The Software Mention Detection and Coreference Resolution Shared Task 2026

Sharmila Upadhyaya[1], Wolfgang Otto[1,2], Julia Matela[3]
Frank Krüger[3], Stefan Dietze[1,2]
[1]GESIS – Leibniz Institute for the Social Sciences, Germany
[2]Heinrich Heine University Düsseldorf, Germany
[3]Wismar University of Applied Sciences, Germany
{sharmila.upadhyaya, wolfgang.otto, stefan.dietze}@gesis.org
{julia.matela, frank.krueger}@hs-wismar.de

Abstract

Software is referenced in research papers in many different ways: full names, abbreviations, misspellings, versioned names, or indirect references via websites and citations. This makes it hard to link mentions to a single software entity, which in turn limits large-scale analyses and knowledge graph construction. The Software Mention Detection and Coreference Resolution (SOMD) shared task 2026, organized at the Natural Scientific Language Processing (NSLP) workshop at LREC 2026, focuses on clustering software mentions that refer to the same software entity. We provide three subtasks covering gold mentions, automatically extracted mentions, and mentions sampled at scale from large-scale publications. Systems are evaluated with established coreference metrics (MUC, B^3, CEAFe) and their CoNLL average. This paper describes the task setup, datasets, evaluation, baseline, and the observed patterns in participant submissions, and outlines future directions for scalable software mention coreference resolution. The shared task was concluded with total five registered participants, with total 43 submissions for all subtasks. Finally, two system papers were submitted with competitive performance against baselines.

Keywords: software mentions, cross-document coreference, scholarly NLP, knowledge graphs, shared task

1. Introduction

Software has emerged as a tool in scientific research, yet it is often cited and described informally in scholarly articles. A single software can appear under multiple surface forms (e.g. different spellings or abbreviations), and a single surface form can refer to other software packages depending on context. As a result, building reliable links from text mentions to software entities remains difficult, especially at the scale of web or biomedical corpora. The Software Mention Detection and Coreference Resolution (SOMD) shared task 2026 targets this problem by posing a challenge that requires grouping (clustering) software mentions referring to the same underlying software entity across multiple documents. The task is widely studied under the framework of cross-document coreference resolution in the Information Extraction community. SOMD 2026 is hosted at the Natural Scientific Language Processing (NSLP) workshop at the Language Resources and Evaluation Conference (LREC) 2026 (NFDI4DataScience, 2026a,b). The shared task follows the earlier SOMD editions described in Krüger et al. 2024 and Upadhyaya et al. 2025, which focused on software-related information extraction. This iteration adds a dedicated evaluation setting for scalable cross-document coreference resolution (Keshtkaran et al., 2017).

Cross-document coreference resolution of software mentions supports downstream tasks, such as Knowledge Graph (KG) construction, in which software entities are linked to publications, authors, datasets, and research outcomes. SoftwareKG is a prominent example of a large-scale knowledge graph that represents software mentions extracted from a large corpus and supports analyses of software use and citation practices (Schindler et al., 2022; GESIS, 2026). Moreover, the quality of coreference resolution directly affects KG quality; collapsing either distinct software into a single node or splitting a single software into multiple nodes, both of which distort statistics and graph queries. Similarly, reliable cross-document coreference resolution supports reproducibility by consolidating mentions of the same underlying software entity. This enables subsequent linking to stable software identifiers (e.g., repository URLs, DOIs, or registry identifiers). This helps researchers identify the exact software used in a study and supports indexing in discovery services. Recent work on software citation and discoverability highlights persistent gaps between recommended citation practices and real-world referencing in scholarly articles (Katz and Chue Hong, 2024). Finally, resolving software mentions across documents enables analytics at scale, such as tracking software adoption across domains, identifying tool dependencies, and studying the impact of software releases. These use cases motivate shared, open benchmarks that combine high-

Proceedings of the 3rd Int. Workshop on Natural Scientific Language Processing (NSLP 2026) @ LREC 2026, pages 247–254

quality gold annotations with realistic noisy inputs.

