

Article

Ontology Matching Method Based on Deep Learning and Syntax

Jiawei Lu and Changfeng Yan * 

School of Mechanical and Electrical Engineering, Lanzhou University of Technology, Lanzhou 730050, China; 18816200508@163.com

* Correspondence: changf_yan@163.com

Abstract

Ontology technology addresses data heterogeneity challenges in Internet of Everything (IoE) systems enabled by Cyber Twin and 6G, yet the subjective nature of ontology engineering often leads to differing definitions of the same concept across ontologies, resulting in ontology heterogeneity. To solve this problem, this study introduces a hybrid ontology matching method that integrates a Recurrent Neural Network (RNN) with syntax-based analysis. The method first extracts representative entities by leveraging in-degree and out-degree information from ontological tree structures, which reduces training noise and improves model generalization. Next, a matching framework combining RNN and N-gram is designed: the RNN captures medium-distance dependencies and complex sequential patterns, supporting the dynamic optimization of embedding parameters and semantic feature extraction; the N-gram module further captures local information and relationships between adjacent characters, improving the coverage of matched entities. The experiments were conducted on the OAEI benchmark dataset, where the proposed method was compared with representative baseline methods from OAEI as well as a Transformer-based method. The results demonstrate that the proposed method achieved an 18.18% improvement in F-measure over the best-performing baseline. This improvement was statistically significant, as validated by the Friedman and Holm tests. Moreover, the proposed method achieves the shortest runtime among all the compared methods. Compared to other RNN-based hybrid frameworks that adopt classical structure-based and semantics-based similarity measures, the proposed method further improved the F-measure by 18.46%. Furthermore, a comparison of time and space complexity with the standalone RNN model and its variants demonstrated that the proposed method achieved high performance while maintaining favorable computational efficiency. These findings confirm the effectiveness and efficiency of the method in addressing ontology heterogeneity in complex IoE environments.

Keywords: IOE; ontology matching; deep learning; syntax; OAEI



Academic Editor: Alberto Abelló

Received: 27 June 2025

Revised: 8 August 2025

Accepted: 11 August 2025

Published: 14 August 2025

Citation: Lu, J.; Yan, C. Ontology Matching Method Based on Deep Learning and Syntax. *Big Data Cogn. Comput.* **2025**, *9*, 208. <https://doi.org/10.3390/bdcc9080208>

Copyright: © 2025 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

Mobile data in the Internet of Everything (IoE) faces a range of challenges, including time delays and high energy consumption [1]. To address these problems, next-generation network technologies such as 6G and Cyber Twins are being introduced and are regarded as key enablers for achieving efficient and reliable wireless interconnections [2]. However, the integration of these network technologies with IoE devices results in a heterogeneous environment, where diverse devices exchange data across various infrastructures. Due to the lack of semantic information, such data exchange often leads to interoperability problems caused by semantic heterogeneity [3]. Ontologies can model data and knowledge in a

machine-interpretable manner [4], thereby addressing data heterogeneity and enhancing interoperability among 6G-based Cyber Twin-driven IoE systems. Nevertheless, during ontology construction, the lack of standardized naming conventions for concepts means that semantically identical concepts may be described differently, leading to ontology heterogeneity problems [5]. To address this challenge, ontology matching is employed to identify equivalent entities across heterogeneous ontologies.

Figure 1 presents an example of two heterogeneous ontologies and their matching results. In the figure, O_1 and O_2 represent the ontologies to be matched, and each bidirectional dashed arrow connects two heterogeneous classes, “ $\equiv$ ” indicating an equivalence relationship between them. For instance, “Input” in O_1 is identified as equivalent to “SensorInput” in O_2 , with a similarity value of 0.8 obtained through a specific similarity measurement method.

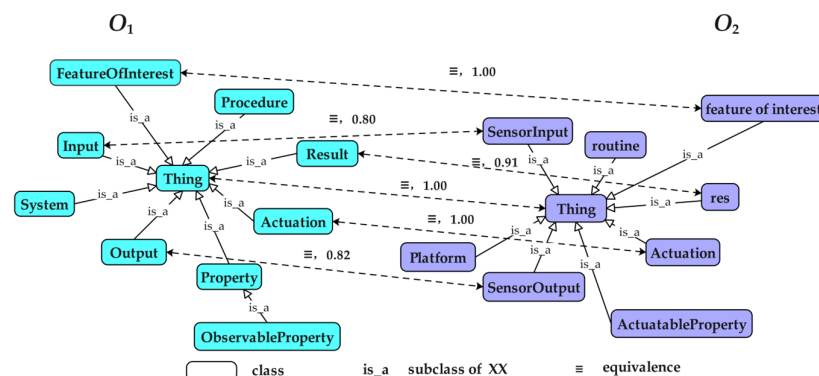


Figure 1. Example of heterogeneous ontologies and matching results.

Similarity measurement method is a critical component of ontology matching, as it is used to compute similarity values between candidate entities. Existing similarity measurement methods can be broadly classified into four categories: syntax-based, semantics-based, structure-based, and instance-based methods. To handle different types of heterogeneity, researchers often combine multiple similarity measurement methods [6]. Although such combinations can improve the recall of matching results to some extent, they also introduce two major problems. First, combining different similarity measurement methods leads to the construction of multiple similarity matrices for the same entity pair, significantly increasing computational complexity and time consumption. Second, because different similarity measurement methods exhibit distinct feature distributions for the same matching task, their sensitivity to filtering thresholds also varies, which may degrade the quality of the matching results and limit their applicability in real-world scenarios.

To address the above challenges, intelligent algorithms represented by deep learning have increasingly been applied to ontology matching, aiming to embed and integrate features such as entity syntax, semantics, and structure into unified vector spaces [7–10]. Deep learning models, by learning the intrinsic relationships among these feature distributions, can automatically identify optimal aggregation parameters through end-to-end training, thereby enhancing the accuracy of matching results [11,12]. Compared with the traditional approach of constructing independent similarity matrices for individual features, deep learning significantly reduces computational overhead and alleviates the complexity and time consumption issues inherent in multi-matrix aggregation frameworks [9,13]. Additionally, deep models’ ability to unify various feature types into joint representations helps mitigate inconsistent threshold sensitivity that typically plagues hybrid similarity methods, leading to improved alignment quality and practical applicability [10,14,15].

