a small labeled set therefore suffices, and modest changes in that set do not move the operating point. Second, it supports the recalibration story of Section 4.8: when a catalog evolves and the optimizer is re-run on the new data, the same kind of stable answer is expected on the refreshed distribution, which is what makes drift signals on a held-out negative set interpretable in the first place. The negative ΔF_1 entries reported in Table 4 are therefore a corpus-specific artifact of the wide empty interval in this airport catalog rather than a general property of the adaptive scheme. In a deployment with a narrower or absent gap—e.g., a catalog dominated by near-duplicate datasets that pull the dissimilar cluster up toward $\tau = 0.5$, or one in which LLM-driven paraphrasing lowers positive-pair scores below the $[0.95, 1.00]$ band observed here—the conventional default $\tau = 0.5$ would lie inside (or even past) the dissimilar cluster, producing false positives that the adaptive scheme avoids by selecting a higher per-modality τ^*; under such conditions the same per-fold stability documented in Table 4 would correspond to *positive* ΔF_1 entries and a genuine classification gain rather than a calibration-only benefit. The empirical stability observed here is therefore the prerequisite that allows the mechanism to deliver in those harder conditions, even when the present corpus does not make that benefit visible.

6.4.2. Deployment and Recalibration in Evolving Data Spaces

To instantiate the adaptive scheme in a new data space, an operator follows a four-step procedure: (i) index the catalog and freeze the IDF background weights over the resulting corpus; (ii) construct a labeled training set, either by applying the threat-model augmentations of Section 3 to existing catalog entries (greenfield bootstrap) or, where available, by drawing on historical confirmed-violation and confirmed-clean records (operational bootstrap); (iii) run the grid-search procedure of Section 4.8 to obtain per-modality (α^*, τ^*) profiles; and (iv) recalibrate as the deployment evolves. The per-modality stratification is intentionally narrow: the three modality buckets (JSON, CSV, text) cover the data types prevalent in current data spaces, whereas stratifying further (e.g., per-modality and per-domain) would require labeling effort per (modality, domain) combination and limit portability across deployments.

The profiles of Table 4 capture the score distribution of the training catalog and are therefore tied to that snapshot rather than to the system itself, so they should be refreshed when this distribution shifts. Three triggers warrant recalibration: changes in catalog composition or scale, which alter the IDF statistics of Equation (3); the appearance of previously-unseen adversarial transformations, which shifts the positive-pair score distribution; and drift in the empirical false-positive rate on a held-out negative set, which an operator can monitor as a safety signal.

Recomputation is inexpensive, since the grid search operates on cached component scores, which makes periodic refresh practical; however, determining an empirically-optimal recalibration cadence would require longitudinal observations across a live deployment and is left as future work (Section 7). The resulting numerical profiles are specific to the catalog and threat distribution, whereas the MinHash and Bloom-filter parameters of the privacy layer (Section 4.9) are catalog-independent and need re-tuning only if the signature-length or false-positive-rate policy itself changes.

6.5. Privacy-Preserving Mode

The privacy-preserving pipeline operates on compact fingerprints (MinHash signatures and Bloom filters) rather than raw data, with calibrated Laplace noise added to the reported scores. At $\tau = 0.5$, the privacy-preserving pipeline achieves $F_1 = 0.656$ with precision 0.862 and recall 0.529, a trade-off relative to the full pipeline ($F_1 = 0.992$), as summarized in Table 3, while running noticeably faster (0.066 s/pair vs. 0.178 s/pair on the revised setup), making it suitable as a first-stage screening mechanism in deployments where latency or computational resources are constrained.

Table 5 presents the impact of the differential privacy parameter ϵ on detection accuracy. As expected, detection performance improves monotonically with increasing ϵ (weaker privacy, less noise): at $\epsilon = 0.1$ (strong privacy), F_1 is 0.613 due to severe score perturbation, while at $\epsilon = 10.0$ (relaxed privacy), F_1 reaches 0.846 with 0.939 precision. The sharpest improvement occurs between $\epsilon = 1.0$ and $\epsilon = 5.0$, suggesting that $2 \leq \epsilon \leq 5$ represents a practical operating range that balances privacy and utility for cooperative auditing scenarios.

Table 5. Privacy-preserving detection performance across differential privacy budgets.

