

How Do Algorithms Interact in Scientific Research? Large-Scale Semantic Relation Identification and Analysis Based on Full-Text Papers

Yuzhuo Wang

Anhui University

Yi Zhao

Anhui University

Wenqing Wu

Nanjing University of Science and Technology

Heng Zhang

zh_heng@ccnu.edu.cn

Central China Normal University

Research Article

Keywords: Algorithm entities, Relation extraction, Full-text paper, Large language models, Relation network

Posted Date: August 3rd, 2026

DOI: <https://doi.org/10.21203/rs.3.rs-10102565/v1>

License: © ⓘ This work is licensed under a Creative Commons Attribution 4.0 International License. [Read Full License](#)

Additional Declarations: No competing interests reported.

Abstract

In the era of artificial intelligence, algorithms have become a core driver of scientific research and innovation. Identifying algorithm entities and their semantic relations is important for understanding the development and innovation trajectories of algorithms. Focusing on natural language processing (NLP), this study identifies semantic relations among algorithm entities in full-text papers and analyzes their interaction patterns and evolution. We first collected full-text papers from three major NLP conferences and constructed an updated corpus. Pre-trained models were then fine-tuned to identify algorithm entities and sentences. After that, we defined an algorithm semantic relation framework and manually annotated a gold-standard corpus. Various large language models (LLMs) and pre-trained language models (PLMs) were used for identifying sentence-level relations among algorithms. Finally, we analyzed representative interacting algorithms and the evolution trends of algorithm relation networks. The results show that SciBERT achieves the best overall performance in relation identification, while LLMs with few-shot learning and chain-of-thought reasoning also obtain competitive results. In the NLP domain, algorithm semantic relations are dominated by the Compare relation, and multiple potential interaction motivations underlie the same kinds of relation. All of the relation-specific networks show expanding scale, deepening interaction, and accelerated renewal over time, reflecting the increasingly intense ecosystem of algorithm competition in the NLP field. This study provides data resources and methodological references for relation identification and graph construction among knowledge entities.

1 Introduction

Against the backdrop of the rapid development of artificial intelligence, algorithms have become a crucial force driving scientific innovation as well as industrial development. As a core component of artificial intelligence, algorithms extend the performance boundaries of intelligent technologies and further consolidate the foundation of scientific understanding and discovery (H. Wang et al., 2023). In scientific research, algorithms have been widely applied to core tasks across both the natural sciences and the humanities and social sciences domains (Bileschi et al., 2022; Suissa et al., 2022; Vuković et al., 2025; Y. Chen et al., 2026). In industrial production, algorithms are participating in the wave of Industry 5.0, helping improve production efficiency, enhance decision-making capabilities, and optimize user experiences (Rashid & Kausik, 2024; Raza et al., 2025). Therefore, in both academic and industrial domains, the continuous advancement of algorithm research and innovation has become a key task for practitioners across fields.

However, algorithm innovation rarely occurs in isolation. An influential algorithm is often gradually formed through repeated interactions and evolutionary processes based on existing algorithms (Zha et al., 2019). As Kuhn (1996) argued in *The Structure of Scientific Revolutions*, the development of science and technology can be understood as a dynamic process of knowledge evolution, in which scientific progress unfolds through the continual alternation between knowledge accumulation and knowledge transformation. As important scientific and technological outputs, algorithms likewise develop through repeated reuse, extension, combination, and competition among individual algorithms. Therefore, understanding the emergence and innovation trajectories of novel algorithms requires more than static description and evaluation of individual algorithms. It is necessary to systematically analyze the relationships and interactions among algorithms.

Academic papers are key knowledge carriers where scholars present proposed and used algorithms, offering rich data for studying algorithm features and their relationships. With the increasing availability of full-text academic papers, existing studies have conducted large-scale automatic identification of algorithms in academic papers, examined their mention frequencies, textual positions, contextual motivations, and academic influence (Y. Wang & Zhang, 2020; Y. Wang et al., 2021, 2024). These studies have advanced our understanding of how algorithms are used and disseminated in specific domains. However, most of them focus on individual algorithm entities, offering limited insight into the modes of interaction and relational structures among different algorithms. Some studies have used co-occurrence relations to capture algorithm interactions (Y. Wang et al., 2025), but co-occurrence without semantic differentiation can only indicate which algorithms appear together. It cannot explain why algorithms co-occur, how algorithm clusters form, or how algorithm networks interact and evolve, thereby constraining our understanding of algorithm innovation trajectories, community evolution patterns, and the competitive landscape of technologies.

Therefore, moving from scattered descriptions of algorithm entities to a structured representation of algorithm relations, this study identifies semantic relations between algorithm entities in academic papers and further examines the structural characteristics and evolutionary differences of algorithm communities formed by different types of relations. Regarding the core research object of this study, we follow prior work and define an algorithm as “an operational process that incorporates a set of inputs and outputs, and that is explicit and efficient, incorporating some of the machine learning models, but distinct from software and application systems” (Y. Wang et al., 2024). Algorithm entities are defined as nouns or noun phrases used to denote algorithms. With the specified research object, this study takes the field of natural language processing as an example and draws on the rich textual information contained in academic papers to address the following research questions:

- (1) What semantic relations commonly exist between algorithm entities, and how well do different models identify these relations?
- (2) What characteristics do algorithm entity pairs exhibit under different relation types?
- (3) How do algorithm communities formed by different relations evolve over time?

By modeling relations between algorithm entities, this study identifies how algorithms are used, extended, compared, and positioned in academic papers, offering finer-grained evidence for understanding their interaction, organization, and evolution. More broadly, algorithm relation identification

enables researchers to move from an “entity list” to a “relation graph,” providing a new basis for exploring knowledge organization and evolution patterns.

2 Related works

This study focuses on identifying semantic relations between algorithm entities mentioned in the full text of academic papers. Accordingly, we systematically review prior research on algorithm entities and metadata in scholarly literature. Next, we regard algorithms as a form of scholarly knowledge and review related studies on knowledge entity relation extraction (RE).

(1) Algorithm identification and analysis based on the academic papers

Algorithm entity and metadata recognition is the fundamental step in algorithm-related research, where the most direct approaches are manual annotation (Handschuh & QasemiZadeh, 2014; Duttax & Patel, 2024) and rule-based methods (Y. Wang & Zhang, 2019). To address the difficulty of updating rules, researchers explored machine learning methods for algorithm identification (Tuarob et al., 2013). Further studies found that combining rules with statistical machine learning can more accurately identify algorithm pseudocode and metadata (Tuarob et al., 2014, 2016). With subsequent technological advances, more recent studies have adopted deep learning and PLMs to improve the identification performance (Lei & Wang, 2019; Saritha et al., 2025).

For the identified algorithm entities and metadata, researchers further examined their syntactic and semantic features. Studies on syntactic features focus on mention frequency and location. Frequency information has been used to assess academic impact (Y. Wang et al., 2021), and location information has been used to support algorithm identification and evaluation (Safder et al., 2017; Y. Wang & Zhang, 2018). Studies on semantic features mainly examine the functions, motivations, and relationships of algorithm mentions. Researchers have classified authors’ motivations for mentioning algorithms into categories such as use and improvement (Safder et al., 2017; Y. Wang et al., 2024). Some studies further explored relationships between algorithms. Tuarob et al. (2012) focused on algorithm co-citation relations and used co-citation networks for algorithm recommendation. Other studies examined the algorithm co-occurrence relationship. Zha et al. (2019) identified algorithm entities co-occurring in tables and regarded them as having competitive relations. Wang et al. (2025) constructed co-occurrence networks to quantify the importance of algorithms within groups. However, existing studies have not yet identified more fine-grained semantic relations between algorithms.

(2) Knowledge entity relation extraction in academic text

Entity relation extraction is a core task in NLP research and supports knowledge graph construction and intelligent information retrieval. Early studies mainly relied on rules, handcrafted features, and statistical machine learning models (H. Wan et al., 2016). With the development of deep learning, researchers began to use it to capture contextual semantic features for relation identification (Gábor et al., 2018; Luan et al., 2018; L. Chen et al., 2020). The emergence of PLMs further reshaped knowledge entity relation extraction (Zhong & Chen, 2021; Jiang et al., 2024). On this basis, few-shot optimization (Moscato et al., 2023), domain pre-training (Yu et al., 2025), and integration with graph neural networks (R. Wu & Ardakani, 2025) and dependency syntax (Zhang et al., 2024) have further improved PLMs’ performance. Nevertheless, challenges remain in identifying the direction of asymmetric relations and conducting deeper semantic reasoning (Lyu et al., 2022).