Despite these advantages, current deep learning-based ontology matching methods still have notable limitations. Most approaches primarily focus on entity labels for semantic

representation, while the rich semantics contained in other properties (e.g., comments) are often underutilized, thereby limiting the depth of semantic understanding [16,17]. Furthermore, although recent methods tend to employ character-level or subword embedding models (such as Char-embeddings) to address out-of-vocabulary problems, these representations frequently lack the semantic association between words, making it difficult to capture context-dependent entity similarity, especially in cross-domain or noisy settings [9,18]. As a result, the expressiveness and generalization of the learned entity embeddings are often insufficient for handling complex real-world heterogeneity. Therefore, a key challenge remains: how to leverage both structural cues and diverse lexical-semantic properties to construct robust, context-sensitive entity embeddings that generalize across ontologies with varied granularity and lexical heterogeneity.

To address these limitations, we enhance the variety of entity attributes used to capture richer semantic information and employ pretrained word embeddings as the base vectors for entities to strengthen semantic associations between words. To better embed domain-specific semantic information of entities into pre-trained word vectors, this paper proposes an efficient ontology matching approach by combining Recurrent Neural Networks (RNNs) with the N-gram similarity measurement method. RNNs capture medium-range dependencies and complex semantic patterns, while N-gram focuses on extracting local character-level features. The complementary strengths of both techniques enable a more comprehensive semantic and syntactic analysis of entities in heterogeneous ontologies, thereby facilitating more accurate equivalence evaluation. The contributions of this paper are summarized as follows:

1. A representative entity extraction method is proposed, which selects higher-priority training entities by leveraging their in-degree and out-degree within the ontology's hierarchical structure. This strategy effectively reduces noise during model training and enhances the generalization capability of the model.
2. By exploiting the RNN's ability to model medium-range dependencies and its relatively low number of learnable parameters, the model achieves rapid dynamic optimization of embedding parameters. This enables efficient extraction of semantic features from multiple annotation properties of the candidate entities, allowing multi-perspective evaluation of entity equivalence without the need to select or integrate other similarity measurement methods.
3. A hybrid ontology matching framework combining RNN and syntax-based similarity is proposed. After performing rapid semantic matching through RNN, the N-gram similarity measurement method is employed to capture sequence fragment information and co-occurrence patterns between them. This provides rich contextual information for the microstructure of entity strings and enables a secondary similarity assessment for unmatched entity pairs, thereby further improving the recall of the matching results.

2. Related Work

In recent years, learning-based ontology matching, including both machine learning and deep learning approaches, has gradually become a research hotspot. Typically, such methods formulate ontology matching as a classification or regression problem [9,17,19]. For instance, Xue et al. [9] framed ontology matching as a regression task and adopted a Siamese neural network to capture the semantic features of concepts. This model emphasized both the similarity and dissimilarity of entity pairs during training, improving its generalization ability. Similarly, Wang et al. [20] transformed biomedical ontologies into a pair-wise connectivity graph and applied graph convolutional networks to propagate similarity features for predicting matching concept pairs, treating the task as a binary classi-

fication problem, with evaluations on biomedical ontologies from the Ontology Alignment Evaluation Initiative.

Deep learning techniques have been increasingly applied to handle complex structural and semantic aspects. Iyer et al. [11] treated the matching problem as a regression task in their VeeAlign approach. Based on relationships between entities and their contexts, they categorized contexts into multiple types and applied a dual-attention mechanism to compute context vectors. These were concatenated with semantic vectors of entity labels to form enriched representations incorporating upper and lower contextual semantics, reducing reliance on external knowledge bases. Bento et al. [14] approached ontology matching as a binary classification task using a Convolutional Neural Network (CNN) to learn semantic representations of entity labels, while incorporating embeddings of subclasses and superclasses to model structural knowledge. Jiang et al. [10] employed a Long Short-Term Memory (LSTM) network to match complex biomedical ontologies by capturing long-range contextual semantic information of entities, achieving robust alignment performance. Chakraborty et al. [21] proposed OntoConnect, an unsupervised ontology alignment approach using a recursive neural network that learns concept representations from structural meta-information, framing matching as a regression problem, and predicting alignments by combining word and meta similarities.

Further advancements incorporate graph neural networks and large language models for enhanced generalization. He et al. [22] explored large language models for ontology alignment, formulating the matching task as classification through prompt engineering in a zero-shot setting, and evaluated their approach on biomedical ontologies. Giglou et al. [23] presented the LLMs4OM framework for matching ontologies with large language models. The framework leverages retrieval-augmented generation and enhances matching via zero-shot prompting across three ontology representations: concept, concept-parent, and concept-children, and is evaluated on 20 OAEI datasets from various domains, outperforming traditional systems in complex scenarios. Hertling and Paulheim [24] used large language models for ontology matching, treating matching as a prompting task with LLMs to learn entity similarities on OAEI tracks. Chen et al. [25] proposed an ontology matching method based on a gated graph attention model, which dynamically adjusts the fusion ratio of semantic and structural embeddings during training to achieve high-quality and adaptive alignment of heterogeneous ontology concept pairs, thereby significantly improving matching performance. He et al. [26] introduced a BERT-based system for ontology alignment, formulating matching as a sequence classification task enhanced by deep language model embeddings, tested on biomedical ontologies.

It is important to note that most of these methods rely heavily on entity labels to learn semantic information, often overlooking the semantic value of other properties [17,20]. Additionally, recent methods [10,14] commonly use character-level embedding models to convert strings into machine-readable vectors. While this facilitates the encoding of out-of-vocabulary words, the resulting representations often lack semantic relevance between words, limiting accurate word similarity assessment. Xue et al. [19] highlighted this in word embedding-based heterogeneous entity matching on the Web of Things, where character-level approaches fall short in capturing deeper semantics.