ϵ	τ^*	Precision	Recall	F1	AUC-ROC
0.1	0.33	0.790	0.501	0.613	0.490
0.5	0.30	0.811	0.600	0.690	0.546
1.0	0.30	0.859	0.637	0.731	0.641
2.0	0.30	0.888	0.703	0.785	0.721
5.0	0.31	0.910	0.762	0.829	0.802
10.0	0.30	0.939	0.770	0.846	0.872

Figure 6 depicts the effect of MinHash signature length on detection accuracy with differential privacy (DP) noise effectively disabled ($\epsilon = 100$). Across signature lengths from 32 to 512, F_1 remains stable at approximately 0.858, indicating that even short signatures (32 hashes) provide sufficient accuracy for the similarity estimation task. This stability is explained by the binary nature of the detection decision: for most pairs, the MinHash similarity estimate is either clearly above or clearly below the threshold, and the estimation variance introduced by shorter signatures does not cross the decision boundary. We suggest a signature length of 128 as a practical default, balancing computational cost with robustness against edge-case pairs whose similarity falls near the threshold.

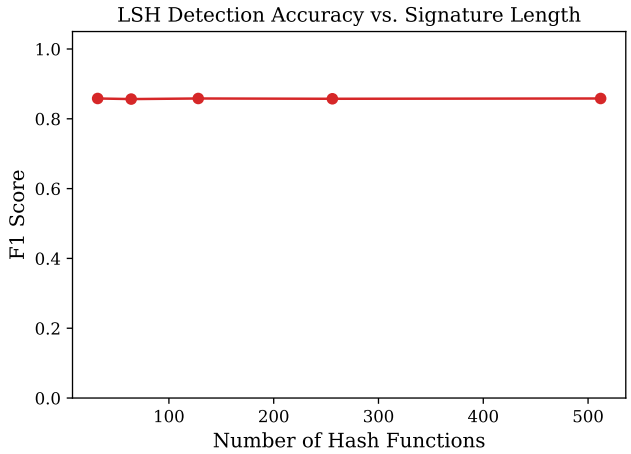


Figure 6. Similarity detection score as a function of MinHash signature length (negligible DP noise).

6.6. Implications for Data Space Governance

As smart city ecosystems increasingly rely on cross-organizational data sharing—from IoT-generated mobility data to energy grid telemetry—the ability to detect unauthorized dataset reuse becomes a critical building block for resilient and trustworthy data governance. The presented results show that a multi-layer similarity-detection pipeline is feasible and effective for identifying dataset reuse in federated data spaces.

The pipeline achieves $F_1 = 0.992$, MCC = 0.959, and 10 false positives out of 201 dissimilar pairs at $\tau = 0.5$, reducing baseline false-positive counts by a factor of at least five; the privacy-preserving variant retains $F_1 = 0.846$ at $\epsilon = 10.0$ without accessing raw data, supporting a two-tier deployment in which screening precedes full-pipeline analysis.

On the cross-table (real) hard-negative category, the Inspector flags 20% of pairs as similar against $\geq$74% for every text-similarity baseline (Section 6.3), confirming that the IDF-weighted Jaccard suppresses but does not eliminate domain-vocabulary false positives.

All 10 of the Inspector's false positives fall on cross-table (real) pairs; raising the operating threshold to $\tau = 0.58$ drives this rate to zero on the present corpus while preserving recall on every augmentation-based threat category.

6.7. Threats to Validity

We identify the following three classes of threats to validity for the presented work and evaluation results:

- Internal validity. To the best of our knowledge, there are no real-world adversarial datasets available in the public. Due to this, we resort to controlled augmentations for the evaluation. The augmentation parameters (noise scales, extraction fractions, renaming rates) were chosen to reflect a sensible range of adversarial effort. However, this may not capture the full spectrum of malicious dataset modifications in the

future. The inclusion of real operational data mitigates the risk of evaluating only on artificially clean datasets.