With the rapid development of generative AI technologies, LLMs have also been applied to entity relation extraction. Some researchers employed zero-shot approaches, directly invoking LLMs to solve RE tasks (Wei et al., 2024). Based on this, Wadhwa et al. (2023) found that few-shot promotion could effectively activate the model’s matching capabilities. Consequently, chain-of-thought reasoning (CoT) has been introduced to optimize LLM applications. Approaches such as the stepwise reasoning in ERA-CoT (Liu et al., 2024), self-supervised CoT-SC (X. Wang et al., 2023), and complexity-aware Complex-CoT (Fu et al., 2023) all significantly outperform direct LLM usage. Another strand of work leverages Retrieval-Augmented Generation (RAG) to mitigate LLM “hallucination” by incorporating external reliable knowledge (Z. Wan et al., 2023; F. Wu et al., 2025). However, researchers also observed that despite LLMs’ strong generalization capabilities, parameter fine-tuning remains the preferred approach for domain-specific tasks or handling relation types not naturally represented in language (Tran et al., 2024).

Overall, existing studies on algorithm entities in academic papers have rarely examined fine-grained semantic relations between individual algorithms. Although relation extraction methods have matured, most research focuses on benchmark-driven methodological improvements rather than large-scale analysis of algorithm interactions and their evolution. Therefore, building on prior work in algorithm entity extraction, this study extends to semantic relation identification and investigates dominant interaction patterns among algorithms and their temporal changes. We aim to provide a data and methodological foundation for future research on semantic relation-based knowledge graphs and algorithm evolution pathways.

3 Method

Taking the field of natural language processing as an example, this study identified and analyzed semantic relations between algorithm entities in academic papers. Specifically, we first collected full-text papers from three top NLP conferences, then performed entity extraction and algorithm-related sentence filtering. Next, we constructed a semantic relation category framework and annotated training data for algorithm relation extraction. Subsequently, we selected different extraction models and strategies to evaluate their performance on the subsequent RE task. Finally, annual relation networks were constructed, and their evolutionary trends were investigated.

3.1 Data collection

We used full-text conference papers published in the three most prominent conferences in the NLP domain as the original dataset, including ACL (Annual Meeting of the Association for Computational Linguistics), EMNLP (Empirical Methods in Natural Language Processing), and NAACL (Annual Conference of the North American Chapter of the Association for Computational Linguistics), through public datasets and format conversion. For conferences held before 2022, we directly obtained full-text papers in XML format from the publicly available ACL OCL dataset (Rohatgi et al., 2023). For conferences held after 2022, since no public XML or TXT-format data are currently available, we collected PDF-format papers published between 2023 and 2025 from the ACL Anthology website¹, and converted them into TXT-format using docling². In this way, we obtained the full-text content of all conference papers published in the three conferences from 1979 to 2025.

3.2 Algorithm entities and sentences identification

This study aims to automatically identify algorithm relations at scale in a specific research field. Therefore, we first extracted sentences mentioning algorithm entities from full-text papers published in the three NLP conferences. These sentences were then used as both training data and unlabeled data for subsequent algorithm relation identification.

(1) Supplementing the training corpus

Existing work has extracted algorithm entities from NLP papers and provided 57647 sentences containing algorithm entities as training data (Y. Wang et al., 2025). However, the existing corpus only covers algorithm entities appearing in papers published between 1979 and 2020, whereas many new algorithm entities have emerged since 2020. Therefore, building on the existing training corpus, we recruited three annotators to annotate algorithm entities mentioned in ACL conference papers published between 2021 and 2025. Specifically, we first randomly selected 500 sentences to assess inter-annotator agreement. After the pairwise agreement among annotators reached 0.851, the remaining unlabeled sentences were evenly assigned to the three annotators for algorithm entity annotation. Finally, we added 6000 newly annotated sentences according to the yearly distribution and merged them with the original training corpus, forming an updated corpus for algorithm entity extraction.

(2) Identifying the algorithm entity and sentences

Using the updated training corpus, we fine-tuned several algorithm entity extraction models following prior studies. The performance of each model is reported in Table 1. Among them, SciBERT+Bi_LSTM + CRF achieved the best performance. We therefore applied this model to identify algorithm entities in the remaining non-training sentences from ACL, EMNLP, and NAACL papers. The identified algorithm-related sentences were then merged with the training corpus to form the complete set of candidate entities and sentences.

Table 1
The performance of algorithm entity identification

Method	P	R	F1
BERT + CRF	92.32%	93.49%	92.90%
BERT+Bi_LSTM + CRF	93.03%	93.19%	93.11%
SciBERT + CRF	91.60%	93.79%	92.68%
SciBERT+Bi_LSTM + CRF	92.59%	94.33%	93.45%

(3) Algorithm entities filtering and standardization.

For candidate entities, we first removed those appearing in ≤ 5 sentences and manually filtered high-frequency noisy terms (≥ 100 mentions). We then applied a rule-based filter to remove entities whose suffixes were non-algorithm terms (e.g., “data,” “dataset,” “system,” “software,” and so on). For the remaining entities, we retained only sentences mentioning at least two algorithms as the unlabeled corpus for relation identification.

On this basis, we further standardized the algorithm entities in these multi-entity sentences. First, we conducted preliminary cleaning of the original algorithm names. All names were converted to lowercase, non-alphanumeric characters were removed, and NLTK was used for tokenization, part-of-speech tagging, stopwords filtering, and lemmatization. During lemmatization, each word was mapped to its corresponding noun, verb, adjective, or adverb form in WordNet according to its part of speech. Low-information words, such as prepositions, articles, conjunctions, and pronouns, were also removed. The cleaned tokens were then sorted alphabetically and concatenated to form a normalized key for each algorithm entity. Algorithm names with the same normalized key were grouped together, and the most frequent original name in each group was selected as its intermediate representative name.

Subsequently, for the unique algorithm entities obtained after preliminary merging, we used the Gemma embedding model (google/embeddinggemma-300m) to generate semantic vectors and used ChromaDB for vector storage and nearest-neighbor retrieval. For each entity, we retrieved 30 similar candidates and computed cosine similarity, retaining only pairs with similarity greater than 0.8. Finally, we performed bottom-up hierarchical clustering with a decreasing similarity threshold. Initially, each algorithm entity was treated as an individual cluster. We then

constructed an adjacency graph based on the similarity relations between entities and calculated average inter-cluster similarity only between candidate clusters connected by similarity relations, thereby improving clustering efficiency. The clustering threshold started at 0.99 and was gradually reduced to 0.85 with a step size of 0.01. When the average similarity between two clusters exceeded the current threshold, the two clusters were merged. If no further merging was possible under the current threshold, the threshold was lowered, and the iteration continued. After clustering was complete, the most frequent algorithm name in each cluster was selected as the final representative name. The original variants retained during preprocessing were then mapped to this representative name, resulting in the final normalized set of algorithm entities. In the end, we obtained 30,274 algorithm entities.

3.3 Algorithm relation extraction and analysis

(1) Annotating the training corpus for algorithm relation extraction

Based on the sentences mentioning multiple algorithm entities obtained in the previous step, we manually annotated a training corpus for algorithm relation identification. Since ACL papers cover a longer time span and a larger number of publications, we focused on algorithm sentences from ACL papers. According to the yearly proportion of papers mentioning algorithm entities, we randomly sampled 9,000 algorithm sentences and assigned them to annotators for semantic relation annotation. We recruited four annotators and conducted two rounds of pilot annotation. Before the pilot annotation, we initially divided algorithm entity relations into eight categories: Use, Trained by, Compare with, Collaborate with, Parallel of, Superclass of, Same as, and Other. In each round, 200 sentences were randomly selected and independently annotated by all four annotators.

In the annotation task, for an algorithm entity pair appearing in the same sentence, we aimed to identify a relational triplet $T = (E_i, R_k, E_j)$, indicating whether a relation ($R_k \in R$) exists between the pair of entities (E_i, E_j). When a sentence contained multiple algorithm entity pairs, each pair and its corresponding relation were annotated separately. Following prior studies, we treated (E_i) and (E_j) as the subject and object, respectively, and annotated the relation from (E_i) to (E_j), regardless of their physical order in the sentence. Following the approach of Zhong and Chen (2021), we performed relation learning and recognition for each entity pair independently, treating other entities in the sentence as contextual information. Accordingly, after formal annotation, we preserved the corpus by retaining only one algorithm entity pair per sentence. Sentences containing multiple algorithm entity pairs were duplicated, differing only in entity labels and relation labels.

After the first round of pilot annotation, inter-annotator agreement was relatively low, and some relation categories had very few instances. After the second round, the average pairwise inter-annotator agreement reached 0.7630 (Cohen’s Kappa). Based on the pilot results, we refined the relation schema into six final categories: Compare with, Use, Cooperate with, Same as, Superclass of, and Other, with Use and Superclass being directed relations and the others undirected. The definitions of these categories are presented in Table 2. For ease of presentation, these relations are hereafter abbreviated as Compare, Use, Collaborate, Synonym, Superclass, and Other.

Table 2
The definition of the semantic relation between a pair of algorithm entities

Relation	Definition
Compare with	E_i and E_j are compared in terms of their features and performance, or it is indicated that they can both independently solve the same task.
Use	E_i directly uses E_j , or E_i ’s training, construction, evaluation, etc., based on E_j .
Cooperate with	E_i and E_j are used together to solve problems.
Same as	E_i and E_j represent the same or different names for the same algorithm.
Superclass of	E_i is a superclass of E_j , encompassing E_j .
Other	E_i and E_j are merely mentioned together without demonstrating a clear semantic relation.