To overcome these limitations, this paper expands the variety of attributes used for determining entity equivalence and incorporates pretrained word embeddings to enhance semantic associations. An ontology matching method is proposed by combining Recurrent Neural Networks (RNNs) with N-gram similarity, where RNNs capture medium-range semantic dependencies and N-gram extracts local character-level features, enabling more comprehensive semantic and syntactic analysis.

3. Basic Concepts

3.1. Ontology and Ontology Matching

An ontology is defined as a triple $O = (C, P, I)$ [27], where C denotes the set of concepts, P denotes the set of properties (including data properties and object properties), and I denotes the set of instances. These three components are collectively referred to as entities. Entities have annotation properties including ID, label, and comment. Specifically, ID is the identifier of the entity, label is the name of the entity, and comment is the description of the entity. To address ontology heterogeneity, it is necessary to identify semantically similar entities between heterogeneous ontologies through the ontology matching process.

The ontology matching process [28] can be formalized as a function m . Given two ontologies to be matched, O_1 and O_2 (where O_1 is the source ontology and O_2 is the target ontology), a reference alignment R , some external resources r used during ontology matching (e.g., using WordNet as a background knowledge base in the ontology matching process; or employing certain intermediate ontologies as anchors), and a set of parameters p (such as aggregation weights and filtering thresholds), ultimately yielding an alignment A : $A = m(O_1, O_2, R, r, p)$. Figure 2 shows the ontology matching process.

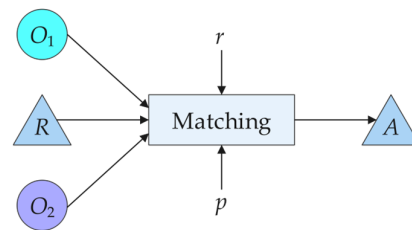


Figure 2. Matching process.

Where the alignment A can be represented as a set of quadruples (e, e', n, L) . The e and e' denote entities from the source and target ontologies, respectively. The value $n \in [0, 1]$ indicates the similarity value between the entity pair obtained via a certain similarity measurement method. When $n = 0$, the entities are completely dissimilar; when $n = 1$, they are fully equivalent. L denotes the semantic relationship between the two entities, which refers to equivalence in this paper.

3.2. Alignment Evaluation Metrics

The quality of an alignment is typically evaluated using three standard metrics: precision, recall, and F-measure [29], all of which range from 0 to 1. Given a reference alignment, precision is defined as the ratio of correct alignments identified by the ontology matching method to the total number of alignments it produces. While precision reflects the accuracy of the matching process, it does not guarantee that all correct alignments have been retrieved. Therefore, recall is needed to provide a more comprehensive assessment, which is defined as the ratio of correct alignments identified by the method to the total number of alignments in the reference alignment. The F-measure is the weighted harmonic mean of precision and recall, offering a balanced evaluation of the alignment quality. Generally, *precision*, *recall*, and *F-measure* are defined as follows:

$$precision = \frac{|R \cap A|}{|A|} \quad (1)$$

$$recall = \frac{|R \cap A|}{|R|} \quad (2)$$

$$F\text{-measure} = \frac{2 \times recall \times precision}{recall + precision} \quad (3)$$

where R represents reference alignments and A represents the alignments found by the method.

3.3. Recurrent Neural Network

Since Elman introduced the Simple Recurrent Network in 1990, Recurrent Neural Network (RNN) models have undergone significant development and have been widely applied in natural language processing, speech recognition, and time series prediction [30]. In the context of ontologies, annotation properties of entities typically appear as short sequences. Due to their simple architecture and relatively small number of trainable parameters, RNNs are well-suited for efficiently extracting semantic information from such sequences.

Compared to the newer generation of more sophisticated models such as Transformers (e.g., BERT or OntoBERT), RNNs capture medium-range dependencies via temporal hidden state propagation, which aligns well with the nature of ontology representations like labels and comments. In contrast, Transformer models use self-attention mechanisms to model long-range dependencies in parallel and are capable of capturing more complex contextual semantics [31]. However, they entail higher model complexity and larger parameter sizes, which may lead to overfitting on small-scale datasets such as those in the OAEI benchmark. OntoBERT [32] introduces domain-specific enhancements through ontology-guided entity and relation definitions, but it also imposes additional computational overhead. In light of these considerations, this study prioritizes the use of RNNs to strike a balance between performance and scalability. As shown in Figure 3, an RNN unrolls along the time axis at time step t .

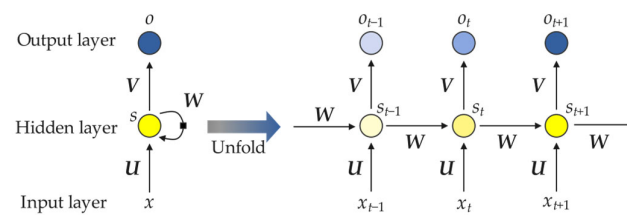


Figure 3. RNN unfolding at time t according to the timeline.

From the left side of Figure 3, it can be seen that an RNN consists of an input layer, a hidden layer, and an output layer. Here, U denotes the weight matrix from the input layer to the hidden layer; V denotes the weight matrix from the hidden layer to the output layer; and W denotes the weight matrix from the hidden layer to itself. As shown on the right side of Figure 3, at time step t , the RNN receives input x_t , and the output of the output layer is o_t , while the hidden state is s_t . The hidden state s_t is determined not only by the current input x_t , but also by the output of the hidden layer at the previous time step s_{t-1} . The definitions of s_t and o_t are as follows:

$$s_t = f(U \bullet x_t + W \bullet s_{t-1} + b_1) \quad (4)$$

$$o_t = g(V \bullet s_t) \quad (5)$$

where f denotes the activation function of the hidden layer, which is typically nonlinear, such as the tanh or ReLU function. The purpose of introducing nonlinear activation functions is to create nonlinear decision boundaries, enhance the expressive capability of the model, and compensate for the limitations of linear models. The function g represents the activation function of the output layer, which is often chosen as the sigmoid function, enabling the classification results to be represented in the form of probabilities.