The first and second SOMD shared tasks addressed software-related information extraction from scholarly publications. SOMD 2024 (Krüger et al., 2024) introduced a benchmark for detecting software mentions and related information in context and reported substantial variation across systems and subtasks. SOMD 2025 (Upadhyaya et al., 2025) extended the scope and introduced a challenging setting for software-related information extraction, while continuing to build community baselines and evaluation practices. Building on last year's challenge, SOMD 2026 focuses on the consecutive steps of the information extraction pipeline: cross-document coreference resolution. Earlier tasks treat mentions as local objects (within a document or sentence). However, for KG construction and large-scale analytics, mentions must be grouped across documents. SOMD 2026, therefore, frames coreference resolution as a cross-document coreference clustering problem over software mentions, using established coreference evaluation metrics.

Cross-document coreference resolution faces challenges such as surface-form variation (e.g., spelling differences and abbreviations), ambiguity in short names (e.g., "R", "SAS"), sparse metadata, and domain shifts across disciplines (Schindler et al., 2021). One of the central constraints for large-scale coreference resolution is scalability, as naive mention-pair approaches have $O(n^2)$ complexity and become infeasible at the corpus scale. Therefore, high-accuracy models based on cross-encoders or large language models are computationally expensive and difficult to deploy for large knowledge graphs. SOMD 2026 thus emphasizes search space reduction, efficient similarity search (e.g., approximate nearest neighbors), and the use of available metadata, while evaluating systems under both clean and noisy input conditions.

SOMD 2026 makes the following contributions:

- **A shared evaluation setting** for cross-document software mention coreference resolution.
- **Three subtasks** covering gold mentions from SoMeSci (Schindler et al., 2021), automatically extracted mentions, and large-scale samples from SoftwareKG (NFDI4DataScience, 2026b).
- **New and extended coreference annotations** derived from SoMeSci and SoftwareKG (Schindler et al., 2021, 2022).
- **A TF–IDF + DBSCAN baseline** (Salton and Buckley, 1988; Ester et al., 1996).
- **Baseline II (Semantic Centroids with Hierarchi- cal Density-Based Clustering.)** proposing blocking as scalable cross document coreference resolution approach.
- **Standard evaluation** using MUC, B^3, CEAFe, and the CoNLL average (Vilain et al., 1995; Bagga and Baldwin, 1998; Luo, 2005; Pradhan et al., 2012).

This paper summarises the challenge (January 20 to February 20, 2026), describing the subtasks, datasets, evaluation setup, baseline, and participating systems, and discusses scalability as the central challenge. The code to reproduce both baselines is publicly available online.[1][2]

2. Task Description

The shared task focuses on cross-document coreference resolution of software mentions in scholarly articles, formulated as a clustering problem.

Subtask 1: Coreference Resolution over Gold Mentions Given a set of gold-standard software mentions across multiple documents, systems must cluster mentions that refer to the same underlying software entity. Input is a list of software mentions with (i) mention string, (ii) sentence-level context, and (iii) optional metadata (e.g., URL, developer/author, references). Expected output is mapping from `mention_id` to `cluster_id`.

Subtask 2: Coreference Resolution over Predicted Mentions Given automatically extracted software mentions across multiple documents, systems must cluster mentions referring to the same software entity under realistic extraction noise. Input includes automatically detected mentions in the same format as Subtask 1. Output is the same as Subtask 1, i.e., a mapping from `mention_id` to `cluster_id`. Evaluation is performed on mentions that can be aligned to gold cluster annotations.

Subtask 3: Coreference Resolution at Scale Given an extensive collection of automatically extracted software mentions sampled at scale, systems must efficiently perform cross-document clustering. Input consists of automatically extracted mentions at a large scale, and output is a mapping from `mention_id` to `cluster_id`, evaluated on an annotated subset.