Following this classification framework, we divided the remaining unlabeled corpus into two subsets. The four annotators were paired into two groups, with each group annotating one subset. For sentences where the two annotators in the same group produced inconsistent labels, they conducted a second round of discussion and adjudication to determine the final relation label. Ultimately, we obtained 16,531 sentences as training data.

(2) Selecting algorithm relation identification models

For the annotated corpus described above, we randomly split all sentences into train, valid, and test sets at a ratio of 8:1:1. The same test set was used to evaluate the performance of different PLMs and LLMs on the relation extraction. We adopted two model-use strategies: direct prompting and supervised fine-tuning. First, we evaluated the performance of LLMs, including GPT-5, DeepSeek-Reasoner, Gemini-3.1-pro-preview, and Claude-Sonnet-4-6. During evaluation, we tested different prompting strategies, including zero-shot prompting, few-shot prompting (providing one to two examples per relation category), and chain-of-thought (CoT) reasoning, with the sampling temperature set to temperature = 0 to ensure deterministic

decoding and reproducibility. For the CoT method, a two-step reasoning process was adopted: the model first identified key trigger words or phrases indicative of the relation, and then produced candidate relation labels with brief justifications based on predefined decision criteria.

Next, we adopted a supervised fine-tuning strategy and selected three pre-trained language models, namely BERT, SciBERT, and DeBERTa, to evaluate their performance on relation identification. On the one hand, we directly fine-tuned different pre-trained models for algorithm entity relation prediction. On the other hand, we added a BiLSTM layer on top of the basic PLM architecture to examine whether an additional sequential modeling component could improve prediction performance. During fine-tuning, we conducted hyperparameter tuning under fixed random seeds. Each model was fine-tuned for five epochs. To ensure robust results, we repeated the experiments five times with different random seeds (11, 22, 33, 44, and 55) and reported the average performance across the five runs as the final performance for each model. In addition, we examined supervised fine-tuning of Qwen-series models on the annotated dataset. Specifically, we fine-tuned the open-source Qwen model using a low-rank adaptation (LoRA) approach on the same data as previous PLMs. During fine-tuning, the LoRA configuration was set to $r = 8$, $\text{lo_alpha} = 32$, and $\text{lo_dropout} = 0.1$. The model was trained for three epochs with an effective batch size = 16, a learning rate = 1×10^{-4} , and a random seed identical to that of the pre-trained model. During inference, the sampling temperature was set to temperature = 0 to ensure deterministic decoding and reproducibility.

(3) Analyzing the algorithm relations

Calculation of algorithm-pair interaction intensity. For each algorithm pair with an identified relation in a sentence, we unified them using normalized names, then calculated their proportions among all algorithm pairs and under each specific relation type, to identify representative algorithm pairs across different interaction contexts. Specifically, when relation types were not considered, all algorithm pairs were treated as unordered pairs and reordered alphabetically. We then counted the paper IDs in which each algorithm pair appeared. When relation types were considered, unordered relations were processed in the same way by alphabetically reordering the two entities. For ordered relations, the original order of the two entities was retained. We then counted the paper IDs in which each algorithm pair appeared under each relation type.

On this basis, we constructed the indicator to measure the interaction intensity of algorithm pairs. Specifically, for an algorithm pair p , we first calculated the proportion of papers in which it appeared in year t relative to the total number of papers published in that year. This value represents the yearly relative paper proportion of the algorithm pair. We then summed the yearly relative proportions from the first year in which the algorithm pair appeared to 2025, and divided the sum by the corresponding time span. This procedure reduces the influence of differences in yearly publication volume and the cumulative advantage of algorithm pairs that appeared earlier. The interaction intensity of algorithm pair p is calculated as follows:

$$\text{Interaction}(p) = \frac{\sum_{t=y_p}^{2025} \frac{D_{p,t}}{D_t}}{2025 - y_p + 1}$$

1

Where $D_{p,t}$ denotes the number of papers in which algorithm pair p appeared in year t , D_t denotes the total number of papers in year t , and y_p denotes the first year in which algorithm pair p appeared in the dataset. A higher value indicates that the algorithm pair appeared more consistently in relevant papers during its lifecycle.

When the relation type (r) is further distinguished, the interaction intensity of an algorithm pair under a specific relation type can be defined as:

$$\text{Interaction}(p, r) = \frac{\sum_{t=y_{p,r}}^{2025} \frac{D_{p,r,t}}{D_t}}{2025 - y_{p,r} + 1}$$

2

where $D_{p,r,t}$ denotes the number of papers in which algorithm pair p appeared with relation type r in year t , and $y_{p,r}$ denotes the first year in which algorithm pair p appeared with relation type r .

Construction and analysis of algorithm relation networks. To further reveal the structural evolution of different types of algorithm relations, we constructed yearly algorithm relation networks for each relation type. Specifically, for the algorithm relation pairs identified in each year, the two algorithm entities in a pair were treated as network nodes, and the relation between them was treated as an edge. Since the purpose of this study is not to evaluate the importance or influence of individual algorithm entities using network centrality indicators, but to analyze the overall evolution of algorithm interaction structures across different relation types, we did not assign edge weights during network construction.

Based on these networks, we analyzed yearly relation networks from three aspects: network size, internal connectivity, and structural renewal. First, we counted the number of nodes and edges in each yearly relation network to measure changes in the overall scale of algorithm interaction networks. Second, we calculated the average node degree and clustering coefficient to examine changes in internal connectivity and local aggregation. The average node degree reflects the average number of other algorithms connected to each algorithm entity, while the clustering

coefficient measures the extent to which algorithm entities form local interaction groups. Finally, we calculated edge similarity, the proportion of newly added edges, and the proportion of disappearing edges between adjacent years to quantify the yearly renewal of algorithm relation networks.

4 Results

4.1 Performance of different methods in algorithm entity relation identification

(1) Results of using large language models

To enable a more objective comparison between PLMs and LLMs, the same test set derived from the aforementioned 8:1:1 split was used as input for different LLMs, and their performance in identifying semantic relations between algorithm entities was evaluated under the same setting. Table 3 summarizes the performance of the evaluated LLMs. Overall, all models demonstrate the ability to perform algorithm entity relation extraction without task-specific training. Under the zero-shot setting, GPT achieves the strongest overall performance, with an F1 score of 74.27%, followed by Gemini at 71.83%. Claude and DeepSeek obtain comparatively lower F1 scores.

Table 3
Performance of Large Language Models

Method	Macro-P				Macro-R				Macro-F1			
	GPT	Deepseek	Claude	Gemini	GPT	Deepseek	Claude	Gemini	GPT	Deepseek	Claude	Gemini
zero-shot	75.21%	74.27%	68.80%	73.50%	73.55%	69.37%	67.33%	71.70%	74.27%	67.14%	67.13%	71.83%
few-shot	75.27%	74.69%	71.06%	75.98%	74.14%	66.68%	70.03%	74.82%	74.36%	63.73%	69.85%	73.99%
few-shot + cot	75.81%	76.51%	71.69%	75.72%	74.71%	68.90%	71.70%	75.54%	74.63%	66.03%	71.21%	74.33%

The effects of prompting strategies vary considerably across models. Few-shot prompting slightly improves the F1 scores of GPT, Claude, and Gemini, whereas it reduces the performance of DeepSeek from 67.14% to 63.73%. Incorporating CoT reasoning further improves performance relative to few-shot prompting for all evaluated models, producing the best result for each model among the three prompting settings. However, the magnitude of improvement differs substantially. GPT improves only marginally from an F1 score of 74.27% under zero-shot prompting to 74.63% under the few-shot and CoT setting. In contrast, Gemini shows a more noticeable improvement from 71.83% to 74.33%. Although CoT reasoning improves DeepSeek relative to few-shot prompting, its F1 score remains lower than its zero-shot performance, indicating that additional demonstrations and explicit reasoning instructions do not necessarily benefit all models.

Across all inference settings, GPT consistently achieves the highest F1 score and reaches its best performance under the few-shot and CoT configuration. Gemini also demonstrates relatively stable performance and obtains a comparable F1 score under the same setting. Notably, DeepSeek achieves the highest precision of 76.51% under the few-shot and CoT setting, but its comparatively low recall limits its overall F1 performance. Overall, the results indicate that LLM performance in algorithm entity relation extraction depends on both model selection and prompting strategy, highlighting the importance of adapting inference protocols to the characteristics of different models.

