Scientometrics
https://doi.org/10.1007/s11192-026-05763-2

A multi-phase reference matching algorithm for bibliometric analysis: design, implementation, and evaluation

Massimo Aria[1,2] · **Luca D'Aniello**[1,2] · **Maria Spano**[1,2]

Received: 8 April 2026 / Accepted: 20 July 2026

Abstract
Bibliometric analyses rely on accurate citation counts, yet bibliographic databases routinely contain variant representations of the same cited reference, differing in journal abbreviation style, author name format, punctuation, or metadata completeness, that fragment citation links and distort standard indicators such as the *h*-index and journal impact metrics. We propose an unsupervised, multi-phase reference matching algorithm designed to consolidate these variants without requiring training data or external authority files beyond the ISO 4 List of Title Word Abbreviations (LTWA). The pipeline operates in seven phases: (i) format detection and string normalisation, which parses heterogeneous reference styles and standardises author names, titles, and pagination; (ii) ISO 4 journal-name normalisation, which maps both abbreviated and full journal names to a canonical short form using the LTWA; (iii) exact matching on DOI identifiers and normalised reference strings; (iv) blocking by first-author surname and publication year; (v) within-block fuzzy matching that combines Jaro-Winkler similarity with agglomerative hierarchical clustering to group near-duplicate references; (vi) post-processing metadata reconciliation, which merges complementary fields across matched records; and (vii) canonical representative selection, which elects the most informative variant as the group representative. Evaluation on a synthetic benchmark of 1 064 source articles under 17 controlled perturbation scenarios, yields precision, recall, and F_1 scores above 0.95 in 15 of 17 scenarios, with the two lowest-scoring scenarios still achieving $F_1 \geq 0.78$. Validation on two real-world Scopus datasets demonstrates that the algorithm reduces unique cited-reference counts by 4.6-−11.1 %, with corresponding increases in concentration-based bibliometric indicators. The algorithm is implemented in the open-source `bibliometrix` R package, which has been widely adopted in the scientometric community.

Keywords Reference matching · Bibliographic record linkage · Reference deduplication · Bibliometrics · String similarity

Extended author information available on the last page of the article

Published online: 08 August 2026

Introduction

Bibliometric analysis relies on accurate citation counting, yet the raw material underpinning every bibliometric indicator is surprisingly noisy. Among the major bibliographic databases, Web of Science (WoS) and Scopus still store cited references as semi-structured text strings rather than as resolved unique identifiers; by contrast, identifier-based providers such as OpenAlex (Priem et al., 2022) and Dimensions expose cited references as resolved record identifiers, which require no string matching. In the former, text-string representation, the same cited work routinely appears under multiple variant representations. Sources of variation include: (a) journal name formatting, where a single title may be recorded as the full name, an ISO 4 abbreviation, a WoS proprietary short title, or an ad-hoc truncation; (b) metadata completeness, with older references frequently lacking DOIs, volume numbers, or page ranges; (c) punctuation, capitalisation, and typographical inconsistencies; and (d) cross-database format differences, since WoS and Scopus adopt fundamentally different reference string layouts (Harzing & Alakangas, 2016). This fragmentation inflates the number of apparently unique cited works and distorts every downstream bibliometric measure.

The consequences of unresolved reference fragmentation are pervasive. At the author level, *h*-index calculations undercount the true impact of individual publications because citations to variant strings are never aggregated. At the journal level, impact metrics are diluted for the same reason. Network-based analyses suffer even more acutely: co-citation analysis (Small, 1973) produces spurious singletons when identical references are split across multiple nodes; bibliographic coupling underestimates the strength of intellectual links; and historiographic mapping introduces false gaps in citation chains. Empirical studies on medium-sized bibliometric datasets suggest that between 10% and 30% of unique cited-reference strings are duplicates of works already present under a different variant. Without correction, the resulting networks are sparser and less informative, and research-front detection loses sensitivity.

The problem of matching variant reference strings falls within the broader field of record linkage and duplicate detection (Fellegi & Sunter, 1969; Christen, 2012; Elmagarmid et al., 2007). Classical record-linkage methods provide a principled statistical framework, and string-similarity metrics such as Jaro-Winkler offer effective building blocks for comparing noisy textual fields. However, existing reference-matching approaches are either supervised, requiring labelled training data that are expensive to produce and may not generalise across disciplines or database formats, or limited to a single source format, or dependent on external APIs such as Crossref for DOI resolution (Hendricks et al., 2020). No existing method provides a fully unsupervised, offline pipeline that simultaneously handles multi-format detection, ISO 4 journal-name normalisation via the List of Title Word Abbreviations (LTWA), blocking for scalability, fuzzy matching, and cross-format metadata reconciliation.

This paper addresses that gap. We present an unsupervised seven-phase reference matching algorithm designed to handle the full spectrum of variation observed in real-world bibliometric datasets. The pipeline proceeds as follows: (1) it detects the reference format (WoS or Scopus) and normalises each string into a canonical representation; (2) it normalises journal names to their ISO 4 abbreviated form using the LTWA database, a step we introduce for the first time in a reference-matching context and one that eliminates the single largest source of reference fragmentation; (3) it performs exact matching on DOIs and fully normalised strings to resolve high-confidence duplicates deterministically; (4) it

applies a blocking strategy based on first-author surname and publication year to reduce the comparison space from $O(n^2)$ (i.e., quadratic growth in the number of pairwise comparisons) to approximately $O(n)$ (linear growth); (5) it computes Jaro-Winkler similarities within blocks and applies agglomerative hierarchical clustering to group fuzzy matches; (6) it reconciles clusters across formats through metadata-based post-processing; and (7) it selects the most complete reference variant within each cluster as the canonical representative. The algorithm is implemented in the open-source `bibliometrix` R package (Aria & Cuccurullo, 2017, 2026; Aria et al., 2026, 2024), one of the most widely used tools in the scientometric community, thereby ensuring broad accessibility and reproducibility.

The individual techniques used here are not new. DOI matching, string normalisation, blocking, Jaro-Winkler similarity and agglomerative clustering are all standard in the record-linkage literature. What this paper adds is their assembly into an unsupervised, reproducible pipeline for the abbreviated reference strings that bibliometric databases export, together with the LTWA-based ISO 4 journal normalisation step.

We evaluate the algorithm along three complementary dimensions. First, a synthetic benchmark comprising 1064 source articles with 17 controlled perturbation scenarios demonstrates match rates approaching 92% even when typographical noise affects up to 20% of characters, together with near-perfect precision exceeding 0.98 across all conditions. Second, two real-world Scopus datasets show that the algorithm reduces unique cited-reference counts by 4.6-−11.1%, recovering citation links that would otherwise be lost. Third, a scalability analysis confirms that runtime grows near-linearly with dataset size, making the pipeline practical for large-scale bibliometric studies.

The remainder of this paper is organised as follows. "Related work" section reviews the relevant literature on record linkage, string similarity, and reference matching. "Methodology" section presents the seven-phase algorithmic design in detail. "Implementation" section describes the software implementation within `bibliometrix`. "Experimental design" section defines the experimental protocol, including the synthetic data generation procedure and the real-world datasets. "Results" section reports the experimental results. "Discussion" section discusses findings, limitations, and practical implications. Conclusion section concludes the paper and outlines directions for future work.

Related work

Matching variant representations of the same entity across noisy records is a classical problem in statistics and computer science. Newcombe et al. (1959) introduced the first formal approach, and Fellegi and Sunter (1969) established the dominant probabilistic framework, in which each record pair is scored by the likelihood ratio of its field-agreement pattern under match versus non-match hypotheses. Comprehensive treatments of the resulting *record linkage* pipeline, from data cleaning and indexing through comparison and classification to evaluation, can be found in Elmagarmid et al. (2007) and Christen (2012).

At the core of any record-linkage system lies the comparison function that quantifies the similarity between textual fields. Edit-based metrics such as the Levenshtein distance (Levenshtein, 1966) count the minimum character operations needed to transform one string into another but treat all positions equally. Jaro (1989) proposed a similarity metric for short strings that weights common characters and their ordering, and (Winkler, 1990) added a prefix bonus that rewards agreement on initial characters. Cohen et al. (2003) showed that Jaro-Winkler consistently outperforms pure edit-distance metrics on

name-matching tasks, while (Piskorski and Sydow, 2007) confirmed this finding across multilingual settings. These results are directly relevant to reference matching, where the most discriminative information, the first-author surname and publication year, appears at the beginning of the string, making Jaro-Winkler a natural choice.

A second fundamental challenge is computational scalability. Naïve pairwise comparison of n records requires $O(n^2)$ operations, which is prohibitive for large bibliometric corpora. *Blocking* addresses this by partitioning records into groups based on a shared key, restricting detailed comparisons to within-group pairs. (Hernandez & Stolfo, 1998) addressed the merge/purge problem in real-world dirty data, proposing a sorted-neighbourhood approach that reduces comparisons to $O(wn)$ for a window of size w. Mccallum et al. (2000) introduced *canopy clustering*, a two-stage approach using a cheap metric to form overlapping canopies and an expensive metric within them, achieving near-linear scaling on bibliographic data. Baxter et al. (2003) compared several blocking strategies and recommended multi-pass approaches for robustness. Our algorithm adopts a composite blocking key combining author surname and publication year, which exploits the structure of reference strings to achieve effective partitioning with minimal false-negative risk.

Turning to reference matching specifically, early systems such as CiteSeer (Giles et al., 1998; Lawrence et al., 1999) demonstrated that automated citation indexing could achieve accuracy comparable to manual curation, but they operated on relatively homogeneous, full-text-parsed references rather than the abbreviated strings exported by WoS and Scopus. The probabilistic model of Pasula et al. (2002) provided principled uncertainty handling for identity resolution but at cubic computational cost; (Daumé & Marcu, 2005) proposed a Bayesian clustering framework that likewise requires labelled supervision and does not scale to large corpora. Foufoulas et al. (2017) proposed high-pass text filtering for the OpenAIRE infrastructure, offering speed at the expense of generality. In the related domain of systematic-review deduplication, (Hair et al., 2023) combines exact and fuzzy matching on full source records; however, it targets well-structured bibliographic entries rather than the abbreviated, metadata-sparse cited-reference strings that concern us here.

At industry scale, Crossref (Hendricks et al., 2020) offers a reference-matching API that resolves raw strings to DOIs against its corpus of over 150 million records, but it requires network access and cannot perform within-dataset deduplication. OpenAlex (Priem et al., 2022) assigns canonical identifiers by aggregating metadata from multiple sources; a recent reference-coverage analysis by Culbert et al. (2025) found that OpenAlex achieves comparable coverage to WoS and Scopus for recent publications but remains inconsistent for older non-source references. More broadly, comparative analyses of bibliographic databases have highlighted systematic discrepancies in citation counts arising from differences in reference resolution strategies (Visser et al., 2021). Nikolić et al. (2024) proposed an open-source tool for merging and deduplicating records from multiple citation databases, achieving up to 99% precision, while Nowakowska (Nowakowska, 2025) described a comprehensive preprocessing approach in which DOI-based unification is complemented by title and author normalisation. These industry-scale and multi-database approaches operate as external services or target source-record deduplication, and are unsuitable for offline within-dataset reference-string matching on user-supplied data.

Table 1 summarises the capabilities of representative approaches along six dimensions.

The review reveals several gaps. No existing method integrates ISO 4 journal-name normalisation via the LTWA into the matching pipeline; journal-title variation is either ignored or handled by incomplete lookup tables. Cross-format processing of WoS and Scopus reference strings has been studied empirically but not operationalised in a general-purpose algorithm. Supervised methods require costly labelled data that may not transfer

Table 1 Feature comparison of reference matching approaches. ● = full support, ◗ = partial, ○ = none

Approach	Unsupervised	Multi-format	ISO 4 norm.	Blocking	Fuzzy match	Open source
CiteSeer Giles et al. (1998)	●	○	○	○	◗	○
Pasula et al. (2002)	○	○	○	○	●	○
Daumé and Marcu (2005)	○	○	○	○	●	○
Hair et al. (2023)	●	○	○	●	●	●
Crossref (Hendricks et al., 2020; Hirai et al., 2025)	●	◗	○	●	●	◗
Our method	●	●	●	●	●	●

across disciplines, while industry-scale services depend on external APIs and cannot support offline, reproducible analysis. Our algorithm is the first to combine unsupervised operation, multi-format detection, LTWA-based journal normalisation, blocking for scalability, fuzzy matching, and open-source availability in a single integrated pipeline; what is new is the combination, since the individual components are established techniques.

Methodology

The formal specification of the algorithm proceeds in three parts: we first establish notation and problem formulation, then describe each phase in detail, and conclude with a complexity analysis of the full pipeline.

Problem formulation and notation

Let $\mathcal{C} = \{c_1, c_2, \ldots, c_n\}$ denote a corpus of n raw cited-reference strings extracted from one or more bibliographic databases. Each string c_i is a Unicode character sequence that encodes, in a database-specific format, the bibliographic metadata of the work being cited (author names, publication year, source title, volume, pagination, and, optionally, a DOI).