- External validity. The numerical (α^*, τ^*) profiles reported in Table 4 are properties of the present evaluation catalog and envisioned data reuse forms. We do not claim that exactly these values apply elsewhere. What is intended to generalize is the *mechanism*: the modality taxonomy (JSON, CSV, text), the IDF-weighted Jaccard component over a catalog-wide background corpus, the grid-search-plus-Youden optimization, and the per-modality routing at inference time. Section 4.8 describes the four-step bootstrap pipeline by which an operator instantiates these components in a new data space alongside three recalibration triggers.
- Construct validity. The binary similar/dissimilar labeling simplifies the nuanced question of "how much reuse constitutes a violation?" into a detection problem. The continuous similarity score and field-level reports provide richer information for human decision-makers to assess contract violations. Data similarity alone, even at the level achieved by the proposed pipeline, is evidence for rather than proof of data reuse: high similarity can arise from common standards, public schemata, repeated templates, or independently produced domain data. The IDF-weighted Jaccard component of Section 4.3 is explicitly motivated by the need to discount paths that recur across the catalog precisely because they reflect such common templates rather than reuse, and the field-level reports of Section 4.2 are the artifacts on which a human decision-maker bases a violation finding.

7. Conclusions and Future Work

This paper presented a modular similarity-detection pipeline for detecting unauthorized reuse, redistribution, or modification of licensed datasets in federated data spaces based on three pillars. First, an IDF-weighted structural similarity mechanism with schema-evidence-gated fusion that discounts common domain vocabulary and prevents embedding-based false positives on intra-domain negatives. Second, an adaptive threshold optimization mechanism that replaces fixed fusion parameters with learned, modality-specific weights and decision thresholds via cross-validated grid search. Third, a privacy-preserving similarity layer based on MinHash LSH signatures, Bloom filters with OR folding alignment, and calibrated Laplace noise, which enables cross-organizational dataset comparison without exposing raw data.

Three lessons emerge from this evaluation. First, the IDF-weighted Jaccard combined with schema-evidence-gated fusion produces a strongly bimodal similarity-score distribution on multi-domain catalogs, which translates directly into robust separation between genuinely reused and merely domain-related datasets—the property that distinguishes the Inspector from text-similarity baselines on hard negatives. Second, the residual failure mode of the pipeline is concentrated on cross-table (real) sibling pairs from the same operational system; it is statistically characterized rather than asymptotically eliminated by structural features alone, and the appropriate engineering response is a slightly raised operating threshold rather than further fusion-weight tuning. Third, the per-modality adaptive scheme contributes deployment-time calibration rather than a corpus-wide accuracy gain: its value emerges when the score-distribution gap is narrower than in the present catalog, which we identify as a direction for future deployment validation.

Several directions for future work emerge here. One direction involves the validation of detection capabilities in other federated spaces for the same (mobility, energy, and automotive) or different application domains. This may include additional modalities, not currently handled by the pipeline. Of particular interest would be a generalization of the adaptive threshold optimization for different domains. Also, the introduction of a

more granular adaptive threshold strategy that considers the threat type as well, enabling threat-specific detection profiles.

A second direction involves the robustness of the pipeline against complex LLM-generated adversarial transformations, including deep cross-lingual paraphrasing and complex schema restructuring to degrade matching performance. For example, one can use an LLM to translate all field names of a CSV file from (say) Greek to English, then paraphrase in English (e.g., rename "personnel" to "staff members" or "employees"), and finally translate the paraphrased field names back to Greek. While semantically equivalent, the retranslated field names would be heavily altered, obfuscating the schema alignment ahead of the value similarity computation. Given the modular architecture of the pipeline, it should be possible to adapt and extend the detection layers so as to consider additional capabilities.

A third direction would be towards optimizing the evaluation of large-scale catalogs with thousands of datasets. Assuming dataset availability, two possible paths are (i) the validation of a two-tier screening approach using the privacy-preserving fingerprint filtering followed by full dataset analysis, and (ii) exploring trust trade-offs to utilize dataset metadata and reduce the volume of comparisons.

Towards expansion, a fourth direction involves the integration of the similarity reports with automated policy engines that can map detected overlap to specific contract clauses and initiate enforcement workflows in PISTIS-enabled data spaces.

The proposed modular similarity-detection pipeline is part of the open source PISTIS framework. As such, it remains available for all interested researchers and parties to extend, experiment with, and integrate novel contributions in the future.