(2) Results of fine-tuning large language models and pre-trained models

Table 4 presents the performance of the fine-tuned models, which were trained, validated, and tested on identical datasets. Overall, compared with the direct use of LLMs, fine-tuning both PLMs and LLMs leads to stronger and more stable performance. Among all evaluated models, SciBERT achieves the best overall result, with a macro-F1 score of 77.79%, followed by DeBERTa with a macro-F1 score of 77.21%. The domain-adapted or architecturally enhanced PLM encoders are particularly effective in capturing the semantic relations of algorithms in academic text. The results also show that adding a BiLSTM layer on top of PLM encoders does not consistently improve performance. For BERT, DeBERTa, and SciBERT, the BiLSTM-enhanced variants all obtain slightly lower macro-F1 scores than their corresponding base models. It can be speculated that the contextual representations learned by PLM encoders already provide sufficient sequence-level semantic information for this task, and the additional BiLSTM layer may introduce limited or even redundant modeling capacity. Compared with the PLMs, fine-tuned LLMs also exhibit competitive performance. Qwen-4B and Qwen-8B achieve macro-F1 scores of 75.51% and 75.60%, respectively, which are comparable to BERT and BERT+BiLSTM. However, their performance remains lower than that of SciBERT and DeBERTa. Therefore, considering both predictive performance and computational cost, fine-tuning PLMs, especially SciBERT and DeBERTa, remains the most effective and practical choice for the algorithm relation extraction task. We selected the fine-tuned SciBERT model as the final relation extraction model and applied it to all algorithm sentences to identify semantic relations between algorithm pairs.

Table 4
Performance of fine-tuned various models

Model	Macro-P	Macro-R	Macro-F1
BERT	76.95%	74.59%	75.53%
BERT + BiLSTM	76.08%	74.45%	75.12%
DeBERTa	78.68%	76.41%	77.21%
DeBERTa + BiLSTM	77.84%	76.52%	77.07%
SciBERT	78.92%	77.06%	77.79%
SciBERT + BiLSTM	78.61%	76.61%	77.33%
Qwen-4B	76.09%	74.54%	75.51%
Qwen-8B	76.45%	74.29%	75.60%

4.2 Distribution and representative patterns of algorithm relations

Based on the predicted relations between algorithm entity pairs in each sentence, we first examined the overall distribution of different relation types. For each relation type, its frequency was increased by one whenever it appeared in an algorithm sentence. We then calculated the proportion of sentences associated with each relation type. As shown in Table 5, among all semantic interaction relations between algorithms, Compare accounts for the largest share, whereas Collaborate accounts for the smallest. Although algorithms may also be linked through hierarchical use or category inclusion, scholars rarely describe multiple algorithm pairs as mutually using or collaborating with one another within the same research context. By contrast, competitive relations can occur across multiple combinations of comparable algorithms. We also find that Synonym relations account for a high proportion, reflecting the diversity and flexibility with which scholars refer to algorithm entities in academic writing. The same algorithm may appear in different sentences under its full name, an alternative name, or an abbreviation, indicating that algorithm entity disambiguation remains a challenging task.

Table 5
The ratio of various relations

Rank	Type	Ratio
1	Compare	54.37%
2	Synonym	12.54%
3	Use	10.67%
4	Other	9.77%
5	Superclass	6.97%
6	Collaborate	5.68%

We then compared how the proportions of different relation types changed over time. As shown in Fig. 1, no clear pattern emerges in the early years, when algorithm entities and relations were relatively rare. During the decade after 1990, the distribution of relations gradually became more stable. Since 2000, Compare has become the most prominent form of algorithm interaction, followed by Synonym and Use, which remained relatively stable during the following 20 years. After 2020, Compare experienced another round of growth, whereas the proportions of other relations decreased slightly.

We further examined representative algorithm pairs within the overall set and across different relations, based on interaction intensity. As shown in Table 6, when relation types are not distinguished, frequently interacting algorithms show clear patterns of technological clustering. Many pairs come from the same lineage or category, such as BERT–RoBERTa, BERT–XLNet, and GPT-3.5–GPT-4. The high-frequency pairs also reveal core algorithm clusters in different technological periods. We observe stable co-occurrence among traditional neural network architectures, as well as frequent interactions among large language model-related algorithms represented by the GPT series. Pre-trained models occupy an intermediate position and play a bridging role. They interact with both traditional deep learning algorithms and LLMs, as shown by pairs such as BERT–T5 and BERT–GPT-2.

Table 6
Algorithm pairs with high interaction intensity across different relation types

1	All	Collaborate	Compare	Superclass	Use
	BERT–RoBERTa	FFN–MHA	BERT–RoBERTa	ChatGPT–GPT-3.5-turbo	LLM–CoT
2	GPT-3.5–GPT-4	BiLSTM–LSTM	GPT-3.5–GPT-4	BERT–BERT-base	Word2vec–Skip-Gram
3	ChatGPT–GPT-4	CNN–LSTM	ChatGPT–GPT-4	RNN–LSTM	RNN–GRU
4	BERT–ELMo	CNN–RNN	BERT–ELMo	ChatGPT–GPT-4	RNN–LSTM
5	ChatGPT–Llama	BiLSTM–CRF	ChatGPT–Llama	Neural Network–LSTM	BERT–Adam
6	BERT–BERT-base	BERT–LSTM	Llama–QWEN	Neural Network–CNN	LLMs–RLHF
7	CNN–LSTM	BERT–BiLSTM	Llama–MISTRAL	Neural Network–RNN	BERT–Softmax
8	BART–T5	ELMo–GloVe	BART–T5	RNN–GRU	LLMs–Transformer
9	LSTM–RNN	HMM–IBM	GPT-4o–GPT-4o-mini	ChatGPT–GPT-3.5	LLMs–RAG
10	BERT–ChatGPT	GPT-4o–GPT-4o-mini	BERT–ChatGPT	BERT–RoBERTa	Neural Network–Softmax
11	BERT–GPT-2	BERT–MLP	DeepSeek-V3–GPT-4o	BERT–BERT-base-uncased	SGD–Adam
12	CNN–RNN	CRF–LSTM	BERT–GPT-2	ChatGPT–GPT-4o	LLMs–LoRA
13	GPT-4–Llama	BERT–GPT-2	GPT-4–Llama	LLMs–DeepSeek	LSA–SVD
14	GloVe–Word2vec	ChatGPT–GPT-4	Llama-3.1-8B–QWEN2.5-7B	Word2vec–Skip-Gram	LSTM–Softmax
15	BERT–T5	BiLSTM–MLP	CLAUDE-3.5-Sonnet–GPT-4o	T5–T5-base	LLMs–Self-Reflection
16	BERT–XLNET	BiLSTM–CNN	GEMINI–GPT-4	ChatGPT–GPT-3.5-turbo-0301	SVM–Bag-of-Words
17	ChatGPT–GPT-3.5-turbo	LSTM–RNN	GloVe–Word2vec	ChatGPT–GPT-4o-mini	LDA–Gibbs Sampling
18	mBERT–XLM-R	GRU–LSTM	BERT–T5	T5–FLAN-T5	BERT–Transformer
19	GRU–LSTM	LSTM–MLP	mBERT–XLM-R	Decision Tree–C4.5	CFG–CKY
20	Llama-3.1-8B–QWEN2.5-7B	BERT–CRF	BERT–XLNET	ChatGPT–GPT-3	HMM-EM

We further analyzed high-population algorithm pairs under different relational frameworks. It should be noted that Synonym relations are excluded because they refer to a single algorithm entity, while Other relations are not considered due to their relatively ambiguous semantics. As for the Collaborate relation, representative algorithm pairs reflect the joint use of different algorithms and modules. For example, BiLSTM–CRF captures a common modular solution in NLP tasks. CNN–RNN and ELMo–GloVe suggest that algorithms within related categories can also be combined due to their technical complementarity. In addition, different algorithms may collaborate by taking different roles in a larger task pipeline. For instance, “GPT-4o serves as the inference and knowledge LLM to generate training data, while GPT-4o-mini is employed as the evaluator,” which is not the traditional joint modeling but reflects a division of labor within the same workflow.

Representative algorithm pairs under the Compare relation mainly involve comparison and substitution between algorithms with similar functions, shared objectives, or overlapping application scenarios. Pairs such as BERT–RoBERTa, BART–T5, and mBERT–XLM-R all represent competition within the same category. Some competitive relations also cut across different technological paradigms, such as BERT–ELMo, BERT–GPT-2, and BERT–T5. These pairs reflect sustained comparisons between methods based on different modeling ideas, generations, or pre-training mechanisms.

The Superclass relation mainly reflects hierarchical and genealogical structures among algorithms. Some pairs capture hierarchical relations among traditional neural network models, such as RNN–LSTM, RNN–GRU. Another Superclass group is especially prominent among PLMs and LLMs, where a base model often gives rise to multiple configuration variants, task-adapted versions, or improved versions, forming a relatively clear model lineage. Therefore, the Superclass relation does not only indicate an abstract “is-a” relationship; it also reflects the serialized development pattern that has become common in modern model evolution.