3. Dataset

SOMD 2026 builds directly on the datasets and annotations developed in previous SOMD shared

[1] Baseline I (TF–IDF + DBSCAN): https://github.com/sarmilaupadhyaya/SOMD2026

[2] Baseline II (Semantic Centroids with Hierarchi- cal Density-Based Clustering.): https://github.com/matjulia/somd2026

tasks, which focused on software mention detection and related attribute extraction. Consequently, we reuse already annotated or automatically predicted software entities and their associated metadata as input for the present task. Depending on the subtask, these mentions are either gold-standard annotations from SoMeSci (Schindler et al., 2021) (Subtask 1) or predicted outputs from prior extraction models (Subtasks 2 and 3) from SoftwareKG (Schindler et al., 2022). The current shared task addresses the subsequent stage of the pipeline, namely, cross-document coreference resolution over these pre-identified mentions.

3.1. Background: SoMeSci and SoftwareKG Datasets

SoMeSci is a gold standard dataset of software mentions in scientific articles, including disambiguation of spelling variants and additional information such as version, developer, URL, and citations (Schindler et al., 2021). It provides a high-quality starting point for defining mention-level clusters. SoftwareKG is a large-scale knowledge graph of software mentions extracted from PubMed Central articles, enabling analyses of software usage and citation practices (Schindler et al., 2022; GESIS, 2026). It serves as the basis for the scale-oriented sampling in Subtask 3.

3.2. Dataset Creation for SOMD 2026

Building on SoMeSci and SoftwareKG, we construct three subtask-specific datasets that differ in input quality and scale.

Subtask 1 For Subtask 1, we use the manually annotated SoMeSci (Schindler et al., 2023) gold-standard mentions and convert them into a unified mention-list format. Each mention is represented with its surface form, sentence-level context, and available metadata (e.g., developer, URL, citation markers). Cross-document coreference clusters are derived from the canonical software identifiers in SoMeSci. The dataset is split into 80% for training and 20% for testing.

Subtask 2 For Subask 2, we use a subset of automatically extracted mentions from SoftwareKG dataset (Schindler et al., 2022) consisting of mentions extracted from the same document collection as in SoMeSci (Schindler et al., 2023). These predicted mentions are converted into the unified list format using the same logic as in Subtask 1. To enable evaluation, we align predicted mentions with existing gold clusters whenever possible and identify *linkable* mentions. The linkable dataset is split into 80% training and 20% test data. The test set is extended by remaining mentions from the same document collection and 2,000 manually annotated mentions from SoftwareKG and, and evaluation is performed on the subset of the test split consisting of linkable or manually annotated mentions.

Subtask 3 For Subtask 3, we apply the SoMeSci extraction model to a larger subset of PubMed Central articles using the SoftwareKG pipeline. From the resulting large pool of predicted mentions, we sample mentions at scale. A subgroup is manually annotated with cross-document coreference clusters for evaluation, while the remaining mentions are used to assess scalability and runtime behavior.

Table 1 provides an overview of the dataset composition across all subtasks and splits, including the number of documents, software mentions, and cross-document coreference clusters. The test splits clusters are withheld and used for the ranking of the leaderboard on CodaBench. This overview illustrates the differences in scale and annotation type across Subtasks 1–3.

The datasets, baseline code, and evaluation scripts are released via the shared task website (NFDI4DataScience, 2026b).

4. Evaluation Metrics

We evaluate systems using standard coreference metrics:

- **MUC** measures how many links between mentions are needed to transform predicted clusters into gold clusters and tends to reward systems that produce fewer, larger clusters (Vilain et al., 1995).
- **B^3** computes precision and recall per mention based on overlap between predicted and gold clusters and is sensitive to both over-merging and over-splitting (Bagga and Baldwin, 1998).
- **CEAFe** aligns predicted clusters to gold clusters in a one-to-one fashion and scores based on the best alignment, penalizing both fragmentation and merging in an entity-centric way (Luo, 2005).