3.4. Syntax-Based Similarity Measure

Deep learning-based ontology matching methods typically utilize character or word embeddings trained on co-occurrence relationships [33]. This means that the probability of matching a target entity depends on its co-occurrence frequency with the source entity. However, in real-world scenarios, non-equivalent entities may still exhibit high co-occurrence frequencies, which can substantially compromise the alignment quality by making it heavily dependent on the quality of the training data. To address this problem, a syntax-based similarity measurement method is employed to capture sequence fragment information and co-occurrence patterns between them. As a well-known syntax-based similarity measurement method, N-gram computes similarity values based on the co-occurrence frequency of overlapping substrings between the candidate entities. Given two strings s_1 and s_2 , with N set to 3, the N-gram is defined as follows:

$$N\text{-gram} = \frac{2 \times \text{comm}(s_1, s_2)}{N_{s_1} + N_{s_2}} \quad (6)$$

where $\text{comm}(s_1, s_2)$ represents the number of common substrings of s_1 and s_2 . N_{s_1} and N_{s_2} are the number of substrings in s_1 and s_2 , respectively.

4. Hybrid Ontology Matching Method Based on RNN and N-Gram

4.1. Construction of Training Dataset

This paper employs the benchmark dataset from the Ontology Alignment Evaluation Initiative (OAEI) to evaluate the effectiveness of the proposed matching method. The dataset exhibits a rich variety of heterogeneities, enabling a comprehensive assessment of ontology matching methods. Specifically, the dataset comprises four major categories of heterogeneous ontologies, each containing multiple specific cases with different types of heterogeneity. Moreover, each case includes a target ontology and its corresponding reference alignment. Table 1 provides a brief description of the heterogeneity types in each case.

Table 1. Overview of benchmark dataset.

Case	Ontology Description	Case	Ontology Description
101	Source ontology	241	No hierarchy + no instance + no property
201	No names	246	Flattened hierarchy + no instance + no property
202	No names, no comment	247	Expanded hierarchy + no instance + no property
221	No hierarchy	248	Scrambled labels + no comments + no hierarchy
222	Flattened hierarchy	249	Scrambled labels + no comments + no instance
223	Expanded hierarchy	250	Scrambled labels + no comments + no property
224	No instances	251	Scrambled labels + no comments + flattened hierarchy
225	No restrictions	252	Scrambled labels + no comments + expanded hierarchy
228	No properties	253	Scrambled labels + no comments + no hierarchy + no instance
232	No hierarchy + no instance	254	Scrambled labels + no comments + no hierarchy + no property
233	No hierarchy + no property	257	Scrambled labels + no comments + no instance + no property
236	No instance + no property	258	Scrambled labels + no comments + flattened hierarchy + no instance
237	Flattened hierarchy + no instance	259	Scrambled labels + no comments + expanded hierarchy + no instance
238	Expanded hierarchy + no instance	261	Scrambled labels + no comments + expanded hierarchy + no property
239	Flattened hierarchy + no property	262	Scrambled labels + no comments + no hierarchy + no instance + no property
240	Expanded hierarchy + no property		

These cases in Table 1 can be categorized as follows: (1) Case 101 represents the source ontology, which is used for matching with all other ontologies; (2) In cases 201–202, entities in the target ontologies are described using different terminologies from those in the source ontology; (3) In cases 221–247, the target ontologies have different hierarchical structures

compared to the source ontology; (4) In cases 248–262, the target ontologies differ from the source ontology in both terminologies and hierarchical structures. Table 2 clearly presents the number of entities and corresponding alignments for each of the four case categories.

Table 2. Number of entities and alignments in benchmark cases.

Case Pair	Number of Entities	Number of Alignments
101–101	109	97
201–202	1090	970
221–247	1538	1194
248–262	5610	4325
Total	8347	6586

Before training the RNN-based ontology matching model, it is necessary to construct a training dataset. To reduce noise during model training and improve the model’s generalization performance, representative reference alignments from each case are adopted in the training data. Considering that an ontology typically exhibits a tree-like structure, where entities are closely connected, the sum of the out-degree and in-degree [34] of an entity can reflect its importance within the ontology. For instance, in Figure 1, the entity “Property” has an out-degree plus in-degree of 2, indicating a higher degree of importance compared to other entities and making it more likely to be selected as a representative entity. When constructing the training dataset, the entities are first ranked in descending order based on the sum of their out-degree and in-degree, and the top 30% of entities are empirically selected as representative entities. Next, reference alignments related to these representative entities are selected as positive samples. Then, as a constraint, negative samples are constructed by randomly replacing one entity in the selected reference alignments with another entity. The number of negative samples is kept equal to that of the positive samples.

4.2. Hybrid Ontology Matching Framework Based on RNN and N-Gram

In the proposed method, the RNN-based ontology matching component formulates the ontology matching task as a binary classification problem, determining the equivalence relationship between candidate entity pairs based on their semantic information. Previous learning-based methods typically use entity labels as matching targets, while overlooking the semantic value of comments that can provide explanatory information about the entities. In this work, both the label and comment are jointly utilized to extract richer semantic information from entities.

To extract semantic information from entities, it is necessary to encode them into machine-recognizable numerical vectors. Common encoding methods include One-Hot encoding [35], Bag of Words [36], and Char-embeddings. Although One-Hot encoding is relatively simple, it only captures the presence of semantic information for entities without reflecting the semantic relationships between them. Moreover, the resulting numerical vectors are overly sparse, which can easily lead to the curse of dimensionality. Similarly, Bag of Words encoding generates vector dimensions based on the dictionary size and suffers from sparsity problems as well. Char-embeddings is a character embedding model trained on the GloVe 840B/300D dataset (available at <https://github.com/minimaxir/char-embeddings>, accessed on 5 March 2025). Although it can provide numerical vector representations for out-of-vocabulary entities, it is not effective at capturing the semantic information of words. In light of the varying degrees of limitations in the expressive capacity of the above encoding methods, this paper adopts the pre-trained word2vec-google-news-300 word embedding model (available at <https://code.google.com/archive/p/word2vec>,

accessed on 6 March 2025) to enhance alignment performance. Trained on approximately 10 billion words from the Google News corpus, this model effectively captures the semantic features of words, thereby contributing to improved semantic representation in the alignment process.