Representative algorithm pairs under the Use relation mainly reflect dependency, incorporation, and invocation between algorithms. Some pairs, such as BERT–softmax, reflect the dependence of a main model on key components. Other pairs, such as LLMs–LoRA, suggest the main algorithm uses other algorithms not as internal components but as means to enhance performance. In addition, frequent interactions among traditional

algorithm pairs, such as SVM–bag-of-words, LDA–Gibbs sampling, indicate that algorithm implementation requires support from external resources or procedures.

4.3 The evolution of the algorithm relation network

In this section, we regarded relations as a key mechanism shaping algorithm communities and moved from scattered algorithm pairs to connected algorithm networks. By examining network expansion, tie accumulation, and renewal dynamics, we investigated how different semantic relations contribute to the organization of algorithm groups. Following the same rationale, Other and Synonym were not included in the network analysis in this section.

(1) Evolution of network size across different relation types

We first examined the yearly changes in the number of nodes and edges across different relation networks. Unlike the frequency-based analysis in Section 4.2, nodes and edges indicate the extent to which a given semantic relation connects algorithm entities, and whether such connections gradually form a stable and expanding algorithm community. Figure 2 shows the changing trends in the number of nodes and edges across different relation networks. Overall, all algorithm networks constructed from semantic relations experienced a transition from sparsity to expansion. In the early stage, all relation networks had relatively few nodes and edges, with noticeable fluctuations. After 2000, all relation networks began to expand steadily, but their growth rates and magnitudes varied. For a long period after 2000, the sizes of the four kinds of networks remained relatively close, suggesting that different types of algorithm interactions jointly contributed to the formation of algorithmic knowledge structures during this period. However, after 2018, the Compare network showed a much faster increase in both nodes and edges. This result indicates that the expansion of algorithm networks is not merely a consequence of the increasing number of algorithms, but also reflects the strengthening of connections and interactions among them.

(2) Evolution of internal connectivity across different relation networks

Building on the growth of network size, we further examine the connectivity among nodes within relation networks. Specifically, we calculate the average degree and clustering coefficient of each network. As shown in Fig. 3, the average degree of all relation networks was generally low in the early stage, with considerable fluctuations, indicating that early networks were small in scale and unstable in their internal connection structures. After 2000, the average degree began to increase steadily. In most years, the Compare network had a higher average degree and gradually developed a relatively stable leading position. Around 2016, the gaps between the Compare and Collaborate networks and the other two networks became increasingly clear, while after 2020, the Compare network rapidly widened its lead. The clustering coefficient further reveals the deepening connectivity. Similar to the average degree, the clustering coefficients of all relation networks were relatively unstable in the last century and entered a fluctuating upward trend after 2000. However, the differences among networks were smaller than those observed for the average degree. Compared with other networks, the Compare network still showed a higher clustering coefficient and continued to rise in recent years, suggesting that triangular connections have increasingly formed within its local structure. As a result, formerly scattered competitive relations gradually develop into denser local comparison clusters.

(3) Evolution of renewal dynamics across different relation networks

We further use the Jaccard similarity of edge sets between adjacent years, the proportion of newly added edges, and the proportion of disappearing edges, to analyze the structural stability and renewal patterns of different relation networks. As shown in Fig. 4(a), the Jaccard similarity of edge sets was generally low in the early stage and fluctuated substantially across all relation networks. A gradual increase in similarity began around 2000, but the overall level remained relatively low, peaking around 2018 and then gradually declining. We can suspect that, as algorithm networks expanded, more relational edges began to persist across adjacent years. However, the specific relations between algorithms still showed strong renewal and fluidity. Among the different relation types, the Compare network exhibited relatively higher edge similarity after 2000, while the other networks were lower than or close to it in most years.

The proportions of newly added and disappearing edges further reveal the frequent renewal of algorithm relation networks. As shown in Fig. 4(b) and (c), both proportions remained high for all relation networks over a long period, especially in the early stage. Although the ratios declined after 2000, they remained at a relatively high level and entered a new stage of growth after 2020. It is worth noting that the Compare network showed a lower proportion of disappearing edges than the other networks over the past two decades. This finding is consistent with the results for edge similarity and average degree. It implies that algorithms in competitive relations may repeatedly appear as classic baselines and be compared with newly proposed algorithms. By contrast, Use and Collaborate relations are more likely to depend on the specific task, model design, and application context of each paper, and therefore show weaker year-to-year stability in their edge structures.

Overall, all relation networks show a dynamic pattern of high entry, high disappearance, and low overlap. These algorithm relation networks are not static accumulations of knowledge, but dynamic systems that continuously absorb new algorithm pairs and new relational connections. The evolution of algorithm knowledge structures is therefore not a single-directional process, but a differentiated development across relation types and structural dimensions.

5 Discussion

5.1 Sources of errors in algorithm relation identification

In this study, we compared the performance of different models on the task of algorithm entity relation identification. The best-performing model was SciBERT, with an F1 of 77.97%. Since existing relation identification tasks have mostly been conducted on different public corpora, and few studies have directly focused on relations between algorithm entities, we compared our results with several related studies on method entity relation identification, whose overall F1 scores are approximately 75% (Luan et al., 2018; Z. Wan et al., 2023; Zhong & Chen, 2021). In comparison, our approach achieves reasonable performance on this multi-class semantic classification task. To further improve algorithm entity relation identification in future work, we analyzed error cases in the training and testing stages and summarized the main causes of inaccurate predictions.

(1) The model deviates from the specific context

This type of error mainly occurs when LLMs are used to judge relations. Because such models contain rich prior knowledge, they may directly infer the relation between two algorithms based on their general knowledge, rather than relying on the specific textual context. For example:

Our approach is by nature based on the hybrid semi-markov conditional random fields (HSCRFs) proposed by Ye and Ling (2018), which connect traditional CRFs (Lafferty et al., 2001) and semi-markov CRFs (Sarawagi and Cohen, 2005) by simultaneously leveraging token-level and segment-level scoring information.

For the entity pair “CRFs–semi-markov CRFs” in this sentence, the original context indicates that previous researchers connected traditional CRFs and semi-markov CRFs to construct hybrid semi-markov conditional random fields. Therefore, the relation between the two entities should be Collaborate. However, the model predicts Compare, because it regards CRFs and semi-markov CRFs as algorithms of the same type that may substitute for or compete with each other in solving similar tasks. Although the prompt explicitly asks the model to determine the relation based on the given context, such context-ignoring errors still occur.

(2) Interference from multiple algorithm entity pairs

In academic texts, a single sentence often contains multiple pairs of algorithm entities. When many algorithm entities appear in the same sentence, the model needs to distinguish multiple entity-pair relations within similar contexts, which increases the difficulty of relation identification. For example:

We also report results for UTD-NB and UTD-SVM (Rink and Harabagiu, 2012), which rely on lexical pattern classification based on naive Bayes and support vector machine classifiers, respectively.

This sentence contains four algorithm entities: UTD-NB, UTD-SVM, naive Bayes, and support vector machine. The correct interpretation is that UTD-NB uses naive Bayes, and UTD-SVM uses the support vector machine, while no direct relation exists between other cross-combinations. However, because the sentence contains four algorithm entities, six possible entity pairs, and multiple plausible relation types, the model may incorrectly assign Use relations to unrelated pairs. Although the phrase “rely on” clearly signals a use relation, identifying which entity uses which component requires not only sentence-level semantics, but also additional disambiguation based on the meanings of the algorithm entities themselves. As a result, the model incorrectly infers that UTD-NB also uses the support vector machine, and that UTD-SVM also uses naive Bayes.

(3) Implicit relation expressions

Another source of error lies in the implicit expression of relations. Different papers use different writing styles, and the same relation type can be expressed in various ways. When a sentence does not contain explicit cue words, the model may fail to identify the correct relation. For example:

The main components of our model are: a character-level Bidirectional LSTM for character encoding, a soft alignment mechanism via optimal transport, and a powerful CNN for scoring alignments.

This sentence indicates that Bidirectional LSTM and CNN are both components of the same model. Therefore, the relation between them should be classified as Collaborate relation. However, because the sentence lacks explicit words such as “combine,” “integrate,” or “connect,” the model may infer that there is no clear semantic relation between the two entities.

5.2 From co-occurrence to semantic relation networks

This study investigates interactions among algorithm entities by identifying semantic relations between algorithm pairs. Previous studies on algorithm co-occurrence have inferred major interaction patterns from high-frequency co-occurring algorithm pairs (Y. Wang et al., 2025). For example, C4.5 and decision tree may co-occur because of a hierarchical relation, and support vector machine and C4.5 may co-occur because of competition. However, our findings show that algorithm interactions are much richer than simple co-occurrence. More importantly, the same algorithm pair may carry different relations in different contexts, which cannot be distinguished by the co-occurrence relations.

Based on various kinds of relations, we further constructed relation-specific algorithm interaction networks. Compared with traditional co-occurrence networks, semantic relation networks reveal not only how algorithms are connected, but also how NLP research practices are changing. The Compare network shows the strongest expansion, connectivity, and structural integration, suggesting that more algorithms are being incorporated into a shared comparison space. It can also be suspected that algorithm interactions in NLP increasingly take the form of performance comparison and method substitution, rather than only method construction or technical use. With the spread of benchmarks, leaderboards, and standardized evaluation frameworks, new methods must not only prove that they work but also show which methods they outperform, by how much, and on which tasks. However, whether the increasing investment in multi-algorithm comparison leads to substantial progress, or merely maintains relative positions and produces a possible “Red Queen effect,” remains an important question for future research.