Following common practice in shared tasks, we also report the **CoNLL F1** score as the unweighted average of the F1 values of MUC, B^3, and CEAFe (Pradhan et al., 2012). Using multiple metrics is important because each metric emphasizes different error modes: MUC can mask over-merging, while CEAFe and B^3 react more strongly to cluster purity and fragmentation.

5. Participation and Approaches

A total of 5 participants registered for the SOMD 2026 shared task. A team fully participated by sub-

	Train			Test		Eval	
Subtask	Docs	Mentions	Clusters	Docs	Mentions	Mentions	Clusters
Subtask 1 (Gold Standard)	973	2,974	733	244	743	743	250
Subtask 2 (Predicted)	967	2,860	699	1,977	12,516	2,697	719
Subtask 3 (Predicted at Scale)	967	2,860	699	173,450	219,950	10,454	1,893

Table 1: Dataset statistics across Train and Test splits for three subtasks. Subtasks 2 and Subtask 3 share identical training sets. Subtask 3 incorporates an extended test set focusing on large scale analysis. The Eval set is the subset of the Test set used for system scoring. Consequently, Subtask 1 has identical Test and Eval splits.

mitting results for all three subtasks, while one team participated in only two subtasks. All three teams provided a system description. In this section, we introduce the three final approaches, along with a baseline model.

5.1. Baseline Systems

Baseline I: TF–IDF + DBSCAN.
We provide a baseline that clusters mentions using vector-space similarity and density-based clustering. Each mention is represented by a TF–IDF vector over character and word n-grams derived from (i) the mention surface form and (ii) a short context window (Salton and Buckley, 1988). Optional metadata fields (URL host, developer string) are appended as tokens when present. We apply DBSCAN with cosine distance on TF–IDF vectors (Ester et al., 1996). DBSCAN does not require the number of clusters as input, which fits open-world disambiguation. Hyperparameters (`eps`, `min_samples`) are tuned on the development split.

Baseline II: Semantic Centroids with Hierarchical Density-Based Clustering.
The second baseline (Matela and Krüger, 2026) is a hybrid framework based on Sentence-BERT embeddings, combining FAISS-based retrieval over training-set cluster centroids with HDBSCAN clustering for unassigned mentions, complemented by surface normalization and blocking strategies for scalability. The blocking strategy partitions mentions first by entity type and then by the first letter of the canonical name. A detailed system description is provided in Matela and Krüger 2026.

5.2. Participants' System

Table 2 provides an overview of the submitted systems. Team **aalkan** from the Harvard-Smithsonian Center for Astrophysics submitted two unsupervised, fine-tuning free systems for all three subtasks. The first system, Fuzzy Matching (FM), computes pairwise lexical similarity between mention strings using the Ratcliff/Obershelp algorithm and applies transitive closure over linked pairs to form clusters. The second system, Context-Aware Representations (CAR), encodes each mention using the lightweight `all-MiniLM-L6-v2` sentence transformer, combining a mention-level embedding with a document-level embedding aggregated from up to ten mention-bearing sentences, and clusters the resulting representations using agglomerative clustering with cosine distance. CAR consistently outperforms FM by approximately one CoNLL F1 point across all subtasks, with the advantage most visible on CEAFe, reflecting improved cluster purity. A controlled noise-injection study reveals complementary failure modes: CAR is substantially more robust to mention boundary errors, while FM degrades more gracefully under mention substitution. In terms of scalability, FM scales superlinearly with corpus size due to its pairwise comparison step, whereas CAR encodes each mention independently and scales approximately linearly, making it the more practical choice for large-scale pipelines.
mhassan from ZBW Leibniz Information Centre for Economics and the University of Greifswald submitted a supervised neural system for Subtasks 1 and 2. The system follows a three-stage pipeline. First, each mention is represented as a structured input string of the form `mention_text [SEP] sentence_context` and encoded using a SciBERT model (Beltagy et al., 2019) trained with Supervised Contrastive (SupCon) loss (Khosla et al., 2021), which optimizes all mention pairs within a batch simultaneously, pulling coreferent mentions together and pushing non-coreferent mentions apart in a shared 256-dimensional embedding space. Second, a set of software-aware heuristics adjusts the pairwise cosine similarity matrix before clustering: a name canonicalization boost ($\delta = +0.5$) is applied when two mention strings normalize to the same canonical form, and a developer conflict penalty ($\delta = -0.8$) is applied when metadata indicates different software developers. Third, Hierarchical Agglomerative Clustering (HAC) with average linkage and a fixed distance threshold of $\theta = 0.5$ produces the final clusters. To prevent