However, the word2vec-google-news-300 model is trained based on word co-occurrence patterns. As a result, even semantically dissimilar words may have similar vector representations if they frequently co-occur, which can negatively impact alignment accuracy. To address this problem, this paper employs the N-gram method, which calculates similarity values between entity pairs to be matched by capturing the co-occurrence probability of common substrings between them. This method enables a secondary similarity computation for entity pairs that cannot be evaluated by RNNs, thereby further improving the recall of the matching results.

Figure 4 illustrates the technical framework of the proposed hybrid ontology matching method based on RNN and N-gram, where the training and testing processes, guided by black and blue connecting lines, respectively. In terms of the specific steps, Steps 1–4 correspond to the training phase, and Steps 5–11 correspond to the testing phase. In particular, during testing, a one-to-one matching strategy is adopted, where each entity in the source ontology is only allowed to be matched to one entity in the target ontology, and vice versa.

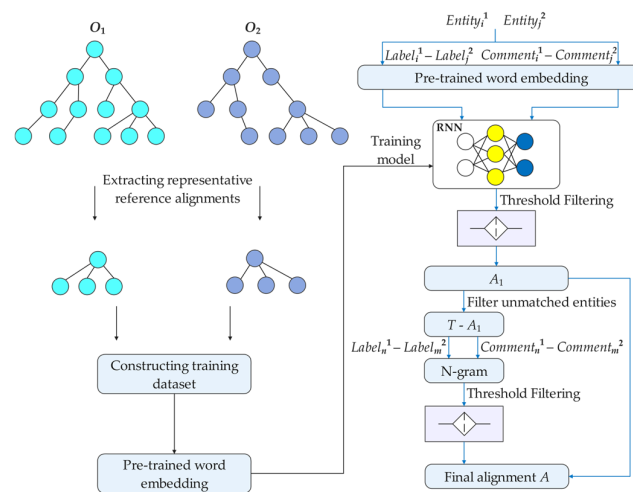


Figure 4. Technical framework of the hybrid ontology matching method based on RNN and N-gram.

- Step 1. Extract representative reference alignments from the ontologies to be matched in the case set;
- Step 2. Construct positive and negative training samples based on the representative reference alignments;
- Step 3. Perform pre-trained word embedding on the training samples;
- Step 4. Feed the samples with completed word embedding into the model for training, and obtain the trained model;
- Step 5. Perform the cartesian product on the three types of entities (classes, object properties, and data properties) in the ontologies to be matched within the test data. For each type of entity, extract the entity pairs to be matched $Entity_i^1$ ($i = 1, 2, 3 \dots$ and $i \leq |Entity^1|$) and $Entity_j^2$ ($j = 1, 2, 3 \dots$ and $j \leq |Entity^2|$) from its cartesian product, and then construct the annotation property pairs $Label_i^1 - Label_j^2$ and $Comment_i^1 - Comment_j^2$ for them.
- Step 6. Perform pre-trained word embedding on the two annotation property pairs to obtain the embedding vectors of label and comment, respectively;

- Step 7. Feed the embedding vectors of annotation property pairs into the trained model to extract semantic information and obtain their similarity values;
- Step 8. In the respective similarity matrices for label and comment, if both the source and target annotation properties are present, the maximum of the two similarity values is taken as the similarity value for the corresponding entity pair. Otherwise, the similarity value of the existing annotation property (present in both source and target) is used as the similarity value. Then, for each entity in the source ontology, the entity in the target ontology with the highest similarity value is selected. Based on a predefined filtering threshold, the equivalence of the entity pair is evaluated, and the matched pairs are added to set A_1 ;
- Step 9. Obtain the not-yet-matched entities by eliminating the already-matched entity pairs A_1 from the test set T ;
- Step 10. Use N-gram to match the remaining unmatched entities by first performing similarity computation and threshold filtering on the label, and then on the comment for entities that remain unmatched. The resulting matched entity pairs are added to set A_2 .
- Step 11. Take a merge of A_1 and A_2 , and achieve the final alignment set A .

5. Experimental Results and Analysis

To evaluate the effectiveness of the proposed method, experiments are conducted on the benchmark dataset introduced in Section 3. The performance is compared with several representative methods from the OAEI (available at <https://oaei.ontologymatching.org>, accessed on 6 March 2025), including AML [37], the LogMap family (LogMap and LogMapBio) [38], and PhenoMF [39], as well as the Transformer-based [31] method. Then, to assess the applicability of the N-gram method within the hybrid ontology matching framework, we replace it with two classic similarity measurement methods: SimRank (based on structure) [40] and Wu-Palmer (based on semantics) [41] for comparative analysis. Next, we conduct ablation experiments on the proposed method to further demonstrate the contribution of each module to the overall performance. Finally, to further evaluate the efficiency of RNN among lightweight models of the same category, a comparative analysis of its time and space complexity with LSTM [42] and GRU [43] is conducted. Additionally, LSTM and GRU are each combined with the N-gram module to construct LSTM+N-gram and GRU+N-gram hybrid methods, respectively, in order to further analyze the performance and efficiency trade-offs of different model combinations within the proposed framework.

The following provides a brief introduction to the methods and neural network architectures involved:

- AML is an automated ontology matching system primarily based on syntax similarity measurement methods and emphasizes the use of external resources.
- LogMap is a scalable, logic-based ontology matching system that supports (real-time) user interaction during the matching process.
- LogMapBio is an extension of LogMap that uses BioPortal as a dynamic provider of mediating ontologies, rather than relying on a few preselected ones.
- PhenoMF builds upon an extended PhenomeNET ontology enriched with equivalence mappings retrieved through BioPortal and AML.
- Transformer model uses self-attention mechanisms to model long-range dependencies in parallel and is capable of capturing more complex contextual semantics.
- SimRank is an iterative similarity measurement method that calculates the similarity value between two entities by examining the similarity of their neighbors, emphasizing structural relationships in the ontology graph.