Definition 1 (Reference matching) A *reference matching* is an equivalence relation $\sim$ on $\mathcal{C}$ such that $c_i \sim c_j$ if and only if c_i and c_j refer to the same underlying publication. The quotient set $\mathcal{C}/\sim$ partitions the corpus into equivalence classes (clusters) $\{G_1, G_2, \ldots, G_K\}$, where $K \leq n$ and each G_k groups all variant representations of a single cited work.

We adopt the term *reference matching* to denote the process of resolving raw bibliographic strings to canonical records, reserving *citation* for the relational act of citing. A few related terms are easily conflated. *Record linkage* is the general statistical problem of identifying co-referent records across or within files (the Fellegi–Sunter tradition). *Entity resolution*, or *deduplication*, is the within-file case, where co-referent records are collapsed.

The objective is to recover $\sim$ from the raw strings alone, without access to external authority records or supervised training data. We approximate $\sim$ through a sequence

of increasingly permissive matching phases, each operating on the residual set of unmatched references left by the preceding phase.

We write $\mathcal{M} = \{\textit{first\_author}, \textit{year}, \textit{journal}, \textit{volume}, \textit{pages}, \textit{DOI}, \textit{title\_words}\}$ for the metadata space associated with a parsed reference. A feature-extraction function $\xi : \mathcal{C} \to \mathcal{M}$ maps each raw string to a structured metadata record. Not every field is populated for every reference; we write $\xi_f(c)$ for the value of field f extracted from c, with $\xi_f(c) = \varnothing$ when the field is absent.

The pipeline comprises seven phases, summarised in Algorithm 1, illustrated in Fig. 1, and formalised in "Phase 1: format detection and normalisation"–"Phase 7: canonical representative selection" section.

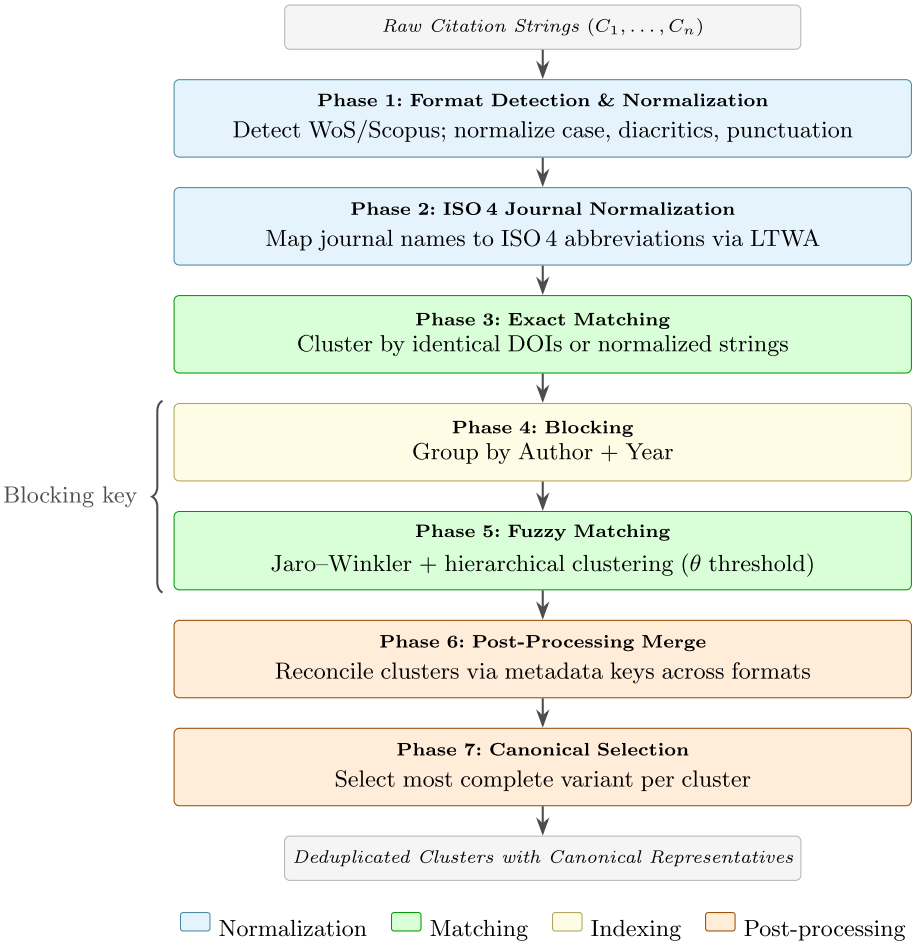

Fig. 1 Overview of the seven-phase reference matching pipeline. Phases are colour-coded by function: normalisation (blue), matching (green), indexing (yellow), and post-processing (orange). The blocking key (left brace) determines which references are compared in Phase 5. (Color figure online)

Algorithm 1 Multi-phase reference matching pipeline

Require: Corpus $\mathcal{C} = \{c_1, \ldots, c_n\}$; similarity threshold $\theta \in (0, 1]$; string distance d
Ensure: Partition $\mathcal{P} = \{G_1, \ldots, G_K\}$ of $\mathcal{C}$; canonical representative r_k for each G_k

Phase 1 – Format Detection & String Normalisation. For each $c \in \mathcal{C}$: detect format $f(c)$; compute $\hat{c} \leftarrow \varphi(c)$; extract $\xi(c)$.
Phase 2 – ISO 4 Journal Name Normalisation. Apply journal normalisation $\mathcal{J}$ to $\xi_{journal}(c)$.
Phase 3 – Exact matching. Cluster references sharing a valid DOI. Cluster remaining references with identical normalised strings $\hat{c}$. Cluster remaining references with identical punctuation-stripped strings $\sigma(c)$.
Phase 4 – Blocking. Compute blocking key $\beta(c)$ for each unmatched c. Partition into blocks $\{B_1, \ldots, B_L\}$.
Phase 5 – Fuzzy matching.
for each block B_l **do**
    if B_l contains only WoS-formatted references **then**
        Match deterministically on metadata key.
    else
        Compute pairwise distance matrix $\mathbf{D}$ using d.
        Convert to similarity: $\mathbf{S} = 1 - \mathbf{D}/\max(\mathbf{D})$.
        Cluster with agglomerative hierarchical clustering (complete linkage); cut at $h = 1 - \theta$.
    end if
end for
Phase 6 – Post-processing merge. Reconcile clusters via format-specific metadata merge keys.
Phase 7 – Canonical selection. For each cluster G_k, select $r_k = \arg\max_{c \in G_k} \text{score}(c)$.
return $\mathcal{P}, \{r_1, \ldots, r_K\}$

Phase 1: format detection and normalisation

Bibliographic databases export cited references in database-specific formats. Our algorithm recognises three principal formats and a residual class.

Definition 2 The format detection function $f : \mathcal{C} \to \mathcal{F}$, where $\mathcal{F} = \{\text{WoS}, \text{Scopus}, \text{ScopusNew}, \text{Unknown}\}$, assigns each reference to its originating format based on structural patterns:

$$f(c) = \begin{cases} \text{WoS} & \text{if } c \text{ matches } \texttt{AUTHOR}, \texttt{YEAR}, \texttt{JOURNAL}, \texttt{V\#\#}, \texttt{P\#\#[}, \texttt{DOI]}, \\ \text{Scopus} & \text{if } c \text{ matches } \texttt{AUTHORS}, \texttt{TITLE (YEAR) JOURNAL}, \texttt{VOL}, \texttt{PP}, \\ \text{ScopusNew} & \text{if } c \text{ matches } \texttt{AUTHORS}, \texttt{TITLE}, \texttt{JOURNAL}, \texttt{VOL}, \texttt{PP (YEAR)}, \\ \text{Unknown} & \text{otherwise.} \end{cases} \quad (1)$$

The WoS format places the publication year immediately after the first author, encodes the volume with a `V` prefix and the starting page with a `P` prefix, and may append a DOI prefixed by `DOI` . The classical Scopus format lists all authors, followed

by the article title, the year in parentheses, the journal name, volume, and page range. The newer ScopusNew variant, available in exports from Scopus since at least late 2025, relocates the year to the end of the string enclosed in parentheses. Format detection is performed via regular-expression matching and succeeds on the vast majority of records; the small fraction assigned to Unknown is carried forward but excluded from metadata-based matching steps.

String normalisation

Raw reference strings exhibit substantial variation in capitalisation, diacritics, punctuation, and whitespace. We define a normalisation function φ as the composition of five transformations, applied in fixed order:

$$\varphi = \tau \circ \rho \circ \pi \circ \delta \circ \upsilon, \tag{2}$$

where the composition is evaluated right-to-left: υ is applied first and τ last. Each component is specified as follows:

The first operator, *case folding* (υ), converts every character to its uppercase equivalent under the Unicode NFKC normalisation form. *Diacritic removal* (δ) then applies Unicode canonical decomposition (NFD), strips all combining marks, and re-encodes the result in ASCII, mapping accented characters to their base forms (e.g., "ö" → "O", "ç" → "C") so that author names with inconsistent transliterations are brought into alignment. *Punctuation normalisation* (π) removes or standardises punctuation symbols: hyphens within author surnames are preserved, while all other punctuation is deleted or replaced by a single space depending on positional context. *Issue and page-range removal* (ρ) strips parenthetical issue numbers (e.g., "`(3-4)`") and trailing page ranges that duplicate the starting-page field already extracted by ξ, eliminating a common source of spurious variation across databases. Finally, *whitespace normalisation* (τ) collapses runs of whitespace to a single ASCII space and trims leading and trailing whitespace.

The composition φ is idempotent in practice: $\varphi(\varphi(c)) = \varphi(c)$ for all well-formed reference strings $c \in \mathcal{C}$.

Feature extraction

The feature-extraction function $\xi : \hat{\mathcal{C}} \to \mathcal{M}$ parses each normalised reference $\hat{c} = \varphi(c)$ according to the format $f(c)$ and populates the metadata record. Two validity filters are applied:

1. **Year validation.** The extracted year must satisfy $1700 \leq \xi_{year}(c) \leq y_{\max}$, where $y_{\max}$ is the current calendar year plus one. Values outside this range are treated as missing ($\xi_{year}(c) = \varnothing$).
2. **Minimum length.** Formally, only references satisfying $|c| \geq 20$ are retained; the effective corpus is $\mathcal{C}' = \{c \in \mathcal{C} : |c| \geq 20\}$, and the pipeline operates on $\mathcal{C}'$ throughout. This threshold discards malformed or truncated fragments, such as bare years or placeholders like "Ibid.", that carry insufficient information for reliable matching, while remaining short enough to retain all well-formed references.

Phase 2: journal name normalisation via ISO 4

Journal names are among the most variable fields in cited references. A single periodical may appear under its full title, one of several abbreviated forms, or an ad-hoc truncation. We address this through a dedicated normalisation step that maps every journal name to a canonical abbreviated form defined by the ISO 4 standard (International Organization for Standardization, 1997).

ISO 4 specifies rules for abbreviating the words in serial titles, drawing on the *List of Title Word Abbreviations* (LTWA), a curated resource maintained by the ISSN International Centre that contains more than 56,000 entries across multiple languages. Our implementation filters this list to English-language entries, yielding approximately 14,000 active abbreviation rules, and encodes them in a trie data structure. The normalisation then applies the following four-step procedure.

Definition 3 Let Σ denote the character alphabet and let $T \subseteq \Sigma^*$ be the vocabulary of tokens (non-empty character strings containing no whitespace). The journal normalisation function $\mathcal{J} : \Sigma^* \to \Sigma^*$ maps a journal name string to its ISO 4 abbreviated form through the composition

$$\mathcal{J} = \mu \circ \alpha_{fallback} \circ \alpha_{single} \circ \alpha_{multi} \circ \varsigma, \tag{3}$$

where $\varsigma : \Sigma^* \to T^*$ jointly tokenises the input and removes stop-words, $\alpha_{multi}, \alpha_{single}, \alpha_{fallback} : T^* \to T^*$ are token-sequence transformations, and $\mu : T^* \to \Sigma^*$ is the detokeniser that joins tokens with a single space. The composition is evaluated right-to-left: ς is applied first and μ last.

Step 1: Stop-word removal (ς). The journal name string is first *tokenised* after which tokens matching the stop-word list defined by ISO 4 are discarded. These include articles (e.g., "the", "a", "an", "le", "der"), prepositions (e.g., "of", "in", "for", "de", "zur"), and conjunctions (e.g., "and", "et", "und") in all languages covered by the LTWA.