Overall, algorithm relation networks reflect not only technological structures, but also research practices, writing conventions, and evaluation logics. The strengthening of Compare relations may be related to model substitution and technological competition, but also to the increasing requirement for baseline comparison and benchmark-based evaluation. Similarly, the decline of Collaborate relations does not necessarily mean that algorithmic collaboration has weakened; it may also reflect the growing encapsulation of methods in the large-model era and the less explicit description of complex pipelines in papers. Therefore, relation-network analysis helps us understand not only how algorithms are connected, but also how academic communities organize and represent technological relations.

6 Conclusions

Taking NLP as an example, this study proposes an analytical framework moving from algorithm entity extraction to semantic relation identification and relation-based analysis. Specifically, we constructed a semantic relation schema for algorithm entities, evaluated various models’ performance in relation extraction, and examined interaction intensity and network evolution across different relation types. The results show that algorithm entities in scholarly papers mainly exhibit six types of semantic relations: Compare with, Use, Cooperate with, Same as, Superclass of, and Other. In the automatic relation extraction task, LLMs achieved competitive performance, but fine-tuning the SciBERT model remains the preferred approach given both efficiency and computational cost. In the NLP field, semantic relations between algorithms are dominated by competitive relations, supplemented by cooperation, use, and hierarchical inclusion. Representative algorithm pairs with high interaction intensity under different relation types reveal diverse underlying mechanisms of algorithm interaction. The relation-specific algorithm interaction networks further show that algorithm interactions have evolved toward larger scale, deeper connectivity, and faster renewal. The Compare network shows the most significant expansion, reflecting an increasingly competitive ecosystem among algorithms in the field.

In general, by shifting from isolated entity identification to structured relation identification, this study provides a useful reference for extracting and organizing relational knowledge among other types of scholarly entities. However, as an initial study, our research has several limitations. Although full-text papers from the three top NLP conferences are highly representative, they do not cover the entire NLP field. For algorithm entity pair identification, further improvement is still needed in post-identification filtering and entity disambiguation. For algorithm relation identification, we only explored selected LLMs and optimization strategies, while more recent methods remain to be examined. In the future, we will expand the disciplinary and data scope, explore more advanced entity and relation identification methods, and further seek an optimal solution for algorithm relation classification. On this basis, semantic relations can also be used to support more fine-grained algorithm evaluation and recommendation.

Declarations

Disclosure of Interests

The authors have no competing interests to declare that are relevant to the content of this article.

The data from this ongoing project are not publicly available at this time. Researchers with a genuine scientific need for the data are encouraged to contact the corresponding author.

Author Contribution

Y.W. collected and processed the data, performed the analysis, prepared the visualizations, wrote the main manuscript text, and reviewed the manuscript. Y.Z. and H.Z. contributed to the study design and reviewed the manuscript. W.W. and H.Z. contributed to the methodology. All authors approved the final manuscript.

Acknowledgments.

This study was funded by the youth project of the National Social Science Fund of China, titled “Identifying the Semantic Relation and Evolutionary Paths of Algorithm Entities Based on the Full-text of Academic Papers” (grant number. 24CTQ027).

Data Availability

The data used in this study include publicly available open-source datasets and datasets constructed by the authors. The datasets generated and/or analyzed during the current study are available from the corresponding author upon reasonable request.

References

1. Bileschi, M. L., Belanger, D., Bryant, D. H., Sanderson, T., Carter, B., Sculley, D., Bateman, A., DePristo, M. A., & Colwell, L. J. (2022). Using deep learning to annotate the protein universe. *Nature Biotechnology*, 40(6), 932–937. <https://doi.org/10.1038/s41587-021-01179-w>
2. Chen, L., Xu, S., Zhu, L., Zhang, J., Lei, X., & Yang, G. (2020). A deep learning based method for extracting semantic information from patent documents. *Scientometrics*, 125(1), 289–312. <https://doi.org/10.1007/s11192-020-03634-y>
3. Chen, Y., Zeng, Z., Lin, X., Du, X., Rida, I., & Xiao, R. (2026). FDEPCA: A novel adaptive nonlinear feature extraction method via fruit fly olfactory neural network for IOMT anomaly detection. *IEEE Journal of Biomedical and Health Informatics*, 30(1), 27–38. <https://doi.org/10.1109/JBHI.2023.3318892>
4. Duttax, B., & Patel, J. (2024). Algorithm metadata vocabulary: A representational model and metadata vocabulary for describing and maintaining algorithms. *Journal of Information Science*, 50(5). <https://dl.acm.org/doi/10.1177/01655515221116557>
5. Fu, Y., Peng, H., Sabharwal, A., Clark, P., & Khot, T. (2023). *Complexity-Based Prompting for Multi-Step Reasoning* (arXiv:2210.00720). arXiv. <https://doi.org/10.48550/arXiv.2210.00720>
6. Gábor, K., Buscaldi, D., Schumann, A.-K., QasemiZadeh, B., Zargayouna, H., & Charnois, T. (2018). SemEval-2018 Task 7: Semantic Relation Extraction and Classification in Scientific Papers. In *Proceedings of the 12th International Workshop on Semantic Evaluation* (pp. 679–688). Association for Computational Linguistics. <https://doi.org/10.18653/v1/S18-1111>
7. Handschuh, S., & QasemiZadeh, B. (2014). The ACL RD-TEC: A dataset for benchmarking terminology extraction and classification in computational linguistics. In *Proceedings of the 4th International Workshop on Computational Terminology (Computerm)* (pp. 52–63). Association for Computational Linguistics and Dublin City University. <https://doi.org/10.3115/v1/W14-4807>
8. Jiang, Y., Li, J., & Chen, H. (2024). Relation classification via bidirectional prompt learning with data augmentation by large language model. In *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024)* (pp. 13885–13897). ELRA and ICCL. <https://aclanthology.org/2024.lrec-main.1212>
9. Kuhn, T. S. (1996). *The Structure of Scientific Revolutions*. University of Chicago Press.
10. Lei, Z., Wang, D., Catalano, G., Daraio, C., Gregori, M., Moed, H. F., & Ruocco, G. (2019). Model entity extraction in academic full text based on deep learning. In *Proceedings of the 17th International Conference on Scientometrics & Informetrics (ISSI 2019), Vol. II* (pp. 2732–2733). International Society for Scientometrics and Informetrics–ISSI.
11. Liu, Y., Peng, X., Du, T., Yin, J., Liu, W., & Zhang, X. (2024). ERA-CoT: Improving Chain-of-Thought through entity relationship analysis. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)* (pp. 8780–8794). Association for Computational Linguistics. <https://doi.org/10.18653/v1/2024.acl-long.476>
12. Luan, Y., He, L., Ostendorf, M., & Hajishirzi, H. (2018). Multi-task identification of entities, relations, and coreference for scientific knowledge graph construction. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing* (pp. 3219–3232). Association for Computational Linguistics. <https://doi.org/10.18653/v1/D18-1360>
13. Lyu, S., Wu, X., Li, J., Chen, Q., & Chen, H. (2022). Do models learn the directionality of relations? a new evaluation: relation direction recognition. *IEEE Transactions on Emerging Topics in Computational Intelligence*, 6(4), 883–892. <https://doi.org/10.1109/TETCI.2021.3136598>
14. Moscato, V., Napolano, G., Postiglione, M., & Sperli, G. (2023). Multi-task learning for few-shot biomedical relation extraction. *Artificial Intelligence Review*, 56(11), 13743–13763. <https://doi.org/10.1007/s10462-023-10484-6>
15. Rashid, A. B., & Kausik, M. A. K. (2024). AI revolutionizing industries worldwide: A comprehensive overview of its diverse applications. *Hybrid Advances*, 7, 100277. <https://doi.org/10.1016/j.hybadv.2024.100277>
16. Raza, M., Jahangir, Z., Riaz, M. B., Saeed, M. J., & Sattar, M. A. (2025). Industrial applications of large language models. *Scientific Reports*, 15(1), 13755. <https://doi.org/10.1038/s41598-025-98483-1>
17. Rohatgi, S., Qin, Y., Aw, B., Unnithan, N., & Kan, M.-Y. (2023). The ACL OCL Corpus: Advancing open science in computational linguistics. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing* (pp. 10348–10361). Association for Computational Linguistics. <https://doi.org/10.18653/v1/2023.emnlp-main.640>
18. Safder, I., Sarfraz, J., Hassan, S.-U., Ali, M., & Tuarob, S. (2017). Detecting target text related to algorithmic efficiency in scholarly big data using recurrent convolutional neural network model. In *International conference on Asian digital libraries* (pp. 30–40). Springer International Publishing. https://doi.org/10.1007/978-3-319-70232-2_3
19. Saritha, M. S., Babu, P. V., & Lakshmi, A. R. (2025). Using deep learning to extract algorithmic metadata from entire academic documents. In *Proceedings of Advances in Electrical and Computer Technologies* (pp. 413–420). CRC Press.