- Wu-Palmer is a semantic similarity measurement method based on concept depth. It calculates the similarity value between concepts by comparing their depths and the depth of their least common ancestor within a background knowledge base (WordNet in this study).
- LSTM model utilizes gating mechanisms to selectively retain and forget information over time, enabling the modeling of long-range dependencies in sequential data.
- GRU model employs a relatively simplified gating structure to capture sequential dependencies efficiently, balancing performance and computational complexity.

5.1. Experimental Configuration

In the experimental configuration of this paper, the vector dimension is set to 300 according to the word2vec-google-news-300 model. For each case pair, 30% of the data is used for training and validation, with 70% of that portion allocated to the training set and 30% to the validation set. The remaining 70% of the data is used as the test set. To fully train the model while conserving computational resources, the number of training epochs and the patience value are set to 100 and 5, respectively. The batch size is set to 32 to enhance training stability and prevent overfitting. Other training parameters are configured as follows: maximum sequence length = 100, maximum vocabulary size = 200,000, RNN output dimension = 10, fully connected layer output dimension = 100, dropout rate in RNN input = 0.1, dropout rate in the fully connected layer = 0.1, activation function of the fully connected layer is set to ReLU, and the activation function of the output layer is set to sigmoid, which constrains the output to the range [0, 1] to facilitate binary classification. During alignment evaluation, the filtering threshold is set to 0.98 to avoid false-positive alignments that may affect the final alignment quality. The experiment is conducted in an environment with 3.3 GHz (6 cores) and 8 GB of RAM allocated.

5.2. Results and Analysis

As shown in Table 3, among the 31 test cases, the proposed method achieved equal or higher precision in 139 out of 155 comparisons with the five comparison methods, and was slightly outperformed by AML and Transformer in only 8 cases. This demonstrates that the proposed method is capable of achieving more accurate entity alignment in most cases. AML exhibits slightly better precision in some cases due to its use of dozens of similarity measurement techniques to enhance matching accuracy. However, the interference among these techniques results in lower recall and increased computational complexity. Similarly, the Transformer-based model leverages multi-head self-attention to capture global semantic dependencies and achieves high precision under a relatively strict matching threshold. However, it exhibits limited sensitivity to local sequential patterns and structural perturbations, such as word order variations or incomplete annotations, due to its weak inductive bias toward positional continuity and token order. As a result, many truly correct entity pairs are missed, leading to a significant decline in recall. In contrast, RNNs possess inherent sequential processing capabilities, making them more suitable for modeling local dependencies and maintaining sensitivity to structural variations in input sequences. Combined with a relatively high matching threshold, this strategy effectively improves the confidence of alignment results. Although the precision is slightly lower than that of some baseline methods, the sequential modeling capability enhances the model's robustness in handling entity pairs with word order variations, structural perturbations, or partial lexical overlap, thereby contributing to improved recall.

Table 3. Comparison of precision between the proposed method and OAEI participants.

Case	AML	LogMap	LogMapBio	PhenoMF	Transformer	Ours
101	1.00	0.94	0.50	0.46	1.00	1.00
201	1.00	0.96	0.49	0.00	1.00	1.00
202	1.00	0.95	0.50	0.00	1.00	1.00
221	1.00	0.94	0.50	0.00	1.00	1.00
222	1.00	0.00	0.00	0.00	1.00	1.00
223	1.00	0.94	0.50	0.00	1.00	0.97
224	1.00	0.94	0.50	0.00	1.00	1.00
225	1.00	0.94	0.50	0.00	1.00	0.98
228	1.00	0.85	0.66	0.00	1.00	1.00
232	1.00	0.94	0.50	0.00	1.00	1.00
233	1.00	0.85	0.66	0.00	1.00	1.00
236	1.00	0.85	0.66	0.00	1.00	1.00
237	1.00	0.00	0.00	0.00	1.00	1.00
238	1.00	0.94	0.50	0.00	1.00	0.97
239	1.00	0.85	0.66	0.00	1.00	0.97
240	1.00	0.85	0.66	0.00	1.00	0.88
241	1.00	0.85	0.66	0.00	1.00	1.00
246	1.00	0.85	0.66	0.00	1.00	0.97
247	1.00	0.85	0.66	0.00	1.00	0.88
248	1.00	0.94	0.47	0.00	1.00	1.00
249	1.00	0.94	0.50	0.00	1.00	1.00
250	1.00	0.90	0.66	0.00	1.00	1.00
251	1.00	0.00	0.00	0.00	1.00	1.00
252	1.00	0.96	0.51	0.00	1.00	1.00
253	1.00	0.97	0.49	0.00	1.00	1.00
254	1.00	0.86	0.64	0.00	1.00	1.00
257	1.00	0.88	0.64	0.00	1.00	1.00
258	1.00	0.00	0.00	0.00	1.00	1.00
259	1.00	0.94	0.49	0.00	1.00	1.00
261	1.00	0.86	0.64	0.00	1.00	0.96
262	1.00	0.87	0.65	0.00	1.00	1.00
mean	1.00	0.79	0.50	0.10	1.00	0.99

In Table 4, the recall of the proposed method is relatively lower in cases with scrambled labels and no comments. This is primarily because such benchmark cases rely heavily on the implicit semantics of entity IDs, which are typically meaningless random strings in real-world scenarios. The proposed method intentionally avoids ID-based alignment and instead focuses on aligning based on meaningful label and comment information. In cases where labels are scrambled and comments are missing, the RNN-based semantic model struggles to extract stable semantic features, and the N-gram character-level method may also be ineffective due to the lack of common character fragments. Nonetheless, the proposed method jointly models both labels and comments to capture semantic relationships more comprehensively, and applies an N-gram strategy to complement the alignment of entities that cannot be matched through semantic similarity alone. As a result, the proposed method demonstrates strong generality in various heterogeneous conditions, such as no hierarchies, missing properties, and altered hierarchies, and achieves an average recall that is 11.69% higher than that of LogMap, the second-best performing method.

Table 4. Comparison of recall between the proposed method and OAEI participants.