Let $\mathcal{T} : \Sigma^* \to T^*$ denote the whitespace tokeniser, so that $\mathcal{T}(s) = (w_1, w_2, \ldots, w_m)$, and let $\mathcal{S} \subset T$ be the stop-word set. Then

$$\varsigma(s) = (w_{i_1}, w_{i_2}, \ldots, w_{i_p}), \qquad \{i_1, \ldots, i_p\} = \{j : w_j \notin \mathcal{S}\}, \tag{4}$$

where $(w_1, \ldots, w_m) = \mathcal{T}(s)$.

Step 2: Multi-word phrase matching (α_{multi}). Scan the remaining token sequence for contiguous subsequences that match multi-word entries in the LTWA (e.g., "CELL BIOLOGY" → "CELL BIOL"). Matching proceeds left-to-right with longest-match priority: if both a two-word and a three-word LTWA entry begin at the same position, the three-word entry takes precedence.

Step 3: Single-word abbreviation (α_{single}). For each token not yet consumed by the multi-word step, attempt abbreviation by the following cascade:

1. *Exact match*: if the token appears verbatim in the LTWA, replace it with the prescribed abbreviation.
2. *Prefix match*: if no exact match exists, find the longest LTWA entry whose pattern is a prefix of the token (e.g., "PRODUCT" matched by the LTWA pattern "PRODUC-"

yields abbreviation "PROD"). In the event of multiple prefix matches of equal length, the entry whose language field matches the majority language of the title is preferred.
3. *No match*: if neither exact nor prefix matching succeeds, the token is left unchanged.

Step 4: Common abbreviation fallback ($\alpha_{fallback}$). A supplementary dictionary of more than 100 common scientific abbreviations (e.g., "INTERNATIONAL" → "INT", "SCIENCE" → "SCI", "RESEARCH" → "RES") is consulted for any token that survived Steps 2 and 3 without abbreviation. This fallback captures domain-specific conventions that the LTWA, which aims for cross-disciplinary generality, does not encode.

Example 1 Applying $\mathcal{J}$ to the string `JOURNAL OF CLEANER PRODUCTION`:

$$\begin{aligned}
\varsigma &: \texttt{JOURNALOFCLEANERPRODUCTION} \rightarrow \texttt{JOURNALCLEANERPRODUCTION},\\
\alpha_{multi} &: \text{no multi-word match},\\
\alpha_{single} &: \texttt{JOURNAL} \rightarrow \texttt{J}, \quad \texttt{CLEANER} \rightarrow \texttt{CLEAN}, \quad \texttt{PRODUCTION} \rightarrow \texttt{PROD},\\
\alpha_{fallback} &: \text{no remaining unabbreviated tokens},\\
\mathcal{J} &: \texttt{JOURNALOFCLEANERPRODUCTION} \rightarrow \texttt{JCLEANPROD}.
\end{aligned}$$

Note that, for the purposes of reference matching, trailing abbreviation periods are omitted from the canonical form; this does not affect compliance with ISO 4 matching rules since periods are removed during the normalisation step φ. By normalising both WoS-style abbreviated titles and Scopus-style full titles to a common ISO 4 form, $\mathcal{J}$ enables cross-database matching that would otherwise fail due to inconsistent journal name representations. This component constitutes, to our knowledge, the first integration of LTWA-based ISO 4 abbreviation into a reference matching pipeline.

Table 2 shows $\mathcal{J}$ at work on journal names taken from the real Scopus corpora. What matters for matching is convergence: a full title and its cited abbreviation map to the same canonical form (`JOURNAL OF CLEANER PRODUCTION` and `J CLEAN PROD` both give `J CLEAN PROD`), so two references that differ only in journal-name style are brought together. The table also covers multi-word phrase abbreviation with stop-word removal and longest-match prefix resolution.

Phase 3: exact matching

The first matching phase exploits two deterministic criteria that yield high-precision clusters at negligible computational cost.

DOI matching

A Digital Object Identifier (DOI) is a persistent identifier that uniquely designates a digital object. Valid DOIs conform to the pattern

$$\texttt{10.d\{4,\}/+} \tag{5}$$

(i.e., the prefix "10." followed by at least four digits, a slash, and one or more characters). For every pair of references (c_i, c_j) satisfying $\xi_{DOI}(c_i) \neq \emptyset$, $\xi_{DOI}(c_j) \neq \emptyset$, and $\xi_{DOI}(c_i) = \xi_{DOI}(c_j)$, we set $c_i \sim c_j$. Because DOIs are globally unique by construction,

Table 2 Worked examples of ISO 4 journal normalisation ($\mathcal{J}$) on real Scopus journal names

Mechanism	Input form	Canonical ISO 4
Full ↔ abbrev. convergence (same canonical)	JOURNAL OF CLEANER PRODUCTION	J CLEAN PROD
	J CLEAN PROD	J CLEAN PROD
Full ↔ abbrev. convergence (same canonical)	JOURNAL OF INFORMETRICS	J INFORMETR
	J INFORMETRICS	J INFORMETR
Multi-word phrase (stop-words removed)	TECHNOLOGICAL FORECASTING AND SOCIAL CHANGE	TECHNOL FORECAST SOC CHANG
Multi-word phrase (stop-words removed)	PROCEEDINGS OF THE NATIONAL ACADEMY OF SCIENCES	PROC NATL ACAD SCI
Longest-match prefix	RESEARCH POLICY	RES POLICY
Longest-match prefix	MANAGEMENT SCIENCE	MANAG SCI

DOI matching has a theoretical false-positive rate of zero, provided the DOIs are correctly extracted.

Formally, let $\mathcal{C}_{DOI} = \{c \in \mathcal{C} : \xi_{DOI}(c) \neq \varnothing\}$ denote the subset of references bearing a valid DOI. We define the equivalence classes induced by DOI matching as

$$G_d^{DOI} = \{c \in \mathcal{C}_{DOI} : \xi_{DOI}(c) = d\}, \qquad d \in \{\xi_{DOI}(c) : c \in \mathcal{C}_{DOI}\}. \tag{6}$$

Normalised string matching

Among the references not matched by DOI, those whose normalised forms are identical are clustered:

$$G_s^{str} = \{c \in \mathcal{C}' \setminus \textstyle\bigcup_d G_d^{DOI} : \varphi(c) = s\}, \qquad s \in \{\varphi(c) : c \in \mathcal{C} \setminus \textstyle\bigcup_d G_d^{DOI}\}. \tag{7}$$

By applying φ before comparison, this step captures pairs that differ only in capitalisation, diacritics, punctuation, or whitespace, the most common sources of superficial variation.

Punctuation-invariant matching

A third deterministic pass addresses formatting noise that survives the normalisation φ, for example, missing commas, doubled spaces, or substituted dashes. For each still-unmatched reference c, we compute a *stripped* representation $\sigma(c)$ obtained by (i) removing diacritics, (ii) converting to uppercase, and (iii) deleting every character that is not a letter or a digit:

$$\sigma(c) = \mathrm{strip}_{\neg[\mathtt{A-Z0-9}]}(\mathrm{upper}(\mathrm{deaccent}(c))). \tag{8}$$

References sharing the same stripped form are clustered: $G_s^{\sigma} = \{c : \sigma(c) = s\}$. Because σ is applied to the original reference string rather than to the reconstructed normalised form, it is immune to any parsing or field-extraction errors introduced by Phases 1–2, operating directly on the raw input. This step adds negligible computational cost (a single linear pass over the residual set) and, as shown in Sect. Results, resolves nearly all formatting-only variants.

Phase 4: blocking

After exact matching, the residual set $\mathcal{C}_R \subseteq \mathcal{C}'$ of unmatched references must be compared pairwise to detect near-duplicate pairs. Naive all-pairs comparison has quadratic complexity $O(|\mathcal{C}_R|^2)$, which is prohibitive for large bibliometric corpora. We therefore adopt a blocking strategy that partitions $\mathcal{C}_R$ into small, homogeneous groups within which pairwise comparison is tractable.

Definition 4 The blocking function $\beta : \mathcal{C}_R \to \Sigma^*$ maps each reference to a blocking key defined as

$$\beta(c) = \xi_{first\_author}(c) \parallel \xi_{year}(c), \tag{9}$$

where $\parallel$ denotes string concatenation with an underscore separator. The blocking key combines only the first-author surname and the publication year; journal-name variation is not part of the key but is reconciled within blocks by the ISO 4 normalisation of Phase 2 and

by the within-block similarity computation of Phase 5. References for which any component of $\beta(c)$ is absent ($\xi_f(c) = \varnothing$ for some required field f) are assigned to a dedicated fallback block and processed using exact string matching only, forgoing fuzzy comparison.

The rationale is that two references referring to the same publication almost always share the first author's surname and the publication year; discrepancies in these fields are rare and typically indicate distinct works.

The blocking key induces a partition $\{B_1, B_2, \ldots, B_L\}$ of $\mathcal{C}_R$:

$$B_l = \{c \in \mathcal{C}_R : \beta(c) = b_l\}, \qquad l = 1, \ldots, L. \tag{10}$$

Only references within the same block are compared in Phase 5. The complexity reduction is substantial: denoting the block sizes by $n_l = |B_l|$, the number of pairwise comparisons drops from $\binom{|\mathcal{C}_R|}{2}$ to $\sum_{l=1}^{L} \binom{n_l}{2}$. In practice, the distribution of block sizes is heavily right-skewed, with most blocks containing fewer than ten references; see Sect. Results for empirical measurements.

Theorem 1 *Let $\bar{n} = |\mathcal{C}_R|/L$ denote the average block size. Then the total number of within-block comparisons satisfies*

$$\sum_{l=1}^{L} \binom{n_l}{2} = \frac{|\mathcal{C}_R|}{2}(\bar{n} - 1) + \frac{L}{2}\,\mathrm{Var}(n_l) \tag{11}$$

which is $O(|\mathcal{C}_R| \cdot \bar{n})$ when the variance of block sizes is bounded. Since $\bar{n} \ll |\mathcal{C}_R|$ in bibliometric corpora, this represents a reduction from $O(|\mathcal{C}_R|^2)$ to near-linear complexity.

Proof By the identity $\sum_l n_l^2 = L\,\bar{n}^2 + L\,\mathrm{Var}(n_l)$ (which follows directly from the definition of variance, $\mathrm{Var}(n_l) = \frac{1}{L}\sum_l n_l^2 - \bar{n}^2$), we have

$$\begin{aligned} \sum_{l=1}^{L} \binom{n_l}{2} &= \frac{1}{2}\left(\sum_{l=1}^{L} n_l^2 - \sum_{l=1}^{L} n_l\right) \\ &= \frac{1}{2}\left(L\,\bar{n}^2 + L\,\mathrm{Var}(n_l) - |\mathcal{C}_R|\right) \\ &= \frac{|\mathcal{C}_R|}{2}(\bar{n} - 1) + \frac{L}{2}\,\mathrm{Var}(n_l). \end{aligned} \tag{12}$$

□ □

Phase 5: within-block fuzzy matching

Within each block B_l, references are compared and grouped using one of two strategies, depending on the reference format.

Deterministic matching for WoS references

WoS cited references have a rigid, field-delimited structure that supports deterministic matching. For a block B_l containing only WoS-format references, we define a metadata key

$$\kappa_{\mathrm{WoS}}(c) = \big(\xi_{\mathit{first\_author}}(c),\ \xi_{\mathit{year}}(c),\ \mathcal{J}(\xi_{\mathit{journal}}(c)),\ \xi_{\mathit{volume}}(c),\ \xi_{\mathit{page\_start}}(c)\big) \tag{13}$$

and set $c_i \sim c_j$ whenever $\kappa_{\mathrm{WoS}}(c_i) = \kappa_{\mathrm{WoS}}(c_j)$ and both keys are fully populated (no $\varnothing$ component). The quintuple (author, year, journal, volume, starting page) is almost always unique to a given article, making this a high-precision matching rule.

Fuzzy matching for Scopus and mixed-format blocks

When a block contains Scopus-formatted references or a mix of formats, the metadata fields may be incomplete or inconsistently structured, necessitating a similarity-based approach.

String distance. Let $d : \Sigma^* \times \Sigma^* \to [0, 1]$ be a normalised string distance function. The default choice is the Jaro-Winkler distance, which we now define.

Definition 5 Given two strings s_1 and s_2 of lengths $|s_1|$ and $|s_2|$, define the *match window* as $w = \lfloor \max(|s_1|, |s_2|)/2 \rfloor - 1$. A character $s_1[i]$ is said to *match* $s_2[j]$ if $s_1[i] = s_2[j]$ and $|i - j| \le w$. Let m denote the number of matching characters and t the number of transposition halves: if t' matched characters appear out of order, then $t = t'/2$. The Jaro similarity is

$$\mathrm{sim}_J(s_1, s_2) = \begin{cases} 0 & \text{if } m = 0, \\ \dfrac{1}{3}\left(\dfrac{m}{|s_1|} + \dfrac{m}{|s_2|} + \dfrac{m - t}{m} \right) & \text{otherwise.} \end{cases} \tag{14}$$

Definition 6 The Jaro-Winkler similarity augments the Jaro similarity with a prefix bonus:

$$\mathrm{sim}_{JW}(s_1, s_2) = \mathrm{sim}_J(s_1, s_2) + \ell \cdot p \cdot \big(1 - \mathrm{sim}_J(s_1, s_2)\big), \tag{15}$$

where $\ell = \min(|\operatorname{lcp}(s_1, s_2)|, 4)$ is the length of the longest common prefix (capped at 4) and $p = 0.1$ is the Winkler scaling constant.