Author Contributions: Conceptualization, C.P. and K.S.; methodology, C.P., A.G.V. and K.S.; software, C.P.; validation, C.P. and A.G.V.; formal analysis, C.P. and A.G.V.; investigation, C.P., A.G.V. and K.S.; resources, K.S.; data curation, C.P.; writing—original draft preparation, C.P. and K.S.; writing—review and editing, C.P., A.G.V. and K.S.; visualization, C.P.; supervision, K.S.; project administration, K.S.; funding acquisition, K.S. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported by the PISTIS project (GA: 101093016), funded by the European Commission under the Horizon Europe programme.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: All the relevant implementations are publicly available in the PISTIS project GitHub repository https://github.com/PISTIS-Platform (accessed on 7 June 2026).

Acknowledgments: We thank the PISTIS project partners for providing the reference datasets used in this study as well as for their contributions to the overall PISTIS platform implementation.

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

References

1. European Commission. Regulation (EU) 2022/868 of the European Parliament and of the Council on European Data Governance (Data Governance Act). *Off. J. Eur. Union* **2022**, *OJ L152*, 1–44.
2. European Commission. Regulation (EU) 2023/2854 of the European Parliament and of the Council on Harmonised Rules on Fair Access to and Use of Data (Data Act). *Off. J. Eur. Union* **2023**, *OJ L*, 1–71.
3. Pettenpohl, H.; Spiekermann, M.; Both, J.R. International data spaces in a nutshell. In *Designing Data Spaces: The Ecosystem Approach to Competitive Advantage*; Springer: Berlin/Heidelberg, Germany, 2022; pp. 29–40.
4. Otto, B.; Steinbuß, S.; Teuscher, A.; Lohmann, S. *GAIA-X: Technical Architecture*; GAIA-X European Association for Data and Cloud AISBL: Brussels, Belgium, 2022.