data leakage, the model is trained using cluster-level splitting, ensuring that all mentions of a given software entity are assigned exclusively to either the training or validation set. The system achieves a CoNLL F1 of 0.92 on both subtasks.
Neither participant system demonstrated practical scalability for Subtask 3. *mhassan* did not submit results for Subtask 3, likely due to the $\mathcal{O}(n^2)$ complexity of HAC over 219,950 mentions. *aalkan* completed Subtask 3 in approximately 2.2 hours using FM, compared to approximately 3 minutes for Baseline II, highlighting that blocking strategies are not merely an optimization but a practical necessity for cross-document coreference resolution at scale.

5.3. Shared Task Implementation

SOMD 2026 is hosted on CodaBench (Xu et al., 2021; CodaBench, 2026). The shared task schedule follows the official task page (NFDI4DataScience, 2026b), with training and development data released on January 20, 2026, and the competition phase closing on February 20, 2026. System description papers were due on March 10, 2026 (see the task website for details).

Participants submit a file that assigns a predicted `cluster_id` to each `mention_id` in the test set. The platform computes MUC, B^3, CEAFe, and CoNLL F1 and publishes leaderboards per subtask.

6. Results and Discussion

6.1. Participants and Overall Results

Two teams submitted valid systems to the shared task: *aalkan*, and *mhassan*. *aalkan* participated in all three subtasks, while *mhassan* submitted results for Subtasks 1 and 2. Across subtasks in which both teams competed the ranking was consistent, suggesting that the participants used approaches that generalized well across the different input conditions and that the relative system strengths were stable under the task variations.

Baseline II outperforms both participants' system across every subtask what demonstrates that the hybrid

Table 3 presents the final results. Compared to the TF-IDF + DBSCAN baseline (Baseline I), all submitted systems achieve consistent improvements in CoNLL and CEAFe, particularly in Subtasks 1 and 2, indicating that embedding-based and hybrid approaches provide more precise entity-level clustering. For the competition phase, *aalkan* (0.96, 0.96, 0.94) achieved the highest CoNLL score for three respective subtasks, following *mhassan* (0.92, 0.92). Additionally, our Baseline II achieved the overall performance (CoNLL F1: 0.98, 0.98, 0.96). From the result, we infer that the MUC scores are near saturation (0.97–0.99) across all systems, whereas larger differences are observed in CEAFe.

Overall, the high and consistent scores across gold mentions, predicted mentions, and the large-scale setting suggest that software mention coreference in scholarly text is primarily driven by surface similarity. Performance decreases only slightly on the large-scale task, indicating that the proposed approaches generalize well to noisy, larger inputs.

6.2. Interpreting Differences across Metrics

The metric patterns indicate that most systems reliably recover the broad coreference structure. High MUC scores indicate that systems correctly establish coarse links between mentions. In contrast, CEAFe separates systems more clearly, reflecting differences in cluster precision and entity-level alignment. Since CEAFe penalizes both over-merging and fragmentation, lower scores suggest less precise cluster boundaries.