The corresponding Jaro-Winkler distance is $d_{JW}(s_1, s_2) = 1 - \mathrm{sim}_{JW}(s_1, s_2)$. The algorithm also supports three alternative metrics, selectable via a configuration parameter: the Levenshtein distance (Levenshtein, 1966), which counts the minimum number of single-character insertions, deletions, and substitutions needed to transform one string into another; the optimal string alignment (OSA) distance (Damerau, 1964), which extends Levenshtein by also allowing transpositions of adjacent characters (each substring may be edited at most once); and the longest common subsequence (LCS) distance (Wagner & Fischer, 1974), which counts only insertions and deletions, equivalent to $|s_1| + |s_2| - 2\,|\operatorname{lcs}(s_1, s_2)|$. Levenshtein and OSA distances are normalised by the length of the longer string; the same quantity normalises LCS distance. Jaro-Winkler is intrinsically bounded in [0, 1] and requires no further normalisation.

Distance matrix and similarity conversion. For a block $B_l = \{c_1^{(l)}, \ldots, c_{n_l}^{(l)}\}$, we compute the $n_l \times n_l$ pairwise distance matrix

$$\mathbf{D}_{ij}^{(l)} = d\big(\varphi(c_i^{(l)}),\ \varphi(c_j^{(l)})\big), \qquad 1 \le i, j \le n_l. \tag{16}$$

The distance matrix is converted to a similarity matrix by

$$\mathbf{S}_{ij}^{(l)} = 1 - \frac{\mathbf{D}_{ij}^{(l)}}{\max_{i,j} \mathbf{D}_{ij}^{(l)}}, \tag{17}$$

When $\max_{i,j} \mathbf{D}_{ij}^{(l)} = 0$, i.e., all references within the block are mutually identical, the block is collapsed to a single cluster without invoking the clustering step.

Hierarchical agglomerative clustering. We apply agglomerative hierarchical clustering with *complete linkage* to the dissimilarity matrix $1 - \mathbf{S}^{(l)}$. Complete linkage defines the distance between two clusters A and B as

$$d_{complete}(A, B) = \max_{c_i \in A,\, c_j \in B} (1 - \mathbf{S}_{ij}^{(l)}), \tag{18}$$

which is the most conservative agglomerative criterion: two clusters merge only when the maximum pairwise distance across all inter-cluster pairs does not exceed the distance threshold $1 - \theta$; equivalently, every pair of elements must be at least as similar as θ. This choice favours precision over recall, producing tight, homogeneous clusters.

The resulting dendrogram is cut at height $h = 1 - \theta$, where $\theta \in (0, 1]$ is a user-specified similarity threshold (default $\theta = 0.90$). References assigned to the same subtree at this cut height form a cluster:

$$c_i \sim c_j \iff \text{the complete-linkage merge height of } c_i \text{ and } c_j \text{ is} \leq 1 - \theta. \tag{19}$$

The choice of complete linkage paired with a conservative threshold reflects the asymmetric cost structure of reference matching: a false positive (merging distinct publications) corrupts downstream bibliometric indicators more severely than a false negative (failing to merge true duplicates), because the former introduces phantom citation links while the latter merely leaves existing fragmentation unresolved.

Phase 6: post-processing metadata merge

After the clustering in Phase 5, an additional reconciliation step identifies clusters that should be merged but were separated because their members fell into different blocks or because string-level similarity fell below the threshold despite referring to the same work.

This step defines format-specific merge keys that exploit structured metadata fields rather than raw string similarity. Let G_k and $G_{k'}$ be two distinct clusters. They are merged ($G_k \cup G_{k'} \to G_k$) if there exist $c \in G_k$ and $c' \in G_{k'}$ sharing a common merge key. The merge keys are defined hierarchically, from most to least specific:

$$\mu(c) = \begin{cases} \xi_D & f = \text{WoS},\ \xi_D \neq \varnothing, \\ (\xi_A,\, \xi_Y,\, \mathcal{J}(\xi_J),\, \xi_V,\, \xi_P) & f = \text{WoS},\ \xi_D = \varnothing, \\ (\xi_A,\, \xi_Y,\, \mathcal{J}(\xi_J),\, \xi_V,\, \xi_P,\, \xi_T) & f \in \{\text{Scopus}, \text{ScopusNew}\},\ \xi_T \neq \varnothing, \\ (\xi_A,\, \xi_Y,\, \mathcal{J}(\xi_J),\, \xi_V) & \text{otherwise}, \end{cases} \tag{20}$$

where we abbreviate $\xi_A = \xi_{first\_author}(c)$, $\xi_Y = \xi_{year}(c)$, $\xi_J = \xi_{journal}(c)$, $\xi_V = \xi_{volume}(c)$, $\xi_P = \xi_{page}(c)$, $\xi_D = \xi_{DOI}(c)$, $\xi_T = \xi_{title\_kw}(c)$ (content-bearing title keywords after stop-word removal), and $f = f(c)$.

The hierarchical design ensures that the most discriminative information available for each format is used. The DOI key is infallible when present; the five-field WoS key is nearly so; the Scopus key augmented with title keywords adds discriminative power for

Scopus records that frequently share author, year, journal, and volume (e.g., conference proceedings or journal special issues); and the four-field fallback key provides a last-resort matching criterion for records with minimal metadata.

During the merge step, metadata fields are also reconciled across the newly merged cluster. When one record contributes a DOI that another lacks, or provides volume and page information absent from its partner, the fields are propagated to produce a composite metadata record that is more complete than any individual member.

Phase 7: canonical representative selection

Each cluster G_k may contain multiple variant representations of the same cited work. For downstream bibliometric analysis, a single canonical representative must be selected to serve as the unique identifier of the cluster.

Definition 7 The completeness score of a reference c is defined as

$$\text{score}(c) = 100 \cdot \mathbf{1}[\xi_{DOI}(c) \neq \varnothing] + 10 \cdot \mathbf{1}[\xi_{volume}(c) \neq \varnothing] + 5 \cdot \mathbf{1}[\xi_{page\_start}(c) \neq \varnothing] + 0.01 \cdot |c|, \tag{21}$$

where $\mathbf{1}[\cdot]$ is the indicator function and $|c|$ is the character length of the raw reference string.

The weights encode a priority ordering: a DOI dominates all other fields because it is a persistent unique identifier; volume and page information are next in importance because they locate the article within the source; and string length serves as a tiebreaker, on the rationale that longer strings tend to carry more complete metadata (e.g., full author lists and titles).

The canonical representative for cluster G_k is

$$r_k = \arg\max_{c \in G_k} \text{score}(c). \tag{22}$$

In the event of a tie (equal score for two or more references), the lexicographically first string is selected, ensuring deterministic output.

Overall complexity analysis

We now analyse the computational complexity of the full pipeline.

Theorem 2 *Let $n = |\mathcal{C}'|$ be the corpus size, K the number of blocks produced by Phase 4, $n_k = |B_k|$ the size of the k-th block, and $\bar{\ell}$ the average reference string length. The total time complexity of the algorithm is*

$$T(n) = O\left(n\bar{\ell} \; + \; n \log n \; + \; \sum_{k=1}^{K} n_k^2 \cdot \bar{\ell} \right). \tag{23}$$

Proof We bound the cost of each phase separately.

Phase 1 (normalisation and feature extraction). Each of the n references is processed in time proportional to its length. The aggregate cost is $O(n\bar{\ell})$.

Phase 2 (journal normalisation). Each journal name is tokenised and each token is looked up in the LTWA trie. Since journal names have bounded length and trie lookup is $O(\bar{\ell})$ per token, the total cost across all references is $O(n\bar{\ell})$.

Phase 3 (exact matching). DOI matching requires hashing the DOIs of all n references and grouping by hash, costing $O(n)$ expected time. Normalised string matching likewise requires hashing the normalised strings, costing $O(n\bar{\ell})$ for hash computation. The punctuation-invariant pass applies a character-filtering function to each reference string and groups by the resulting key, adding another $O(n\bar{\ell})$ term.

Phase 4 (blocking). Computing blocking keys and partitioning references by key requires $O(n\bar{\ell})$ for key construction and $O(n \log n)$ for sorting or hash-based grouping.

Phase 5 (fuzzy matching). Within each block B_k of size n_k, computing the $\binom{n_k}{2}$ pairwise string distances takes $O(n_k^2 \cdot \bar{\ell})$ time (each distance computation is $O(\bar{\ell})$ for Jaro-Winkler). Agglomerative clustering with complete linkage runs in $O(n_k^2)$ using a nearest-neighbour chain algorithm. The aggregate cost across all blocks is $\sum_{k=1}^{K} O(n_k^2 \cdot \bar{\ell})$.

Phases 6 and 7 (post-processing and canonical selection). These phases involve hash-based grouping and linear scans over all references, costing $O(n)$.

Summing across all phases yields the stated bound. □

In typical bibliometric datasets, the blocking strategy ensures that $n_k \leq n_{\max}$ for a small constant $n_{\max}$ (empirically, $n_{\max} < 50$ for more than 99% of blocks). Under this assumption, $\sum_{k=1}^{K} n_k^2 \leq n_{\max} \sum_{k=1}^{K} n_k = n_{\max} \cdot |\mathcal{C}_R| \leq n_{\max} \cdot n$, and, since $n_{\max}$ is treated as a dataset-dependent constant, the overall complexity simplifies to

$$T(n) = O(n \cdot (1 + n_{\max}) \cdot \bar{\ell}) = O(n \cdot \bar{\ell}), \tag{24}$$

which is linear in the corpus size, a substantial improvement over the $O(n^2\bar{\ell})$ cost of naive pairwise comparison.

Implementation

The multi-phase reference matching algorithm is implemented in R (R Core Team, 2024) and distributed as part of the `bibliometrix` package (version 5.4 and later) (Aria & Cuccurullo, 2017), available on CRAN and GitHub,[1] with a programmatic interface, configurable parameters, and integration with the Biblioshiny graphical user interface described below.

Software architecture

The user interface consists of two exported functions that map directly onto the seven phases described in Algorithm 1 and Fig. 1.

The high-level wrapper `applyReferenceMatching(M, threshold=0.90, method="jw", min_chars=20)` accepts a bibliometrix data frame `M` and returns a modified data frame in which the cited-reference field (`CR`)

[1] https://github.com/massimoaria/bibliometrix

has been replaced by its normalised counterpart. `M` is a `data.frame` produced by `bibliometrix`'s database conversion routines, representing a collection of articles downloaded from one of the supported bibliographic databases (e.g., Web of Science, Scopus, PubMed). Each row corresponds to a single article and each column to a bibliographic metadata field (e.g., authors, title, source, year, cited references). This function extracts the reference string vector from `M`, delegates the matching logic to the core function described below, and reintegrates the results into the data frame, updating all reference-dependent fields, including `CR_AU` (citing-author list), `CR_SO` (cited-source list), and related derived fields; full details are documented in the `bibliometrix` package manual.

The core matching function `normalize_citations(CR_vector, threshold=0.90, method="jw", min_chars=20)` operates directly on a character vector of raw reference strings and implements the full seven-phase pipeline described in "Methodology" section. This separation of concerns allows the matching engine to be invoked independently of the bibliometrix data frame infrastructure, facilitating use in custom workflows and batch-processing pipelines.

Both exported functions expose the same three configurable parameters. `threshold` (default 0.90) is the similarity cut-off θ applied in Phase 5, and `method` (default "jw") selects the string-distance function; the effect of both is examined in "Robustness to the string-distance metric" and "Threshold stability" sections. The `min_chars` parameter (default 20, the `bibliometrix` package default) sets the minimum reference-string length: strings shorter than this, typically malformed or truncated fragments such as bare years or "Ibid." placeholders, are excluded from the pipeline because they carry insufficient information for reliable matching, while the threshold remains low enough to retain all well-formed references.

Both exported functions rely on a set of internal (non-exported) helper functions that encapsulate specific tasks: format detection via regular-expression matching, string normalisation (case folding, diacritic removal, punctuation standardisation), metadata extraction from each recognised format, ISO 4 journal abbreviation lookup, blocking key construction, pairwise distance matrix computation, hierarchical clustering and dendrogram cutting, post-processing merge-key generation, and canonical representative selection. This modular decomposition permits each component to be tested and profiled in isolation.

Biblioshiny integration

In addition to the programmatic interface, the algorithm is integrated into Biblioshiny, the interactive web application that serves as the graphical front-end for `bibliometrix` (Aria & Cuccurullo, 2026; Aria et al., 2026). Biblioshiny is built with the Shiny framework for R Chang et al. (2024) and allows users without programming experience to perform bibliometric analyses through a browser-based interface.