Case	AML	LogMap	LogMapBio	PhenoMF	Transformer	Ours
101	0.00	0.96	0.56	0.34	0.22	1.00
201	0.28	0.76	0.44	0.00	0.19	0.99
202	0.28	0.76	0.44	0.00	0.15	0.70
221	0.34	0.96	0.56	0.00	0.22	1.00
222	0.33	0.00	0.00	0.00	0.20	1.00
223	0.34	0.95	0.56	0.00	0.22	0.96
224	0.34	0.95	0.56	0.00	0.22	1.00
225	0.34	0.96	0.56	0.00	0.22	0.98
228	1.00	1.00	1.00	0.00	0.12	1.00
232	0.34	0.95	0.56	0.00	0.22	1.00
233	1.00	1.00	1.00	0.00	0.12	1.00
236	1.00	1.00	1.00	0.00	0.12	1.00
237	0.33	0.00	0.00	0.00	0.20	1.00
238	0.34	0.96	0.56	0.00	0.22	0.96
239	1.00	1.00	1.00	0.00	0.07	1.00
240	1.00	1.00	1.00	0.00	0.12	0.88
241	1.00	1.00	1.00	0.00	0.12	1.00
246	1.00	1.00	1.00	0.00	0.07	1.00
247	1.00	1.00	1.00	0.00	0.12	0.88
248	0.29	0.74	0.42	0.00	0.15	0.70
249	0.27	0.75	0.44	0.00	0.15	0.70
250	0.83	0.79	0.79	0.00	0.18	0.70
251	0.27	0.00	0.00	0.00	0.14	0.70
252	0.28	0.76	0.45	0.00	0.15	0.70
253	0.27	0.76	0.44	0.00	0.15	0.70
254	0.83	0.79	0.79	0.00	0.18	0.70
257	0.82	0.79	0.79	0.00	0.18	0.70
258	0.28	0.00	0.00	0.00	0.14	0.70
259	0.28	0.75	0.44	0.00	0.15	0.70
261	0.83	0.79	0.79	0.00	0.18	0.70
262	0.83	0.79	0.79	0.00	0.18	0.70
mean	0.56	0.77	0.61	0.01	0.16	0.86

Table 5 presents Friedman’s test on the alignment quality in terms of F-measure (computed rank in parentheses). Firstly, from the perspective of F-measure, the proposed method demonstrates consistently stable overall performance across all 31 test cases. In particular, it maintains a high F-measure (ranging from 0.81 to 0.82) even in complex scenarios involving multiple overlapping types of ontology heterogeneity (e.g., Cases 248–262), where the performance of competing methods tends to fluctuate significantly, and in some cases, their F-measure drops to 0. Furthermore, from the comparative results across different types of heterogeneous scenarios, the proposed method achieves the highest ranking in three task categories: terminological heterogeneity (Cases 201–202), structural heterogeneity (Cases 221–247), and complex heterogeneity involving both terminological and structural differences (Cases 248–262), with F-measure improvements of 6.5%, 19.88%, and 17.87% over the second-best method, respectively. In terminological heterogeneity scenarios, explicit modeling of labels and comments, combined with pre-trained word embeddings, effectively enhances the perception of semantic differences. In structural heterogeneity scenarios, the contextual modeling capability of RNNs effectively captures hierarchical relationships between entities. In composite heterogeneity scenarios, the synergy of semantic modeling and character-level N-gram matching ensures semantic accuracy while addressing shallow differences such as spelling and abbreviations, thereby comprehensively improving matching performance. Overall, the proposed method achieves the highest average F-measure,

outperforming the second-best method by 18.18%, which strongly validates not only its robustness and generality but also highlights its potential as a reliable solution for ontology alignment tasks across diverse and complex domains.

Table 5. Friedman’s test on the alignment quality in terms of F-measure (computed rank in parentheses).