B^3 scores fall between MUC and CEAFe and generally reflect overall system quality. The identical ranking across all three subtasks indicates that clustering strategy is the main factor influencing performance, while sensitivity to extraction noise or corpus scale appears limited.

In summary, the main remaining challenges lie in improving cluster purity and entity-level alignment rather than in establishing basic cross-document links or handling scale.

6.3. Scalability Analysis

Subtask 3 was explicitly designed to challenge systems on whether they can maintain accuracy while handling a significant increase in computational load. Runtime measurements conducted on the same machine illustrate how differently systems respond to this challenge. The system of a winning participant (*aalkan*) requires approximately 19 seconds on Subtask 2 and approximately 2.2 hours on Subtask 3. In contrast, Baseline II (Matela and Krüger, 2026) completes Subtask 2 in approximately 8 seconds and Subtask 3 in approximately 3 minutes, representing a significant runtime reduction on the large-scale task.

These results highlight that blocking is an essential component for scalable cross-document coreference resolution,

7. Conclusion and Future Directions

SOMD 2026 introduces a shared evaluation setting for cross-document software mention coreference resolution and provides three subtasks spanning

System	Rep.	Method	Clustering
aalkan	FM (1) / CAR (2)	Lexical (1) / Embedding sim. (2)	Transitive closure(1) / Agglomerative(2)
mhassan	SciBERT (SupCon)	Supervised contrastive	HAC (avg.)
Baseline I	TF–IDF	Cosine similarity	DBSCAN
Baseline II	SBERT	FAISS + hybrid retrieval	HDBSCAN (+ blocking)

Table 2: Overview of submitted systems and baselines for SOMD 2026. **Rep.** denotes the mention representation model used. **Method** summarizes the core modeling strategy for computing mention similarity or entity assignment (e.g., retrieval, contrastive learning, or lexical similarity). **Clustering** indicates the algorithm used to group mentions into cross-document coreference clusters. FM = Fuzzy Matching; CAR = Context-Aware Representations; SupCon = Supervised Contrastive Loss; HAC = Hierarchical Agglomerative Clustering.

System	F1-MUC	F1-B^3	F1-CEAFE	F1-CoNLL
Subtask 1: Coreference resolution over gold standard mentions across multiple documents				
aalkan	0.98	0.96	0.93	**0.96**
mhassan	0.97	0.94	0.86	0.92
Baseline I	0.92	0.86	0.68	0.82
Baseline II	**0.99**	**0.99**	**0.96**	**0.98**
Subtask 2: Coreference resolution over predicted mentions across multiple documents				
aalkan	0.99	0.96	0.93	**0.96**
mhassan	0.98	0.93	0.85	0.92
Baseline I	0.94	0.87	0.72	0.84
Baseline II	**0.99**	**0.99**	**0.95**	**0.98**
Subtask 3: Coreference resolution over predicted mentions across multiple documents at scale				
aalkan	0.99	0.94	0.90	**0.94**
Baseline I	0.98	0.94	0.90	0.94
Baseline II	**0.99**	**0.97**	**0.92**	**0.96**

Table 3: Results for SOMD 2026 across three subtasks, reported as F1 scores for MUC, B^3, CEAFe, and their unweighted average (CoNLL F1). MUC measures link-level overlap, B^3 evaluates mention-level cluster overlap, and CEAFe scores entity-level alignment via one-to-one cluster matching. Two teams submitted systems: *aalkan* participated in all three subtasks, and *mhassan* participated in Subtasks 1 and 2 only. Baseline I is a TF–IDF + DBSCAN system; Baseline II is a hybrid Sentence-BERT + FAISS + HDBSCAN system. Bold indicates the best score per column within each subtask.

gold mentions, automatically extracted mentions, and scale-oriented sampling from SoftwareKG. The new annotations and the benchmark design support research on robust and scalable clustering methods, which are needed for knowledge graph construction and large-scale studies of software use in science.