A dedicated "Reference Matching" module lets users configure the similarity threshold and distance method, execute the full pipeline with a single click, and inspect the results through real-time summary statistics and an interactive cluster-size histogram. Users can also manually refine individual clusters, merging under-matched groups or splitting over-merged ones, before exporting the normalised dataset for further analysis.

Performance considerations

The blocking strategy described in "Phase 4: blocking" section is central to the practical scalability of the implementation. Without blocking, the pairwise comparison step would require $O(n^2)$ string distance evaluations, which becomes prohibitive for corpora exceeding a few thousand references. Blocking reduces this to a sum of within-block quadratic terms, each involving a small number of references (median block size below 5 in practice; fewer than 50 references for over 99% of blocks; see "Results" section).

As an additional safeguard, blocks exceeding a configurable size (`max_block_size`, default 100 unique strings), which may arise when the blocking key is insufficiently discriminative (e.g., for prolific author–year combinations), are handled by a fallback strategy. Rather than computing the full pairwise distance matrix, these large blocks are processed using exact string matching on normalised forms only, forgoing fuzzy matching. This fallback sacrifices recall for such blocks but prevents individual blocks from dominating the total computation time. In practice this safeguard never fires on the real datasets analysed in "Real-data validation: results" section: the largest block observed holds only 44 unique strings, no block reaches 100, and the unique-reference reduction is identical when the cap is raised to 200, 500 or removed altogether. The author+year key spreads even heavily cited works across many small blocks, so the cap rarely binds. We have therefore exposed it as the parameter `max_block_size` rather than a hard-coded constant.

Empirically, the core matching routine processes a corpus of 71,379 cited references in approximately 106 s on a modern desktop computer (Intel Core i7, 16 GB RAM), with the Phase 5 fuzzy matching step accounting for roughly 70% of the total running time. "Experimental design" section presents a detailed scalability analysis.

Experimental design

Assessing a reference matching algorithm requires careful experimental design: evaluation must cover both controlled conditions, where ground truth is known exactly, and real-world data, where noise is unpredictable. The framework described here addresses both dimensions. The evaluation comprises two complementary components: a controlled experiment using synthetic data derived from a gold standard with known ground truth, and a real-data validation on bibliographic records exported from Scopus.

Gold standard construction

Evaluating reference matching algorithms requires a ground truth that maps each reference string to the publication it refers to. Because real-world bibliographic databases do not provide such mappings at scale, and manual annotation is prohibitively expensive for large corpora, we construct a gold standard from which synthetic variants with known identity can be derived.

We retrieved metadata for 1,064 journal articles from WoS using the query `TI=("bibliometrics" OR "science mapping")`, restricted to English-language journal articles. This query targets a well-defined bibliometric community, producing articles whose citation patterns span a broad range of journals, publication years, and formatting conventions. For each article, the following metadata fields were collected:

author list (full names), article title, journal name (both full title and ISO 4 abbreviation as recorded in WoS), publication year, volume, issue, start page, end page, and DOI. Only records with complete metadata across all fields were retained.

Each article was converted into a Scopus-format reference string following the pattern:

```
Authors, Title (Year) Journal, Volume, Issue, pp. Pages
```

Author names were formatted using Scopus conventions (`Surname I.N.`), and the journal name was randomly assigned as either the full title or the ISO 4 abbreviation from the WoS metadata, with approximately equal probability (≈50% each). This random assignment introduces realistic heterogeneity in journal name representation, mirroring the variation encountered when merging records across databases. The resulting 1,064 reference strings, each with a unique article identifier, constitute the gold standard against which all perturbation experiments are evaluated. The gold standard dataset is publicly available on Zenodo (Aria, 2026) to support reproducibility and enable benchmarking of alternative reference matching algorithms.

Perturbation design

The algorithm's blocking strategy partitions references using a key formed from the first-author surname and the publication year ("Phase 4: blocking" section). A central principle of our primary perturbation design (Families A–E) is that *blocking key elements are never perturbed*. By keeping the author surname and the publication year fixed at their gold standard values, every original–variant pair is guaranteed to land in the same block. This design isolates the within-block matching capability, the core of the algorithm, from the blocking mechanism, so that any failure to match can be attributed to the similarity computation rather than to a blocking miss. To assess the complementary failure mode, in which the blocking key itself is corrupted, we additionally introduce Family F ("Blocking-key robustness (family F)" section), whose scenarios perturb the first-author surname and the publication year and therefore drive the original and its variant into different blocks.

Within this constraint, the perturbable fields are: article title (typos, removal), issue number (removal), and page range (removal). We define 17 perturbation scenarios organised into five families, summarised in Table 3. Each scenario applies a specific perturbation to all 1,064 gold standard references, producing 1,064 perturbed variants per scenario.

Family A: Missing metadata (2 scenarios). These scenarios simulate the incomplete metadata fields frequently observed in cited references. Scenario A1 removes the page range. Scenario A2 removes the article title entirely, leaving only author, year, journal, volume, issue, and pages, a pattern commonly found in older WoS-exported references.

Family B: Typographical errors in title (4 scenarios). Character-level errors are introduced into the article title at rates of 1% (B1), 3% (B2), 5% (B3), and 10% (B4) of the title length. Errors are generated by randomly selecting positions and applying one of four operations: character substitution (40% probability), adjacent-character transposition (30%), character deletion (15%), or character insertion (15%). These proportions reflect the empirical distribution of typographic errors in bibliographic data. These proportions were chosen to represent a realistic mix of common typographic error types, with substitutions being the most frequent.

Family C: Formatting noise (2 scenarios). Scenario C1 applies punctuation noise (removal of periods, insertion of extra spaces, removal of commas, replacement of

Table 3 Summary of the 17 perturbation scenarios. Each scenario generates 1,064 perturbed reference strings (one per gold standard article). All scenarios preserve the blocking key (first-author surname and publication year)

ID	Family	Description	Severity
A1	Missing metadata	Remove pages	Low
A2	Missing metadata	Remove title	High
B1	Typographical	Title typos at 1%	Low
B2	Typographical	Title typos at 3%	Medium
B3	Typographical	Title typos at 5%	High
B4	Typographical	Title typos at 10%	High
C1	Formatting	Punctuation noise on full reference	Low
C2	Formatting	Punctuation + case noise on title and journal	Medium
D1	Combined	Journal case + punctuation noise + no issue	Low
E1	Fuzzy stress test	Title typos at 1%	Low
E2	Fuzzy stress test	Title typos at 3%	Low
E3	Fuzzy stress test	Title typos at 5%	Medium
E4	Fuzzy stress test	Title typos at 10%	High
E5	Fuzzy stress test	Title typos at 15%	Extreme
E6	Fuzzy stress test	Title typos at 20%	Extreme
E7	Fuzzy stress test	3% typos + no pages	Medium
E8	Fuzzy stress test	5% typos + no pages + no issue	High

hyphens with en-dashes) to the full reference string. Scenario C2 applies both punctuation and case noise selectively to the title and journal name fields.

Family D: Light combined perturbation (1 scenario). Scenario D1 combines journal case change with punctuation noise on the title and removal of the issue number, representing a mild but realistic combination of multiple perturbation types.

Family E: Fuzzy matching stress tests (8 scenarios). This family isolates the fuzzy string matching component by applying escalating title error rates while preserving all blocking key fields. Scenarios E1–E6 introduce title typos at rates of 1%, 3%, 5%, 10%, 15%, and 20%, respectively. Scenario E7 combines 3% title typos with page removal. Scenario E8 combines 5% title typos with removal of both pages and issue number. The high error rates in E5 and E6 (15% and 20%) deliberately push the algorithm beyond its expected operating range, probing the boundary at which fuzzy matching degrades. Scenarios E1–E4 apply the same error rates as B1–B4 to the same field (article title) but use independent random seeds, thereby serving as independent replications that verify the stability of results across different random draws. Family E extends beyond Family B in two respects: it probes higher error rates (E5–E6: 15%–20%) and introduces combined perturbations (E7–E8) that pair typographical noise with metadata removal.

For each scenario, the 1,064 original gold standard references and the 1064 perturbed variants are combined into a single set of 2128 reference strings. This set is processed by the matching algorithm, which produces a clustering of the input. The ground truth specifies that each article should form a cluster of exactly two members, the original and its perturbed variant, yielding 1064 ground-truth clusters.

Evaluation metrics

We evaluate matching quality using four complementary metrics.

Match rate. The match rate is defined as the fraction of articles whose original and perturbed reference strings are assigned to the same cluster. This metric provides a direct, interpretable measure of the algorithm's ability to recognise perturbed variants. Note that in the synthetic benchmark, where each ground-truth cluster contains exactly two elements, the match rate is numerically equivalent to pairwise recall R; both metrics are reported separately for clarity and to maintain consistency with the real-data evaluation, where clusters may have more than two members.

Pairwise metrics. Given the predicted partition $\mathcal{P}$ and the ground-truth partition $\mathcal{G}$, a true positive (TP) is a pair of references co-clustered in both $\mathcal{P}$ and $\mathcal{G}$; a false positive (FP) is a pair co-clustered in $\mathcal{P}$ but not in $\mathcal{G}$; a false negative (FN) is a pair co-clustered in $\mathcal{G}$ but not in $\mathcal{P}$. Pairwise precision, recall, and F_1-score are defined as

$$P = \frac{TP}{TP + FP}, \qquad R = \frac{TP}{TP + FN}, \qquad F_1 = \frac{2PR}{P + R}. \tag{25}$$

When $P = R = 0$, we define $F_1 = 0$ by convention.

Adjusted Rand Index. The Adjusted Rand Index (ARI) (Hubert and Arabie, 1985) measures cluster-level agreement between the predicted and ground-truth partitions, corrected for chance. The ARI ranges from -1 to 1, with 1 indicating perfect agreement and 0 indicating chance-level performance.

Real-world validation and parameter configuration

The controlled experiment evaluates the algorithm across four similarity thresholds: $\theta \in \{0.80,\ 0.85,\ 0.90,\ 0.95\}$, using the Jaro-Winkler distance (Jaro, 1989; Winkler, 1990) as implemented in the `stringdist` R package. The full experimental grid comprises 17 scenarios $\times$ 4 thresholds = 68 configurations. Both synthetic and real-world experiments use $\theta = 0.90$ as the default configuration.

We complement the controlled evaluation with experiments on two real-world datasets exported from the Scopus database. Both datasets cover the bibliometrics and science-mapping literature but differ in export format and retrieval period, providing complementary test cases for the algorithm.

- **scopus.csv** (legacy format, exported 7 May 2025): retrieved using the Scopus query `TITLE-ABS-KEY("bibliometric*" OR "science map*")`, restricted to the subject category *Information Science & Library Science*, English-language journal articles published between 1985 and 2023. The dataset contains 1,193 articles citing 63,748 unique reference strings. The legacy Scopus format stores cited references as semicolon-delimited strings with abbreviated metadata, which tends to produce a higher rate of reference fragmentation.
- **scopus2.csv** (current format, exported 19 March 2026): the 500 most recent English-language articles with "bibliometrix" in the title. This dataset contains 28,806 unique reference strings. The current Scopus export format provides more structured and standardised reference fields, resulting in less heterogeneity across reference strings. We acknowledge that querying on our own tool's name may select for users of stand-

ardised reference-management workflows, which plausibly contributes to this dataset's lower fragmentation; we therefore treat `scopus2.csv` as a test of the current Scopus export format rather than as an estimate of population-level fragmentation.

Both datasets are processed at $\theta = 0.90$ using Jaro-Winkler distance. For each dataset, we record the reference count reduction, the cluster size distribution, and wall-clock processing time. The real-data evaluation serves as a plausibility check: the algorithm should produce substantial reference consolidation without generating implausibly large clusters, and the processing time should confirm the near-linear scaling predicted by the complexity analysis in "Overall complexity analysis" section.

Results

The experiments span two complementary settings: a controlled synthetic benchmark with known ground truth, and real-world Scopus datasets where noise is unpredictable. Results for both are reported below.