5. Curry, E.; Scerri, S.; Tuikka, T. *Data Spaces: Design, Deployment and Future Directions*; Springer: Berlin/Heidelberg, Germany, 2022. [CrossRef]
6. Doukas, T.; Hadjipavlis, G.; Tsougiannis, E.; Fokeas, K.; Krikochoriti, M.; Antonakopoulou, A.; Sourlas, V.; Amditis, A. A Secure Approach on Data Trading: How the PISTIS Monetary Ledger Supports Digital Transactions. In *Proceedings of the 2025 6th International Conference in Electronic Engineering & Information Technology (EEITE)*; IEEE: Piscataway, NJ, USA, 2025; pp. 1–7.
7. Zheng, Z.; Xie, S.; Dai, H.N.; Chen, X.; Wang, H. Blockchain challenges and opportunities: A survey. *Int. J. Web Grid Serv.* **2018**, *14*, 352–375. [CrossRef]
8. Christidis, K.; Devetsikiotis, M. Blockchains and Smart Contracts for the Internet of Things. *IEEE Access* **2016**, *4*, 2292–2303. [CrossRef]
9. PISTIS Project Consortium. Pistis: Promoting and Incentivising Self-Sovereign Data Sharing Through Trust-Based Data Monetisation. 2023. Available online: https://cordis.europa.eu/project/id/101093016 (accessed on 28 May 2026).
10. Panagiotou, C.; Stefanidis, K. Battling Data Counterfeiting in Industrial Data Trading Environments. In *Proceedings of the 2024 IEEE 29th International Conference on Emerging Technologies and Factory Automation (ETFA)*; IEEE: Piscataway, NJ, USA, 2024; pp. 1–4.
11. Panagiotou, C.; Stefanidis, K. Detecting Dataset Reuse and Modification in Data Spaces via Structure-Aware Similarity Analysis. In *Proceedings of the 11th IEEE International Smart Cities Conference (ISC2)*; IEEE: Piscataway, NJ, USA, 2025.
12. MISP Project. MISP 2.4.140 Released (OpenID Support, Cross Object References in Extended Events and Many Improvements). 2021. Available online: https://www.misp-project.org/2021/03/10/MISP.2.4.140.released.html (accessed on 7 June 2026).
13. Giel, A.; Hupperz, M.; Schoormann, T.; Möller, F. What does it take to connect? Unveiling characteristics of data space connectors. In Proceedings of the 57th Annual Hawaii International Conference on System Sciences, Hawaii, HI, USA, 3–6 January 2024; pp. 4238–4247.
14. Broder, A.Z. On the resemblance and containment of documents. In Proceedings of the Compression and Complexity of Sequences, Salerno, Italy, 11–13 June 1997; pp. 21–29.
15. Broder, A.Z.; Charikar, M.; Frieze, A.M.; Mitzenmacher, M. Min-wise independent permutations. *J. Comput. Syst. Sci.* **2000**, *60*, 630–659. [CrossRef]
16. Manku, G.S.; Jain, A.; Das Sarma, A. Detecting near-duplicates for web crawling. In Proceedings of the 16th International Conference on World Wide Web, Banff, AB, Canada, 8–12 May 2007; pp. 141–150.
17. Henzinger, M. Finding near-duplicate web pages: A large-scale evaluation of algorithms. In Proceedings of the 29th Annual International ACM SIGIR Conference, Seattle, WA, USA, 6–11 August 2006; pp. 284–291.
18. Charikar, M.S. Similarity estimation techniques from rounding algorithms. In Proceedings of the 34th Annual ACM Symposium on Theory of Computing, Montreal, QC, Canada, 19–21 May 2002; pp. 380–388.
19. Reimers, N.; Gurevych, I. Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks. In Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing (EMNLP), Hong Kong, China, 3–7 November 2019; pp. 3982–3992.
20. Wang, L.; Yang, N.; Huang, X.; Jiao, B.; Yang, L.; Jiang, D.; Majumder, R.; Wei, F. Text Embeddings by Weakly-Supervised Contrastive Pre-training. *arXiv* **2022**, arXiv:2212.03533.
21. Johnson, J.; Douze, M.; Jégou, H. Billion-scale similarity search with GPUs. *IEEE Trans. Big Data* **2021**, *7*, 535–547. [CrossRef]
22. Buneman, P.; Khanna, S.; Tan, W.C. Why and where: A characterization of data provenance. In *Proceedings of the 8th International Conference on Database Theory (ICDT)*; Springer: Berlin/Heidelberg, Germany, 2001; pp. 316–330.
23. Herschel, M.; Diestelkämper, R.; Ben Lahmar, H. A survey on provenance: What for? What form? What from? *Vldb J.* **2017**, *26*, 881–906. [CrossRef]
24. Acar, A.; Aksu, H.; Uluagac, A.S.; Conti, M. A Survey on Homomorphic Encryption Schemes: Theory and Implementation. *ACM Comput. Surv.* **2018**, *51*, 79. [CrossRef]
25. Lindell, Y. How to Simulate It—A Tutorial on the Simulation Proof Technique. In *Tutorials on the Foundations of Cryptography*; Springer: Berlin/Heidelberg, Germany, 2017; pp. 277–346.
26. Pinkas, B.; Schneider, T.; Weinert, C.; Wieder, U. Efficient Circuit-Based PSI via Cuckoo Hashing. In *Proceedings of the Advances in Cryptology—EUROCRYPT 2018*; Springer: Berlin/Heidelberg, Germany, 2018; pp. 125–157.
27. Indyk, P.; Motwani, R. Approximate nearest neighbors: Towards removing the curse of dimensionality. In Proceedings of the 30th Annual ACM Symposium on Theory of Computing, Dallas, TX, USA, 24–26 May 1998; pp. 604–613.
28. Datar, M.; Immorlica, N.; Indyk, P.; Mirrokni, V.S. Locality-sensitive hashing scheme based on p-stable distributions. In Proceedings of the 20th Annual Symposium on Computational Geometry, Brooklyn, NY, USA, 8–11 June 2004; pp. 253–262.
29. Bloom, B.H. Space/time trade-offs in hash coding with allowable errors. *Commun. ACM* **1970**, *13*, 422–426. [CrossRef]
30. Dwork, C.; Roth, A. The Algorithmic Foundations of Differential Privacy. *Found. Trends Theor. Comput. Sci.* **2014**, *9*, 211–407. [CrossRef]
31. Levenshtein, V.I. Binary codes capable of correcting deletions, insertions, and reversals. *Sov. Phys. Dokl.* **1966**, *10*, 707–710.

32. Wilson, E.B. Probable inference, the law of succession, and statistical inference. *J. Am. Stat. Assoc.* **1927**, *22*, 209–212. [CrossRef]
33. Chicco, D.; Jurman, G. The advantages of the Matthews correlation coefficient (MCC) over F1 score and accuracy in binary classification evaluation. *BMC Genom.* **2020**, *21*, 6. [CrossRef] [PubMed]

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.