Future work can extend the benchmark in several ways: (i) adding explicit links to external identifiers (e.g., registry ids or repository ids) to bridge clustering and entity linking, (ii) providing standard blocking candidates to compare scalable approaches more directly, (iii) expanding to more domains and languages, and (iv) evaluating end-to-end pipelines that combine detection, metadata extraction, and disambiguation.
Furthermore, we would like to emphasize on investigating blocking strategies that reduce the clustering search space without sacrificing recall as a future direction. As the shared task results demonstrate, search space reduction is the primary bottleneck to deploying coreference resolution systems at knowledge graph scale. While Baseline II demonstrates that partitioning mentions by entity type and canonical name initial yields substantial runtime gains. More systematic approaches such as learned blocking, approximate nearest-neighbor indexing, or ontology-guided partitioning remain largely unexplored for software mention disambiguation and represent a promising direction for future research.

8. Ethics Statement

This shared task uses software mentions from scholarly publications and builds on the SoMeSci

and SoftwareKG resources. The data consists of scientific articles and derived mention-level annotations for software entities and their cross-document coreference relations. The task does not involve private communication, patient records, or other directly sensitive personal data.

9. Limitations

This benchmark is limited to the entity type *software* and does not cover cross-document coreference for other scientific entity types. In addition, the data is grounded in SoMeSci and SoftwareKG, with the large-scale setting based on PubMed Central articles. The extent to which the reported results generalize across disciplines, publication cultures, and languages remains to be tested.

A second limitation is that the task is not fully end-to-end. The benchmark starts from gold or automatically extracted software mentions and evaluates clustering over these mentions, rather than jointly evaluating mention detection, metadata extraction, and cross-document disambiguation in a single pipeline.

A third limitation concerns evaluation coverage. In Subtask 2, evaluation is restricted to mentions that can be aligned to existing gold clusters or were manually annotated, and in Subtask 3, system scoring is performed on an annotated subset of the large-scale test data. Reported scores should therefore be interpreted as benchmark scores for the evaluated subsets rather than exhaustive quality estimates over the full mention pool.

A fourth limitation is the limited number of competitive submissions. Although five participants registered, only two teams submitted valid systems. This restricts how strongly the shared task results can be used to characterize the broader state of the field.

Finally, the strong results across subtasks suggest that many benchmark cases can be handled using surface similarity and lightweight metadata signals. Harder cases involving short ambiguous names, sparse metadata, and stronger domain shift may still be underrepresented. This is consistent with the paper's own future directions, which include expansion to more domains and languages and evaluation of end-to-end pipelines.

10. Acknowledgements

This paper was prepared within the NFDI4DS and the BERD@NFDI consortium in the context of the work of the National Research Data Infrastructure (NFDI) Association. NFDI is funded by the Federal Republic of Germany and the 16 federal states. The NFDI4DS consortium is supported within NFDI by the German Research Foundation (DFG) – NFDI 27/1-2026, project number 460234259. The BERD@NFDI consortium is supported within NFDI by the German Research Foundation (DFG) – NFDI 27/1-2026, project number 460037581. We thank both NFDI4DS and BERD@NFDI for their funding and support. Special thanks go to all institutions and individuals contributing to the association and its goals. Finally, we would like to thank Luke Friedrich for his contribution to the annotation of the software coreferences.

References

Amit Bagga and Breck Baldwin. 1998. Algorithms for scoring coreference chains. In *Proceedings of the First International Conference on Language Resources and Evaluation (LREC'98), Workshop on Linguistic Coreference*, pages 563–566, Granada, Spain.

Iz Beltagy, Kyle Lo, and Arman Cohan. 2019. SciBERT: A pretrained language model for scientific text.

CodaBench. 2026. Codabench. https://www.codabench.org/. Accessed: 2026-02-20.

Martin Ester, Hans-Peter Kriegel, Jörg Sander, and Xiaowei Xu. 1996. A density-based algorithm for discovering clusters in large spatial databases with noise. In *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining (KDD-96)*, pages 226–231.