Table 4 Per-scenario evaluation metrics at $\theta = 0.90$ (JW distance) on the synthetic benchmark of 1064 canonical articles

Scenario	Description	Match rate	P	R	F_1	ARI
A: Missing metadata						
A1	No pages	0.995	0.993	0.995	0.994	0.994
A2	No title	0.720	0.990	0.720	0.834	0.833
B: Typographical errors						
B1	1% character error rate	0.993	0.992	0.993	0.993	0.993
B2	3% character error rate	0.983	0.992	0.983	0.988	0.988
B3	5% character error rate	0.976	0.992	0.976	0.984	0.984
B4	10% character error rate	0.946	0.992	0.946	0.969	0.969
C: Formatting changes						
C1	Punctuation variation	0.996	0.998	0.996	0.997	0.997
C2	Punctuation + case variation	0.997	0.996	0.997	0.997	0.997
D: Combined perturbation						
D1	Light combined	0.644	0.986	0.644	0.779	0.779
E: Fuzzy-matching stress tests						
E1	1% fuzzy error	0.991	0.992	0.991	0.992	0.992
E2	3% fuzzy error	0.980	0.992	0.980	0.986	0.986
E3	5% fuzzy error	0.966	0.992	0.966	0.979	0.979
E4	10% fuzzy error	0.955	0.992	0.955	0.973	0.973
E5	15% fuzzy error	0.938	0.992	0.938	0.964	0.964
E6	20% fuzzy error	0.917	0.992	0.917	0.953	0.953
E7	Fuzzy + no pages	0.977	0.992	0.977	0.984	0.984
E8	Fuzzy + minimal metadata	0.965	0.992	0.965	0.979	0.979

All scenarios preserve the blocking key (first-author surname and publication year)

Synthetic benchmark results

Table 4 reports the evaluation metrics for all 17 perturbation scenarios at $\theta = 0.90$. Of the 17 scenarios, 15 achieve $F_1 \geq 0.953$; all maintain precision ≥ 0.986.

Results are stable across all four tested similarity thresholds ($\theta \in \{0.80, 0.85, 0.90, 0.95\}$), with no meaningful variation in any metric.

Fuzzy matching robustness. The E-series and B-series scenarios provide a systematic assessment of matching performance under escalating typographical noise (Fig. 2).

Match rates decrease from 99.1% at 1% character error (E1, $F_1 = 0.992$) to 91.7% at 20% error (E6, $F_1 = 0.953$), with precision constant at 0.992 throughout. The B-series (B1–B4) shows a similar pattern with slightly higher match rates. Combined perturbations E7 (3% typos + no pages) and E8 (5% typos + no pages + no issue) achieve 97.7% and 96.5%, respectively.

Punctuation-invariant matching. Scenarios C1 (punctuation variation) and C2 (punctuation + case variation) achieve match rates of 99.6% and 99.7%, respectively ($F_1 = 0.997$ for both). The combined scenario D1 (journal case change + punctuation noise + issue removal) achieves a match rate of 64.4% ($F_1 = 0.779$, precision = 0.986).

Impact of missing metadata. Scenario A1 (page removal) achieves a match rate of 99.5% ($F_1 = 0.994$). Scenario A2 (title removal) achieves a 72.0% match rate ($F_1 = 0.834$), with precision at 0.990.

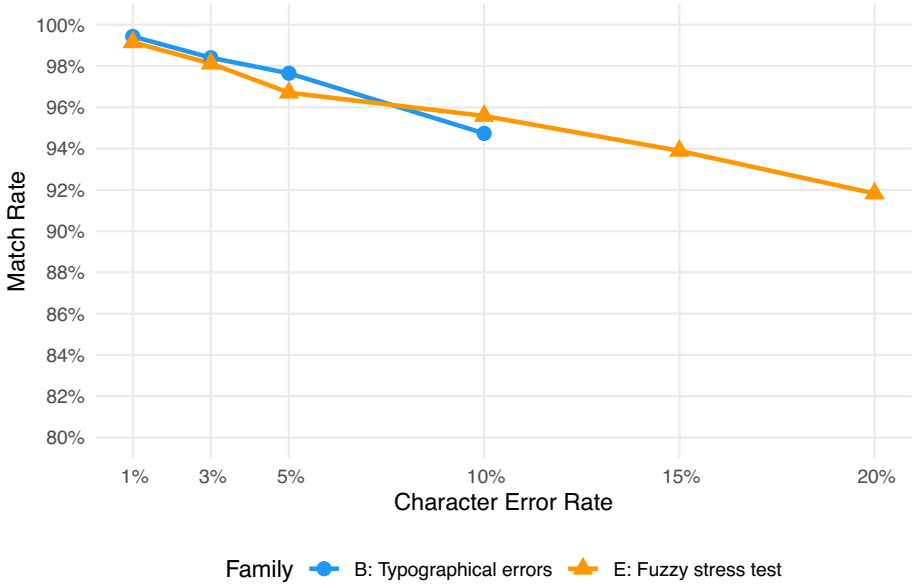

Fig. 2 Match rate as a function of character error rate for the B-series (typographical errors) and E-series (fuzzy stress tests). Both series show gradual degradation from 99% at 1% error to approximately 92% at 20% error, with precision constant at 0.992 throughout

Table 5 Blocking-key corruption (Family F)

ID	Perturbation	Match rate	P	R	F_1	ARI
F1	First-author surname typo	0.015	0.842	0.015	0.030	0.030
F2	Publication year shift (±1)	0.128	0.932	0.128	0.225	0.225
F3	Surname + year	0.000	-	0.000	0.000	0.000

Unlike Families A–E, these scenarios perturb fields that form the blocking key, so original and variant fall into different blocks. $\theta = 0.90$, JW distance, 1,064-article benchmark

Blocking-key robustness (family F)

Families A–E hold the blocking key fixed, so they test only the within-block matching. Family F instead corrupts the key itself (first-author surname and publication year), which sends the two variants of a work into different blocks. Scenario F1 introduces a one- or two-character typo into the surname; F2 shifts the year by ±1, as happens with online-first versus print dates; F3 does both. The results are in Table 5. Recall collapses, because once the variants sit in different blocks they are never compared: a surname typo leaves a match rate of 1.5%, a year shift 12.8%, and the two together 0%. Precision stays high wherever a match survives (0.842 and 0.932), so the failure is one of missed comparisons, not false merges. This is the main weakness of single-pass blocking, and the reason we point to multi-pass and sorted-neighbourhood schemes in "Conclusion" section.

Component ablation

To see which components matter, we disable each phase in turn (using switches added to the implementation) and measure the effect on the synthetic benchmark (scenario B2, 3% title typos; pairwise F_1) and on the real legacy-Scopus corpus (unique-reference reduction). Table 6 reports the results. On the synthetic data the ISO 4 / LTWA step does almost all the work: removing it drops F_1 from 0.988 to 0.218, because the matching string ignores the title and ISO 4 is what aligns the journal forms (which are full or abbreviated at random). Post-processing and fuzzy matching add nothing here, since the exact phase already merges these variants. On the real corpus the gain is split across phases: post-processing accounts for most of it (+2.8 percentage points), ISO 4 for +1.0, and within-block fuzzy matching for +0.2. A DOI-only configuration is uninformative on this material, because Scopus cited references almost never carry a DOI ("Real-data validation: results" section) and so nothing is merged; the useful reading of that result is that the whole reduction is obtained *without* any DOI resolution.

Table 6 Component ablation. Each row removes one component from the full pipeline. Synthetic: pairwise F_1 on B2 (3% typos). Real: unique-reference reduction on `scopus.csv`. $\theta = 0.90$, JW

Configuration	Synthetic F_1	Real reduction (%)
Full pipeline	0.988	11.1
− ISO 4 / LTWA normalisation	0.218	10.08
− Post-processing	0.988	8.34
Exact phases only (− fuzzy, − post-proc.)	0.988	8.13

Table 7 String-distance comparison under aggressive title noise (F_1)

Scenario	JW	Levenshtein	OSA	LCS
B4 (10% typos)	0.969	0.969	0.969	0.969
E5 (15% typos)	0.964	0.964	0.964	0.964
E6 (20% typos)	0.953	0.953	0.953	0.953

The four metrics yield identical accuracy within each scenario. $\theta = 0.90$

Robustness to the string-distance metric

The algorithm supports four string-distance metrics (Jaro-Winkler, Levenshtein, OSA, LCS), but the results so far use only Jaro-Winkler. Table 7 compares all four under the heaviest noise (title typos at 10–20%). Within each scenario they give *identical* match rate, F_1 and ARI, and their runtimes are indistinguishable (4.5–4.7 s). This follows from the pipeline design: the exact phases and the narrow author+year blocking already resolve most references, so fuzzy matching runs only inside very small blocks, where all four metrics make the same merge decisions at $\theta = 0.90$. We therefore soften our earlier wording: Jaro-Winkler is not better than the alternatives, the pipeline is simply *insensitive to the distance metric*. We keep Jaro-Winkler as the default for its prefix weighting and built-in [0, 1] range, at no cost in accuracy.

Real-data validation: results

Before examining matching quality in depth, it is worth noting the computational efficiency of the approach: as shown in Fig. 3, processing time scales approximately linearly with corpus size ($R^2 > 0.999$), remaining under 110 s for collections of up to 71,379 reference occurrences (the full, non-deduplicated `scopus.csv`). On a standard notebook (Intel i7,

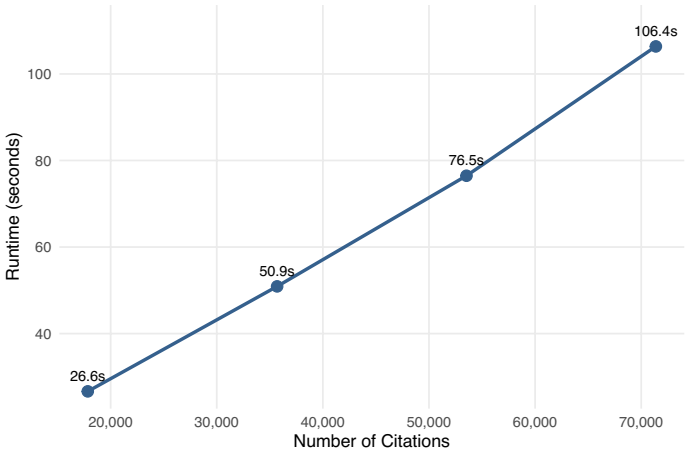

Fig. 3 Wall-clock processing time as a function of dataset size, confirming near-linear scaling

Table 8 Reference matching results on real Scopus datasets ($\theta = 0.90$, JW distance)

Dataset	Articles	Total	Original	Canonical	Reduction
Old format (`scopus.csv`)	1193	63,748	63,214	56,178	11.1%
New format (`scopus2.csv`)	500	28,806	28,584	27,266	4.6%

"Total" is the number of unique cited-reference strings in the raw export; "Original" is the count after excluding strings shorter than 20 characters (`min_chars = 20`); "Canonical" is the count after matching

16 GB RAM), the core matching routine sustains over 670 references per second on parsed inputs (Table 8).

Table 8 reports the results on the two real-world Scopus datasets.

The legacy-format dataset yields an 11.1% reduction in unique cited references (63,214 to 56,178). The current-format dataset shows a smaller 4.6% reduction (28,584 to 27,266), consistent with the more standardised reference strings in recent Scopus exports.

Cluster size distribution. The vast majority of clusters are singletons: 93.9% for the old-format dataset and 97.1% for the new-format dataset. Among non-singleton clusters, the median size is 2 (i.e., a pair of variants), while the 95th percentile reaches 4 variants. The largest cluster in the old-format dataset contains 35 variant strings, all referring to the same heavily cited foundational work in bibliometrics (Small, 1973). This long-tailed distribution is expected: most references appear in only one or two variant forms, but a small number of highly cited works accumulate many variants due to the diversity of journals and databases that cite them.

Qualitative examples. Manual inspection of the largest clusters reveals patterns consistent with known sources of reference fragmentation. For instance, a single seminal article on science mapping appears under variants that differ in journal abbreviation (full title vs. ISO 4 vs. ad-hoc truncation), author initial formatting (with or without periods), and presence or absence of the DOI, all resolved into one canonical cluster by the algorithm. Similarly, clusters of size 3-5 typically consist of variants that mix punctuation styles (e.g., en-dash vs. hyphen in page ranges) or omit the issue number, reflecting the heterogeneity introduced by different citing journals' reference formatting conventions. Concretely, for the three most cited references in `scopus.csv`, Small (1973), van Eck & Waltman (2010), and Kessler (1963), the approach found 35, 18, and 22 distinct variant strings, respectively. Supplementary Table S1 lists verbatim reference strings for representative successful and failed clusters.

Format-dependent differences. The higher reduction rate for the legacy format (11.1% vs. 4.6%) reflects the greater heterogeneity in older Scopus exports, where journal names

Table 9 Data-quality characterisation of the two real datasets

Metric	`scopus.csv`	`scopus2.csv`
Unique cited references	63,748	28,806
% carrying a DOI	0.01	0.02
Mean string length (characters)	166	189
Journal name full / abbreviated (%)	65/34	42/58
Reduction at $\theta = 0.90$ (%)	11.1	4.6

The contrast in journal-name representation, not DOIs, explains the difference in reduction

were less consistently abbreviated and metadata fields were more frequently incomplete. The current-format dataset, with its more standardised reference structure, leaves less room for fragmentation, and correspondingly less room for the matching algorithm to consolidate variants.