20. Suissa, O., Elmalech, A., & Zhitomirsky-Geffet, M. (2022). Text analysis using deep neural networks in digital humanities and information science. *Journal of the Association for Information Science and Technology*, 73(2), 268–287. <https://doi.org/10.1002/asi.24544>
21. Tran, Q., Thanh, N. X., Anh, N. H., Hai, N. L., Le, T., Ngo, L. V., & Nguyen, T. H. (2024). Preserving generalization of language models in few-shot continual relation extraction. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing* (pp. 13771–13784). Association for Computational Linguistics. <https://doi.org/10.18653/v1/2024.emnlp-main.763>
22. Tuarob, S., Bhatia, S., Mitra, P., & Giles, C. L. (2013). Automatic detection of pseudocodes in scholarly documents using machine learning. In *Proceedings of the 2013 12th International Conference on Document Analysis and Recognition* (pp. 738–742). IEEE. <https://doi.org/10.1109/ICDAR.2013.151>
23. Tuarob, S., Bhatia, S., Mitra, P., & Giles, C. L. (2016). AlgorithmSeer: A system for extracting and searching for algorithms in scholarly big data. *IEEE TRANSACTIONS ON BIG DATA*, 2(1), 3–17. <https://doi.org/10.1109/TBDATA.2016.2546302>
24. Tuarob, S., Mitra, P., & Giles, C. L. (2012). Improving algorithm search using the algorithm co-citation network. In *Proceedings of the 12th ACM/IEEE-CS Joint Conference on Digital Libraries* (pp. 277–280).
25. Tuarob, S., Mitra, P., & Giles, C. L. (2014). Building a search engine for algorithms. *ACM SIGWEB Newsletter*, 5.
26. Vuković, D. B., Dekpo-Adza, S., & Matović, S. (2025). AI integration in financial services: A systematic review of trends and regulatory challenges. *Humanities and Social Sciences Communications*, 12(1), 562. <https://doi.org/10.1057/s41599-025-04850-8>
27. Wadhwa, S., Amir, S., & Wallace, B. (2023). Revisiting relation extraction in the era of large language models. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)* (pp. 15566–15589). Association for Computational Linguistics. <https://doi.org/10.18653/v1/2023.acl-long.868>
28. Wan, H., Moens, M.-F., Luyten, W., Zhou, X., Mei, Q., Liu, L., & Tang, J. (2016). Extracting relations from traditional Chinese medicine literature via heterogeneous entity networks. *Journal of the American Medical Informatics Association: JAMIA*, 23(2), 356–365. <https://doi.org/10.1093/jamia/ocv092>
29. Wan, Z., Cheng, F., Mao, Z., Liu, Q., Song, H., Li, J., & Kurohashi, S. (2023). GPT-RE: In-context learning for relation extraction using large language models. In *Proceedings of the 2023 conference on empirical methods in natural language processing* (pp. 3534–3547). Association for Computational Linguistics. <https://doi.org/10.18653/v1/2023.emnlp-main.214>
30. Wang, H., Fu, T., Du, Y., Gao, W., Huang, K., Liu, Z., Chandak, P., Liu, S., Van Katwyk, P., Deac, A., Anandkumar, A., Bergen, K., Gomes, C. P., Ho, S., Kohli, P., Lasenby, J., Leskovec, J., Liu, T.-Y., Manrai, A., ... Zitnik, M. (2023). Scientific discovery in the age of artificial intelligence. *Nature*, 620(7972), 47–60. <https://doi.org/10.1038/s41586-023-06221-2>
31. Wang, X., Wei, J., Schuurmans, D., Le, Q., Chi, E., Narang, S., Chowdhery, A., & Zhou, D. (2023). *Self-Consistency Improves Chain of Thought Reasoning in Language Models* (arXiv:2203.11171). arXiv. <https://doi.org/10.48550/arXiv.2203.11171>
32. Wang, Y., Xiang, Y., & Zhang, C. (2024). Exploring motivations for algorithm mention in the domain of natural language processing: A deep learning approach. *Journal of Informetrics*, 18(4), 101550. <https://doi.org/10.1016/j.joi.2024.101550>
33. Wang, Y., & Zhang, C. (2018). Using full-text of research articles to analyze academic impact of algorithms. In *Proceedings of iConference 2018* (pp. 395–401). Cham: Springer International Publishing. https://doi.org/10.1007/978-3-319-78105-1_43
34. Wang, Y., & Zhang, C. (2019). Finding more methodological entities from academic articles via iterative strategy: A preliminary study. In *Proceedings of International Society for Scientometrics and Informetrics 2019* (pp. 2702–2703).
35. Wang, Y., & Zhang, C. (2020). Using the full-text content of academic articles to identify and evaluate algorithm entities in the domain of natural language processing. *Journal of Informetrics*, 14(4), 101091. <https://doi.org/10.1016/j.joi.2020.101091>
36. Wang, Y., Zhang, C., Song, M., Kim, S., Ko, Y., & Lee, J. (2025). Exploring academic influence of algorithms by co-occurrence network based on full-text of academic papers. *Aslib Journal of Information Management*, 77(4), 651–680. <https://doi.org/10.1108/AJIM-09-2023-0352>
37. Wang, Y., Zhang, H., & Zhang, C. (2021). Algorithm entities usage in chinese academic articles from the domain of information science. In *Proceedings of International Society for Scientometrics and Informetrics 2021* (pp.1559–1560).
38. Wei, X., Cui, X., Cheng, N., Wang, X., Zhang, X., Huang, S., Xie, P., Xu, J., Chen, Y., Zhang, M., Jiang, Y., & Han, W. (2024). *ChatIE: Zero-Shot Information Extraction via Chatting with ChatGPT* (arXiv:2302.10205). arXiv. <https://doi.org/10.48550/arXiv.2302.10205>
39. Wu, F., Dwivedi, V. P., & Leskovec, J. (2025). Large language models are good relational learners. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)* (pp. 7835–7854). Association for Computational Linguistics. <https://doi.org/10.18653/v1/2025.acl-long.386>
40. Wu, R., & Ardakani, S. P. (2025). Towards scientific knowledge graphs: dependency graph analysis using graph neural networks for extracting scientific relations. *Electronics*, 14(11), 2276. <https://doi.org/10.3390/electronics14112276>
41. Yu, J.-Y., Yang, D., & Lee, K.-H. (2025). Scilinkbert: A BERT-style language model for understanding scientific texts with citations. *The Journal of Supercomputing*, 81(15), 1356. <https://doi.org/10.1007/s11227-025-07838-y>
42. Zha, H., Chen, W., Li, K., & Yan, X. (2019). Mining algorithm roadmap in scientific publications. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining* (pp. 1083–1092). Association for Computing Machinery. <https://doi.org/10.1145/3292500.3330913>

43. Zhang, H., Liu, Y., Liu, X., Liang, T., Sharma, G., Xue, L., & Guo, M. (2024). Sentence bag graph formulation for biomedical distant supervision relation extraction. *IEEE Transactions on Knowledge and Data Engineering*, 36(9), 4890–4903. <https://doi.org/10.1109/TKDE.2024.3377229>

44. Zhong, Z., & Chen, D. (2021). A frustratingly easy approach for entity and relation extraction. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies* (pp. 50–61). Association for Computational Linguistics. <https://doi.org/10.18653/v1/2021.naacl-main.5>

Footnotes

- 1. <https://aclanthology.org/>
- 2. <https://github.com/docling-project/docling>