GESIS. 2026. SoftwareKG dataset portal. https://data.gesis.org/softwarekg/. Accessed: 2026-02-20.

Daniel S. Katz and Neil P. Chue Hong. 2024. Special issue on software citation, indexing, and discoverability. *PeerJ Computer Science*, 10:e1951.

Aliakbar Keshtkaran, Siti Sophiayati Yuhaniz, and Suhaimi Ibrahim. 2017. An overview of cross-document coreference resolution. In *2017 International Conference on Computer and Drone Applications (IConDA)*, pages 43–48. IEEE.

Prannay Khosla, Piotr Teterwak, Chen Wang, Aaron Sarna, Yonglong Tian, Phillip Isola, Aaron Maschinot, Ce Liu, and Dilip Krishnan. 2021. Supervised contrastive learning.

Frank Krüger, Saurav Karmakar, and Stefan Dietze. 2024. SOMD@NSLP2024: Overview and insights from the software mention detection shared task. In *International Workshop on Natural Scientific Language Processing and Research Knowledge Graphs*, pages 247–256. Springer.

Xiaoqiang Luo. 2005. On coreference resolution performance metrics. In *Proceedings of Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing*, pages 25–32, Vancouver, British Columbia, Canada. Association for Computational Linguistics.

Julia Matela and Frank Krüger. 2026. Semantic centroids and hierarchical density-based clustering for cross-document software coreference resolution. arXiv:2603.24246.

NFDI4DataScience. 2026a. NSLP 2026 workshop website. https://nfdi4ds.github.io/nslp2026/. Accessed: 2026-02-20.

NFDI4DataScience. 2026b. SOMD shared task 2026 (NSLP 2026). https://nfdi4ds.github.io/nslp2026/docs/somd_shared_task.html. Accessed: 2026-02-20.

Sameer Pradhan, Alessandro Moschitti, Nianwen Xue, Olga Uryupina, and Yuchen Zhang. 2012. CoNLL-2012 shared task: Modeling multilingual unrestricted coreference in OntoNotes. In *Joint Conference on EMNLP and CoNLL - Shared Task*, pages 1–40, Jeju Island, Korea. Association for Computational Linguistics.

Gerard Salton and Chris Buckley. 1988. Term-weighting approaches in automatic text retrieval. *Information Processing & Management*, 24(5):513–523.

David Schindler, Felix Bensmann, Stefan Dietze, and Frank Krüger. 2021. Somesci—a 5 star open data gold standard knowledge graph of software mentions in scientific articles. In *Proceedings of the 30th ACM International Conference on Information and Knowledge Management (CIKM 2021)*, pages 4574–4583.

David Schindler, Felix Bensmann, Stefan Dietze, and Frank Krüger. 2022. The role of software in science: a knowledge graph-based analysis of software mentions in pubmed central. *PeerJ Computer Science*, 8:e835.

David Schindler, Tazin Hossain, Sascha Spors, and Frank Krüger. 2023. A multilevel analysis of data quality for formal software citation. *Quantitative Science Studies*, 5:637–667.

Sharmila Upadhyaya, Wolfgang Otto, Frank Krüger, and Stefan Dietze. 2025. SOMD2025: A challenging shared tasks for software related information extraction. In *Proceedings of the Fifth Workshop on Scholarly Document Processing (SDP 2025)*, pages 137–145, Vienna, Austria. Association for Computational Linguistics.

Marc Vilain, John Burger, John Aberdeen, Dennis Connolly, and Lynette Hirschman. 1995. A model-theoretic coreference scoring scheme. In *Sixth Message Understanding Conference (MUC-6): Proceedings of a Conference Held in Columbia, Maryland, November 6-8, 1995*.

Zhen Xu, Cédric Ré, et al. 2021. Codabench: Flexible, easy-to-use and reproducible meta-benchmark platform. arXiv:2110.05802.