Table 9 characterises the two corpora and explains the gap. The legacy export keeps full, free-text journal names (65% full vs. 34% abbreviated), which appear in many slightly different forms, so there is more for the matcher to merge. The current export already uses ISO 4-style abbreviations (58% abbreviated), so little variation is left. DOI coverage is near zero in both formats ($\approx 0.01\%$ and 0.02%); the consolidation therefore comes from the author+year blocking and the ISO 4, exact and fuzzy phases, not from DOIs.

To check accuracy beyond the reduction count, we manually inspected a stratified sample of real clusters, oversampling the largest ones. The Small (1973) cluster merges heavily fragmented references correctly: the 35 variants, for instance, differ only in journal abbreviation, initials, punctuation, and pagination. The main error pattern is *over-merging* of distinct works that share first author, year, journal, and volume but have different titles; it arises because the matching string omits the full title, and shows up most clearly in multi-part series, such as Newman (2001) "Scientific collaboration networks I" and "II", or Braam et al. (1991) "Part I: Structural aspects" and "Part II: Dynamical aspects".

Over-merging in series can be addressed by the algorithm's optional purification step, available from next bibliometrix 5.4.2. For each cluster, the step reads a series marker, a roman numeral, or a "Part *N*" label from each member's title and, using complete linkage, splits the cluster so that members carrying *different* markers are separated, and no incompatible pair remains together. Because the marker is a short, stable token, an original reference and its perturbed variant always share the same one; the step therefore relies on the marker alone rather than on full-title similarity, leaving it essentially inert on the synthetic benchmark while still separating real series.

Effect on downstream bibliometric indicators

The paper argues that reference fragmentation distorts bibliometric structures, so we measure the effect on `scopus.csv` by recomputing the top-cited references and the co-citation network before and after matching (Table 10). These indicators come from the standard co-citation pipeline, which operates on all distinct reference strings and does not apply the `min_chars` validity filter used by the matcher; its base is therefore the raw count of 63, 748 distinct strings rather than the 63, 214 valid strings of Table 8, so the 9.5% reduction reported here and the 11.1% headline reduction are computed on different bases and

Table 10 Co-citation network and top-cited indicators on `scopus.csv`, before vs. after reference matching ($\theta = 0.90$)

Indicator	Before	After	Change
Distinct cited references	63,748	57,692	−9.5%
Co-citation network nodes	63,473	56,969	−6,504
Disconnected components	221	90	−59%
Largest connected component	50,153	52,096	+1,943
Network density	0.00141	0.00172	+21.9%
Most-cited reference (Small 1973)	79	149	+89%
Citations held by top-20	867	1367	+58%

are not directly comparable. Matching cuts the number of distinct references by 9.5% and produces a more connected network: the count of disconnected components drops from 221 to 90 (−59%), the largest component gains 1,943 nodes, and density rises by 21.9%. The effect is largest at the top of the citation distribution. The most-cited reference, Small (1973), goes from 79 to 149 citations once its variants are merged; the top-20 references gain 58% more citations between them; and four meaningless fragment "references" drop out of the pre-matching top-20 altogether. Fragmentation therefore changes both the rankings and the network used in science mapping.

Discussion

The experimental results demonstrate that the reference matching algorithm achieves robust, high-precision performance across a wide range of perturbation conditions. We discuss the key findings, compare the approach with existing methods, and identify directions for improvement.

Robustness of fuzzy matching

The gradual degradation observed across the E-series and B-series (Table 4) can be attributed to two properties of the pipeline design. First, the Jaro-Winkler metric assigns higher weight to prefix agreement (Winkler, 1990), which is particularly well suited to reference strings where the most stable elements (author surname, year) appear at the beginning. Second, the blocking strategy restricts comparisons to references that already share author and year, so that even moderately corrupted strings remain close enough for the hierarchical clustering to recover the correct grouping.

The near-perfect precision (≥ 0.986 in all scenarios, > 0.99 in all but D1) is architecturally guaranteed by this same blocking design: the candidate pool within each block is already highly homogeneous, making false merges between unrelated works extremely unlikely.

From a practical standpoint, empirical studies of citation databases suggest that most real-world typographical errors affect fewer than 5% of characters (Christen, 2012). At that noise level, the algorithm recovers over 96% of reference pairs (scenarios B3, E3), indicating that the pipeline operates well within its comfort zone on typical bibliometric data. The combined-perturbation scenarios (E7, E8) further confirm that multiple simultaneous sources of noise-a common occurrence in real exports-do not cause catastrophic performance loss when the article title remains available.

Threshold stability

The threshold invariance reported in "Synthetic benchmark results" section reveals that, within each block, reference pairs are either clearly similar (accepted at all thresholds) or clearly dissimilar (rejected at all thresholds), with virtually no borderline cases. This bimodal distribution of within-block similarities is a direct consequence of the blocking granularity: blocks defined by author and year are narrow enough that genuine variants cluster tightly, while unrelated references remain far apart. In practical terms, the deterministic phases (exact string and metadata matching) resolve the majority of pairs before the threshold-dependent fuzzy step is reached, further reducing the influence of θ.

This invariance has an important implication for usability: users can adopt the default $\theta = 0.90$ without parameter tuning, a significant advantage over supervised methods that require careful calibration of decision boundaries (Christen, 2012).

Role of metadata fields

The contrast between scenarios A1 and A2 (Table 4) reveals a clear hierarchy among metadata fields. Page numbers are largely redundant for matching purposes: the title and journal name carry the dominant discriminative signal within each block. This finding is consistent with the observation that older WoS records frequently lack pagination yet remain matchable. Conversely, the title is the single most informative field; its removal (A2) forces the algorithm to rely on volume, issue, and pages alone, which are often shared by multiple articles in the same journal–year block. That precision remains high even in A2 confirms the conservative design: the algorithm prefers missed matches over false merges, a desirable property for bibliometric indicators where an erroneous merge can artificially inflate citation counts.

The near-perfect scores on C1 and C2 validate the punctuation-invariant comparison introduced in Phase 3. Punctuation and case variation are among the most frequent sources of superficial heterogeneity in real-world reference exports (Elmagarmid et al., 2007); the stripped alphanumeric representation neutralises them deterministically, before the costlier fuzzy-matching step is invoked. The lower performance of D1 illustrates the limit of this approach: when journal-name case changes interact with the ISO 4 normalisation pipeline, a fraction of references is routed to a slightly different normalised form, escaping the exact-match phase and falling outside the fuzzy-matching threshold.

Role of ISO 4 journal normalisation

The LTWA-based journal normalisation (ISSN International Centre, 2024) plays a critical role in real-world settings where the same journal appears under full names, ISO 4 abbreviations, and ad hoc truncations. By mapping diverse journal-name forms to a single canonical abbreviation before blocking (Phase 2), the normalisation step ensures that references referring to the same journal are routed into the same block and thus become candidates for comparison.

To our knowledge, this is the first integration of the LTWA in a reference matching context; our implementation uses the approximately 14,000 English-language entries from the full 56,000-entry multilingual resource. Prior approaches have either ignored journal-name variation, relied on small hand-curated lookup tables, or delegated the problem to external services such as Crossref (Hendricks et al., 2020).

Comparison with existing approaches

Our algorithm occupies a distinctive position among reference matching methods. Unlike supervised approaches (Pasula et al., 2002; Daumé & Marcu, 2005), our pipeline requires no labelled training data and can be applied to any bibliometric dataset without domain-specific configuration. Unlike the Crossref reference-matching service (Hendricks et al., 2020), our algorithm operates entirely offline without rate limits or network dependencies. Compared with deduplication tools such as Hair et al. (2023) and the recent multi-database

merging tool of Nikolić et al. (2024), which target well-structured source records with full titles and complete metadata, our algorithm handles the abbreviated, heterogeneous cited-reference strings exported by WoS and Scopus. The preprocessing framework of Nowakowska (Nowakowska, 2025) and the reference-coverage analysis of Culbert et al. (2025) further underscore the need for robust within-dataset normalisation that does not rely on external identifier resolution alone.

Recent work applies large language models (LLMs) to entity matching and data wrangling (Narayan et al., 2022; Peeters & Bizer, 2023). These methods can resolve semantically hard cases with little or no task-specific training, and they complement rather than replace the deterministic approach used here. They cost money per call, are non-deterministic (a problem for citation indicators that have to be auditable and reproducible), do not scale easily to the tens of thousands of references in a corpus, and raise data-governance questions when reference data cannot be sent to a third-party service. Our pipeline is free, deterministic and reproducible, and runs locally at hundreds of references per second. Using an LLM only as a fallback for the few hard cases that remain would be a sensible extension.

The F_1 scores of 0.953-−0.997 achieved across the perturbation scenarios compare favourably with those reported for supervised methods on curated datasets, and the threshold-invariant behaviour eliminates parameter tuning as a practical concern.

Practical implications and limitations

The real-data experiments confirm that reference matching yields meaningful improvements on actual bibliometric datasets. The old-format Scopus dataset shows an 11.1% reduction in unique cited references, while the newer, more standardised format shows a 4.6% reduction. These reductions correct citation counts that would otherwise be fragmented across variant strings, with downstream effects on co-citation networks (Small, 1973), bibliographic coupling strengths, conceptual structure analysis (Aria et al., 2024), and citation-based bibliometric indicators more broadly (e.g., *h*-index, *g*-index, citation counts) (Harzing & Alakangas, 2016).

Near-linear computational scaling, 71,379 references processed in approximately 106 s, makes the algorithm suitable for interactive use. The implementation within `biblio-metrix`, with its interactive Biblioshiny GUI, ensures broad accessibility for researchers across disciplines.

Despite these strengths, several limitations should be acknowledged. First, the current LTWA integration focuses primarily on English-language title words; extending it to exploit the full multilingual abbreviation set would improve normalisation quality for non-English journals. Books, conference proceedings and other non-article references also lack the journal, volume and page structure that the blocking and merge keys need, so they are handled only by the exact and punctuation-invariant phases and are not consolidated by fuzzy matching. Second, the `max_block_size` fallback forgoes fuzzy matching for very large blocks; in practice it is never triggered on our data ("Performance considerations" section), but it remains a theoretical limit for pathologically large author-year blocks. Third, the synthetic evaluation applies perturbations independently and uniformly; real reference errors may exhibit correlated patterns that the benchmark does not fully capture. The synthetic benchmark should also be read as an upper bound on real-world performance: (i) its source records have complete metadata, whereas real cited references, especially older ones, are often incomplete; (ii) its journal names vary in only two ways

(full title or the LTWA-compatible ISO 4 form) and never show the ad-hoc truncations of real exports; and (iii) its perturbations are applied independently and uniformly. Fourth, because the matching string omits the article title for robustness to typos, the pipeline can over-merge distinct works that share first author, year, journal and volume but differ only in a non-series title and have no usable pages. The purification step will remove the multi-part cases; the rest cannot be separated by the title without giving up robustness to title noise. This is a trade-off built into the title-independent design, not an implementation gap. Fifth, the experiments cover only WoS and Scopus format reference strings, the two databases that export cited references as free text. Identifier-based providers (OpenAlex, Dimensions, Lens, PubMed) give cited references as resolved record identifiers, which need no string matching at all. Handling further free-text formats would mean adding rules to the format detection of Eq. (1).

Conclusion

This paper has presented an unsupervised seven-phase algorithm for reference matching in bibliometric datasets, addressing the pervasive problem of reference-string fragmentation caused by variation in journal-name formatting, metadata completeness, typographical noise, and punctuation conventions.

The evaluation on a synthetic benchmark of 1064 articles with 17 controlled perturbation scenarios demonstrates that the algorithm achieves match rates between 92% and 99% for typographical noise levels ranging from 1% to 20% of characters, with pairwise precision above 0.99 in all but the most aggressive combined perturbation scenario (D1, precision = 0.986). Missing page information has negligible impact on performance (match rate 99.5%), and even the complete removal of the article title still yields a 72.0% match rate with no false merges. Results are stable across all tested similarity thresholds ($\theta \in \{0.80, 0.85, 0.90, 0.95\}$), confirming that the default configuration ($\theta = 0.90$, Jaro-Winkler distance) is adequate without user adjustment. On real-world Scopus data, the algorithm reduces unique cited references by 11.1% for legacy-format exports and 4.6% for current-format exports, with near-linear computational scaling (71,379 references processed in approximately 106 s).

Three aspects of this work represent novel contributions to the field. First, the integration of the LTWA for ISO 4 journal-name normalisation is, to our knowledge, the first application of this resource in a reference-matching context, and it eliminates one of the largest sources of reference fragmentation. Second, the format-aware processing module enables seamless handling of references from different database generations and providers. Third, the comprehensive evaluation framework, which systematically varies perturbation type and severity while controlling the blocking key, provides a rigorous methodology for benchmarking reference matching algorithms.

The practical impact of these findings is amplified by the algorithm's integration into `bibliometrix` and its interactive companion `Biblioshiny`: researchers can apply robust reference matching directly within their existing scientometric workflow, inspecting and validating normalised references through a point-and-click interface without requiring programming expertise.