Figures

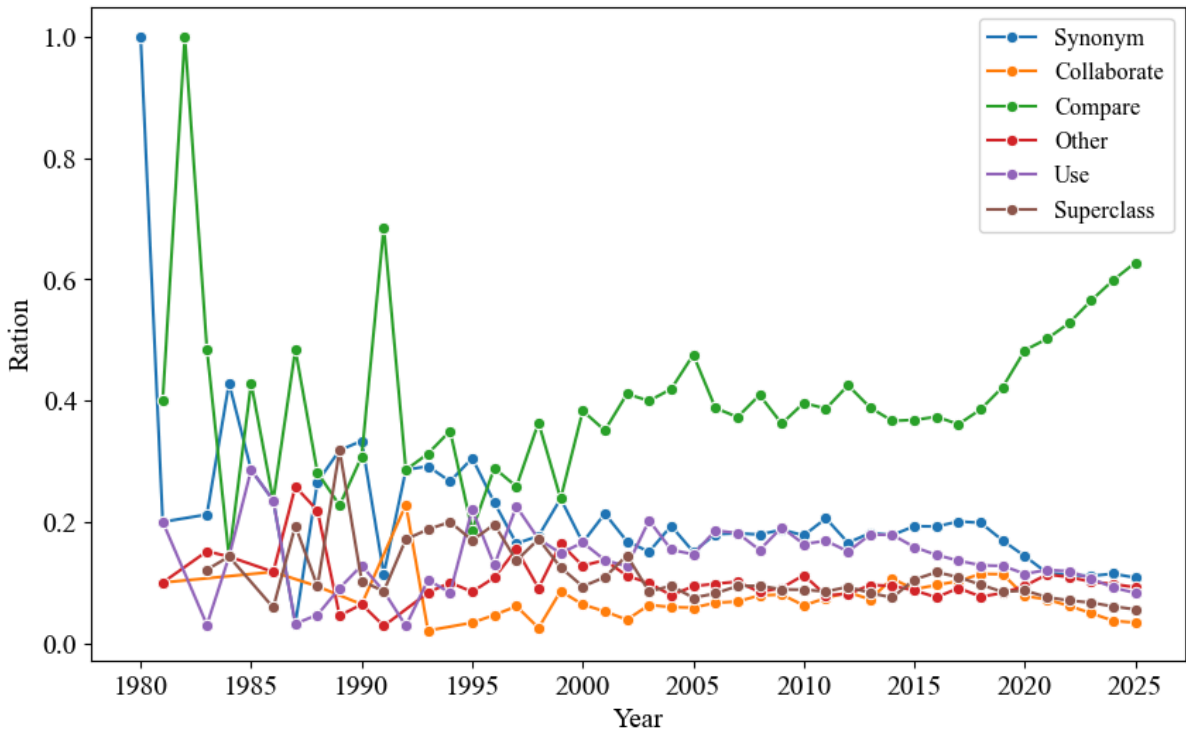
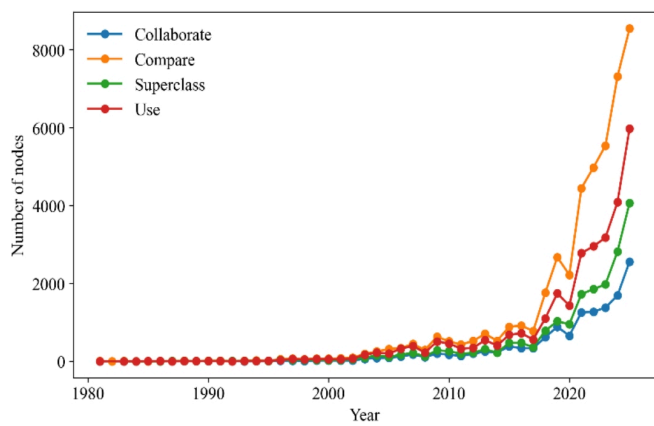
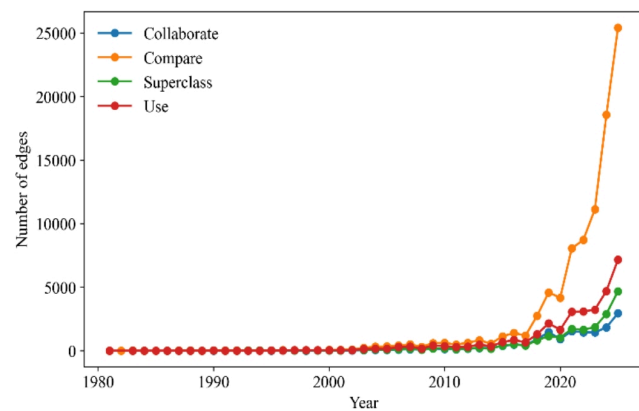


Figure 1

Temporal changes in the proportions of different relation types



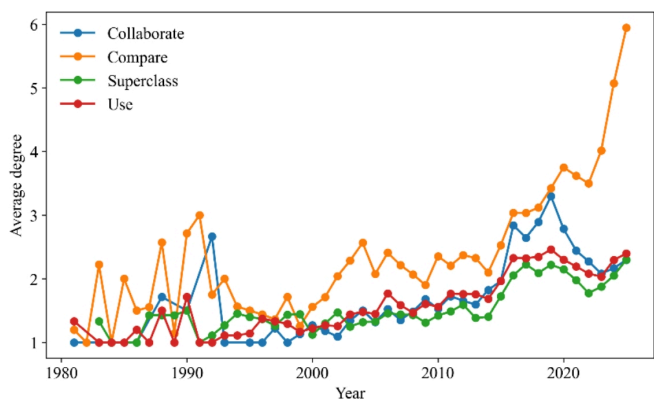
(a)



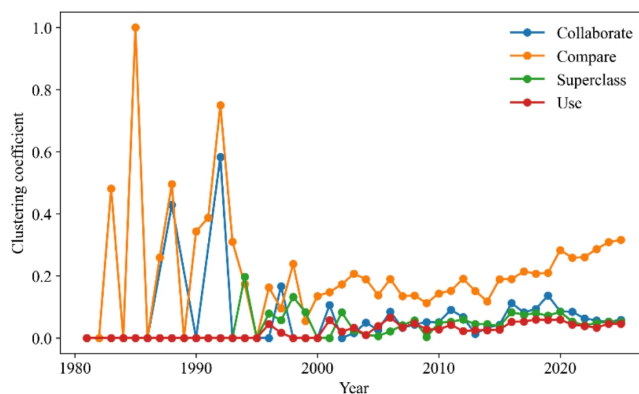
(b)

Figure 2

Evolution of algorithm relation network sizes



(a)



(b)

Figure 3

Evolution of algorithm relation network connectivity

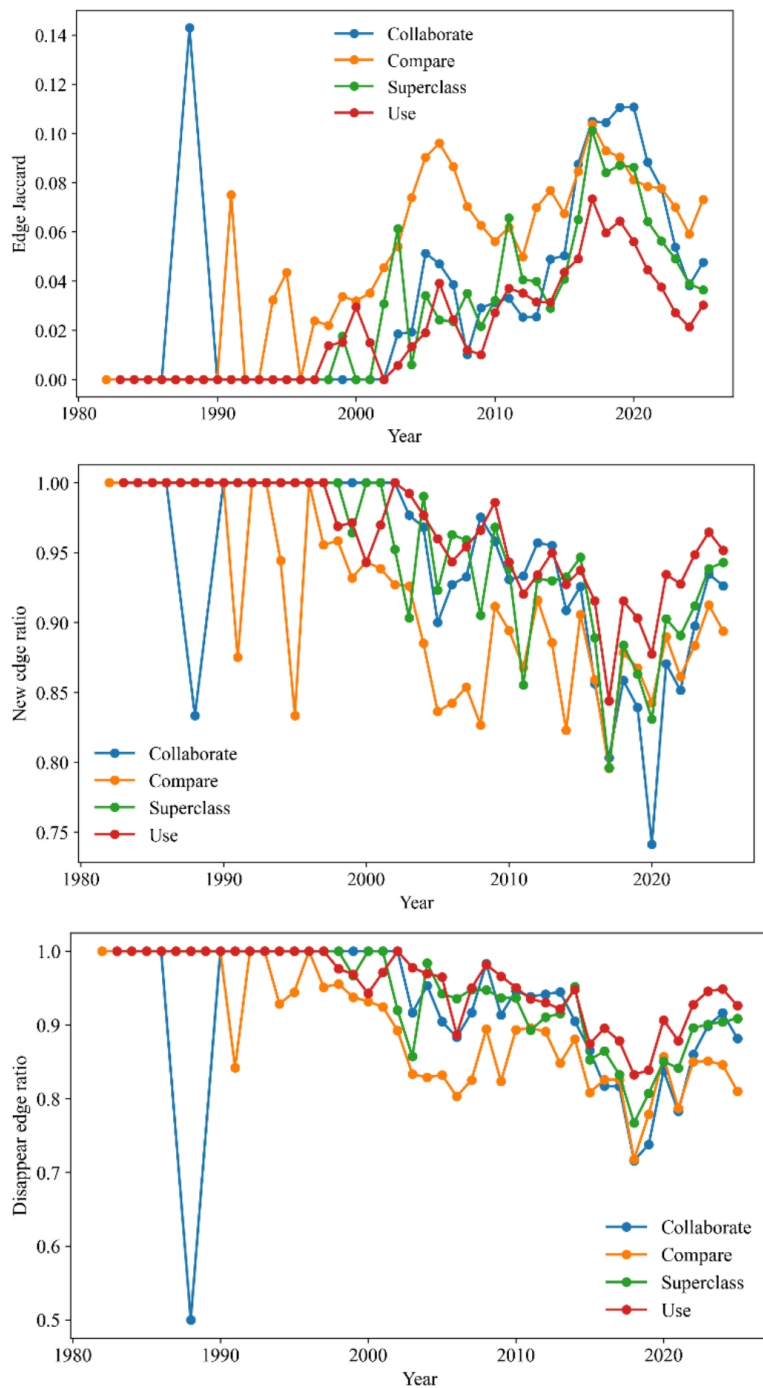


Figure 4

Structural renewal dynamics of different relation networks