More broadly, the validation of our reference matching algorithm represents a necessary step toward a larger goal: building a robust framework for merging bibliographic collections from multiple databases, a problem that remains open in the field of

scientometrics. Currently, `bibliometrix` and Biblioshiny already support the merging of collections from different sources (WoS, Scopus, OpenAlex (Aria et al., 2024), Dimensions, and others), but the merge procedure excludes fields that are not standardised in the same way across databases, most notably cited references, which each provider records in a different format. By enabling the normalisation and matching of heterogeneous reference strings, the algorithm presented here removes this barrier and opens the way to fully integrated multi-source bibliometric analyses in which citation-based indicators and network structures are computed on a unified, deduplicated reference corpus. The order of operations matters here. Reference matching is applied *within* each single-format collection (a WoS collection and a Scopus collection separately), and the collections are merged only afterwards. Matching is the step that has to come first before references can be unified across databases, which is why the current `mergeDbSources` routine sets cited references aside when it combines collections of different formats. Running the matcher on an already-merged, mixed-format collection is outside its present scope.

Future work should address several additional directions. Expanding the LTWA integration to support multilingual abbreviation rules would improve normalisation for non-English journals. The single-pass blocking key is the algorithm's main weakness: as the Family F experiment shows ("Blocking-key robustness (family F)" section), corrupting the first-author surname or the publication year drives the two variants of a work into different blocks, where they are never compared, producing irrecoverable false negatives. The natural remedy is to move beyond a single key. Multi-pass blocking (Baxter et al., 2003), which indexes each reference under several complementary keys and unions the resulting candidate pairs, and the sorted-neighbourhood method (Hernandez & Stolfo, 1998), which slides a window over a sorted ordering of the references, both tolerate an error in any one blocking field and are the most promising next step for recovering these missed matches. Adaptive blocking strategies that adjust key components based on data characteristics could further improve recall without sacrificing precision. Deep learning approaches for learning contextual string representations offer a promising avenue for handling heavily corrupted references.

Supplementary Information The online version contains supplementary material available at https://doi.org/10.1007/s11192-026-05763-2.

Author contributions Conceptualisation of the work: M. Aria, L. D'Aniello. Data collection: M. Aria. Data analysis and interpretation: M. Aria, L. D'Aniello, M. Spano. Original draft preparation: M. Aria, L. D'Aniello. Review and editing: M. Aria, L. D'Aniello, M. Spano. All authors have read and agreed to the published version of the manuscript.

Funding Open access funding provided by Università degli Studi di Napoli Federico II within the CRUI-CARE Agreement.

Data availability The gold standard benchmark dataset used for the synthetic evaluation is publicly available on Zenodo (Aria, 2026; https://doi.org/10.5281/zenodo.19471288). The two real-world Scopus datasets (scopus.csv and scopus2.csv) cannot be publicly shared due to Elsevier copyright restrictions on redistributing Scopus-exported bibliographic data; they are available from the corresponding author on reasonable request. The algorithm is implemented in the open-source bibliometrix R package, freely available on CRAN.

Declarations

Conflict of interest The authors declare no conflict of interest.

References

Aria, M. (2026). Gold standard benchmark dataset for bibliometric reference matching evaluation. *Zenodo Data set*. https://doi.org/10.5281/zenodo.19471288

Aria, M., & Cuccurullo, C. (2017). bibliometrix: An R-tool for comprehensive science mapping analysis. *Journal of Informetrics, 11*(4), 959–975. https://doi.org/10.1016/j.joi.2017.08.007

Aria, M., & Cuccurullo, C. (2026). *Science Mapping Analysis—A Primer with Biblioshiny*. McGraw-Hill.

Aria, M., Cuccurullo, C., D'Aniello, L., Misuraca, M., & Spano, M. (2024). Comparative science mapping: A novel conceptual structure analysis with metadata. *Scientometrics, 129*(11), 7055–7081. https://doi.org/10.1007/s11192-024-05161-6

Aria, M., Cuccurullo, C., D'Aniello, L., & Spano, M. (2026). Biblioshiny and the SAAS workflow: An integrated framework for transparent and reproducible science mapping a demonstration through the replication of a study. *Journal of Informetrics,20*(3), Article 101837. https://doi.org/10.1016/j.joi.2026.101837

Aria, M., Le, T., Cuccurullo, C., Belfiore, A., Choe, J. (2024). openalexR: An R-tool for collecting bibliometric data from OpenAlex. The R Journal. 16(2), 167–180. https://doi.org/10.32614/RJ-2023-089

Baxter, R., Christen, P., & Churches, T. (2003). A comparison of fast blocking methods for record linkage. *Proceedings of the ACM SIGKDD Workshop on Data Cleaning, Record Linkage, and Object Consolidation* (pp. 25–27)

Chang, W., Cheng, J., Allaire, J.J., Sievert, C., Schloerke, B., Xie, Y., Allen, J., McPherson, J., Dipert, A., Borges, B. (2024). Shiny: Web Application Framework for R. R package version 1.9.1. https://CRAN.R-project.org/package=shiny

Christen, P. (2012). *Data matching: Concepts and techniques for record linkage, entity resolution, and duplicate detection*. Springer.

Cohen, W. W., Ravikumar, P., & Fienberg, S. E. (2003). A comparison of string distance metrics for name-matching tasks. *Proceedings of the IJCAI-03 Workshop on Information Integration on the Web (IIWeb-03)* (pp. 73–78)

Culbert, J. H., Hobert, A., Jahn, N., Haupka, N., Schmidt, M., Donner, P., & Mayr, P. (2025). Reference coverage analysis of OpenAlex compared to Web of Science and Scopus. *Scientometrics, 130*(4), 2475–2492.

Damerau, F. J. (1964). A technique for computer detection and correction of spelling errors. *Communications of the ACM, 7*(3), 171–176. https://doi.org/10.1145/363958.363994

Daumé, H., III., & Marcu, D. (2005). A Bayesian model for supervised clustering with the Dirichlet process mixture. *Journal of Machine Learning Research, 6*, 1551–1577.

Elmagarmid, A. K., Ipeirotis, P. G., & Verykios, V. S. (2007). Duplicate record detection: A survey. *IEEE Transactions on Knowledge and Data Engineering, 19*(1), 1–16. https://doi.org/10.1109/TKDE.2007.250581

Fellegi, I. P., & Sunter, A. B. (1969). A theory for record linkage. *Journal of American Statistical Association, 64*(328), 1183–1210. https://doi.org/10.1080/01621459.1969.10501049

Foufoulas, Y., Stamatogiannakis, L., Dimitropoulos, H., & Ioannidis, Y. (2017). High-pass text filtering for citation matching. *International Conference on Theory and Practice of Digital Libraries* (pp. 355–366). Springer.

Giles, C.L., Bollacker, K.D., & Lawrence, S. (1998). CiteSeer: An automatic citation indexing system. In: Proceedings of the Third ACM Conference on Digital Libraries, pp. 89–98. https://doi.org/10.1145/276675.276685

Hair, K., Bahor, Z., Macleod, M., Liao, J., & Sena, E. S. (2023). The automated systematic search deduplicator (ASySD): A rapid, open-source, interoperable tool to remove duplicate citations in biomedical systematic reviews. *BMC Biology, 21*(1), 189.

Harzing, A.-W., & Alakangas, S. (2016). Google Scholar, Scopus and the Web of Science: A longitudinal and cross-disciplinary comparison. *Scientometrics, 106*(2), 787–804. https://doi.org/10.1007/s11192-015-1798-9

Hendricks, G., Tkaczyk, D., Lin, J., & Feeney, P. (2020). Crossref: The sustainable source of community-owned scholarly metadata. *Quantitative Science Studies, 1*(1), 414–427. https://doi.org/10.1162/qss_a_00022

Hernandez, M. A., & Stolfo, S. J. (1998). Real-world data is dirty: Data cleansing and the merge/purge problem. *Data Mining and Knowledge Discovery, 2*(1), 9–37. https://doi.org/10.1023/A:1009761603038

Hirai, K., Kanazawa, T., & Hayashi, T. (2025). A lexically-driven adaptive validation framework for search-based reference matching. *2025 ACM/IEEE Joint Conference on Digital Libraries (JCDL)* (pp. 320–323). IEEE.

Hubert, L., & Arabie, P. (1985). Comparing partitions. *Journal of Classification, 2*(1), 193–218. https://doi.org/10.1007/BF01908075

ISSN International Centre. (2024). List of title word abbreviations (LTWA). Retrieved June 15, 2025, from https://portal.issn.org/ltwa

International Organization for Standardization. (1997). ISO 4:1997 - Information and documentation - Rules for the abbreviation of title words and titles of publications. *International Standard. Revised by ISO,4*, 2024.

Jaro, M. A. (1989). Advances in record-linkage methodology as applied to matching the 1985 census of Tampa, Florida. *Journal of American Statistical Association, 84*(406), 414–420.

Kessler, M. M. (1963). Bibliographic coupling between scientific papers. *American Documentation, 14*(1), 10–25.

Lawrence, S., Giles, C. L., & Bollacker, K. (1999). Digital libraries and autonomous citation indexing. *IEEE Computer, 32*, 67–71. https://doi.org/10.1109/2.769447

Levenshtein, V. I. (1966). Binary codes capable of correcting deletions, insertions, and reversals. *Soviet Physics Doklady, 10*(8), 707–710.

McCallum, A., Nigam, K., & Ungar, L.H. (2000). Efficient clustering of high-dimensional data sets with application to reference matching. In: Proceedings of the Sixth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '00), pp. 169–178. https://doi.org/10.1145/347090.347123

Narayan, A., Chami, I., Orr, L., Ré, C. (2022). Can foundation models wrangle your data? Proceedings of the VLDB Endowment. 16(4), 738–746. https://doi.org/10.14778/3574245.3574258

Newcombe, H. B., Kennedy, J. M., Axford, S. J., & James, A. P. (1959). Automatic linkage of vital records. *Science, 130*(3381), 954–959. https://doi.org/10.1126/science.130.3381.954

Nikolić, D., Ivanović, D., & Ivanović, L. (2024). An open-source tool for merging data from multiple citation databases. *Scientometrics, 129*(7), 4573–4595.

Nowakowska, M. (2025). A comprehensive approach to preprocessing data for bibliometric analysis. *Scientometrics, 130*(9), 5191–5225.

Pasula, H., Marthi, B., Milch, B., Russell, S. J., & Shpitser, I. (2002). Identity uncertainty and citation matching. Advances in Neural Information Processing Systems, p. 15

Peeters, R., & Bizer, C. (2023). Using ChatGPT for entity matching. In: New Trends in Database and Information Systems (ADBIS 2023). Communications in Computer and Information Science, vol. 1850, pp. 221–230. Springer, Cham. https://doi.org/10.1007/978-3-031-42941-5_20

Piskorski, J., & Sydow, M. (2007). Usability of string distance metrics for name matching tasks in multilingual settings. *Roczniki Kolegium Analiz Ekonomicznych, 18*, 121–143.

Priem, J., Piwowar, H., & Orr, R. (2022). OpenAlex: A fully-open index of scholarly works, authors, venues, institutions, and concepts. arXiv preprint arXiv:2205.01833

R Core Team. (2024). R: A Language and Environment for Statistical Computing. R Foundation for Statistical Computing, Vienna, Austria. R Foundation for Statistical Computing. https://www.R-project.org/

Small, H. (1973). Co-citation in the scientific literature: A new measure of the relationship between two documents. *Journal of the American Society for Information Science, 24*(4), 265–269. https://doi.org/10.1002/asi.4630240406

Van Eck, N., & Waltman, L. (2010). Software survey: VOSviewer, a computer program for bibliometric mapping. *Scientometrics, 84*(2), 523–538.

Visser, M., Eck, N. J., & Waltman, L. (2021). Large-scale comparison of bibliographic data sources: Scopus, Web of Science, Dimensions, Crossref, and Microsoft Academic. *Quantitative Science Studies, 2*(1), 20–41.

Wagner, R. A., & Fischer, M. J. (1974). The string-to-string correction problem. *Journal of the ACM, 21*(1), 168–173. https://doi.org/10.1145/321796.321811

Winkler, W.E. (1990). String comparator metrics and enhanced decision rules in the Fellegi–Sunter model of record linkage. Technical report, U.S. Bureau of the Census, Statistical Research Division. Research Report RR99/04

Authors and Affiliations

Massimo Aria[1,2] · Luca D'Aniello[1,2] · Maria Spano[1,2]

✉ Massimo Aria
massimo.aria@unina.it

Luca D'Aniello
luca.daniello@unina.it

Maria Spano
maria.spano@unina.it

[1] Department of Economics and Statistics, University of Naples Federico II, Via Cinthia 26, Monte S. Angelo, Naples 80126, Italy

[2] K-Synth srl, Academic Spin-Off of University of Naples Federico II, Via Cinthia 26, Monte S. Angelo, Naples 80126, Italy