Case	AML	LogMap	LogMapBio	PhenoMF	Transformer	Ours
101	0.00 (6.0)	0.95 (2.0)	0.52 (3.0)	0.39 (4.0)	0.36 (5.0)	1.00 (1.0)
201	0.44 (4.0)	0.85 (2.0)	0.47 (3.0)	0.00 (6.0)	0.31 (5.0)	0.99 (1.0)
202	0.43 (4.0)	0.85 (1.0)	0.47 (3.0)	0.00 (6.0)	0.27 (5.0)	0.82 (2.0)
221	0.51 (4.0)	0.95 (2.0)	0.52 (3.0)	0.00 (6.0)	0.36 (5.0)	1.00 (1.0)
222	0.50 (2.0)	0.00 (5.0)	0.00 (5.0)	0.00 (5.0)	0.34 (3.0)	1.00 (1.0)
223	0.51 (4.0)	0.94 (2.0)	0.53 (3.0)	0.00 (6.0)	0.36 (5.0)	0.96 (1.0)
224	0.51 (4.0)	0.94 (2.0)	0.53 (3.0)	0.00 (6.0)	0.36 (5.0)	1.00 (1.0)
225	0.51 (4.0)	0.95 (2.0)	0.52 (3.0)	0.00 (6.0)	0.36 (5.0)	0.98 (1.0)
228	1.00 (1.5)	0.92 (3.0)	0.80 (4.0)	0.00 (6.0)	0.22 (5.0)	1.00 (1.5)
232	0.51 (4.0)	0.94 (2.0)	0.53 (3.0)	0.00 (6.0)	0.36 (5.0)	1.00 (1.0)
233	1.00 (1.5)	0.92 (3.0)	0.80 (4.0)	0.00 (6.0)	0.22 (5.0)	1.00 (1.5)
236	1.00 (1.5)	0.92 (3.0)	0.80 (4.0)	0.00 (6.0)	0.22 (5.0)	1.00 (1.5)
237	0.50 (2.0)	0.00 (5.0)	0.00 (5.0)	0.00 (5.0)	0.34 (3.0)	1.00 (1.0)
238	0.51 (4.0)	0.95 (2.0)	0.52 (3.0)	0.00 (6.0)	0.36 (5.0)	0.96 (1.0)
239	1.00 (1.0)	0.92 (3.0)	0.80 (4.0)	0.00 (6.0)	0.13 (5.0)	0.98 (2.0)
240	1.00 (1.0)	0.92 (2.0)	0.80 (4.0)	0.00 (6.0)	0.22 (5.0)	0.88 (3.0)
241	1.00 (1.5)	0.92 (3.0)	0.80 (4.0)	0.00 (6.0)	0.22 (5.0)	1.00 (1.5)
246	1.00 (1.0)	0.92 (3.0)	0.80 (4.0)	0.00 (6.0)	0.13 (5.0)	0.98 (2.0)
247	1.00 (1.0)	0.92 (2.0)	0.80 (4.0)	0.00 (6.0)	0.22 (5.0)	0.88 (3.0)
248	0.44 (3.5)	0.82 (1.5)	0.44 (3.5)	0.00 (6.0)	0.27 (5.0)	0.82 (1.5)
249	0.43 (4.0)	0.84 (1.0)	0.47 (3.0)	0.00 (6.0)	0.27 (5.0)	0.82 (2.0)
250	0.91 (1.0)	0.84 (2.0)	0.71 (4.0)	0.00 (6.0)	0.31 (5.0)	0.82 (3.0)
251	0.43 (2.0)	0.00 (5.0)	0.00 (5.0)	0.00 (5.0)	0.25 (3.0)	0.82 (1.0)
252	0.43 (4.0)	0.85 (1.0)	0.48 (3.0)	0.00 (6.0)	0.27 (5.0)	0.82 (2.0)
253	0.43 (4.0)	0.85 (1.0)	0.47 (3.0)	0.00 (6.0)	0.27 (5.0)	0.82 (2.0)
254	0.90 (1.0)	0.82 (2.5)	0.70 (4.0)	0.00 (6.0)	0.31 (5.0)	0.82 (2.5)
257	0.90 (1.0)	0.83 (2.0)	0.71 (4.0)	0.00 (6.0)	0.31 (5.0)	0.82 (3.0)
258	0.44 (2.0)	0.00 (5.0)	0.00 (5.0)	0.00 (5.0)	0.25 (3.0)	0.82 (1.0)
259	0.44 (4.0)	0.84 (1.0)	0.47 (3.0)	0.00 (6.0)	0.27 (5.0)	0.82 (2.0)
261	0.90 (1.0)	0.82 (2.0)	0.70 (4.0)	0.00 (6.0)	0.31 (5.0)	0.81 (3.0)
262	0.90 (1.0)	0.83 (2.0)	0.71 (4.0)	0.00 (6.0)	0.31 (5.0)	0.82 (3.0)
mean	0.66 (2.60)	0.77 (2.42)	0.54 (3.69)	0.01 (5.81)	0.28 (4.74)	0.91 (1.74)

Secondly, Friedman’s test [44] was applied to evaluate the alignment quality in terms of F-measure, aiming to determine whether significant differences exist among the participating methods. Specifically, under the null hypothesis, Friedman’s test assumes that all evaluated methods exhibit equivalent performance. To reject the null hypothesis, indicating statistically significant performance differences, the computed statistic χ_r^2 must meet or exceed the critical value from the reference chi-square distribution table. This study adopts a significance level of $\alpha = 0.05$, and with six participating methods in the matching tasks, the degrees of freedom correspond to 5, with the critical chi-square value being $\chi_{0.05}^2 = 11.07$. As indicated by the mean ranks in the last row of Table 5, the proposed method achieves an average rank of 1.74 across the 31 test cases, significantly lower than that of other methods, demonstrating its superior average performance. Across the entire set of tasks, Friedman’s test statistic $\chi_r^2 = 105.13$ substantially exceeds the critical value of 11.07, providing strong evidence to reject the null hypothesis of no significant performance differences among methods, thereby confirming the presence of statistically significant performance variations.

Based on the Friedman test results indicating significant performance differences among the evaluated methods, this study further applies the Holm test [45] to assess whether the differences in alignment quality between each of the compared methods and the proposed method are statistically significant. Given that the proposed method achieves the lowest average rank and thus demonstrates the best overall performance, using it as the reference method for multiple comparisons is statistically justified. In the Holm test, the process involves the following steps. First, the z -values are calculated to represent the ranking differences between each of the compared methods and the baseline. Then, based on the standard normal distribution, these z -values are used to derive the unadjusted p -values. Next, these p -values are compared against the adjusted significance thresholds under the significance level $\alpha = 0.05$, computed as $\alpha / (m - i + 1)$, where m is the number of methods compared with the baseline and i denotes the rank position of the unadjusted p -value among all comparisons. This allows us to determine whether the observed performance differences are statistically significant relative to the proposed method.

Table 6 presents the statistical analysis results of the Holm test. The results show that the proposed method is statistically superior to all other ontology matching methods. All comparison methods yield p -values that are significantly lower than their corresponding Holm-adjusted significance thresholds, indicating that these methods are statistically different from the proposed method in terms of alignment performance. For instance, LogMap, which has the lowest z -value among the baselines, yields a p -value of 0.0156, which is far below its threshold of 0.05. Similarly, other methods such as AML, LogMapBio, Transformer, and PhenoMF also produce extremely low unadjusted p -values, all well below their corresponding Holm-adjusted thresholds. These small p -values further confirm that the differences in alignment quality between the proposed method and the compared methods are statistically significant at the 5% level. The consistency of these results across the compared methods further reinforces the statistical advantage of the proposed method, thereby validating the effectiveness of its innovative design for the ontology matching task.

Table 6. Holm’s test on the alignment quality.

i	Method	z -Value	Unadjusted p -Value	$\frac{\alpha}{m - i + 1}, \alpha=0.05$
5	LogMap	2.42	1.56×10^{-2}	0.050
4	AML	2.60	9.40×10^{-3}	0.025
3	LogMapBio	3.69	2.20×10^{-4}	1.67×10^{-2}
2	Transformer	4.74	2.10×10^{-6}	1.25×10^{-2}
1	PhenoMF	5.81	6.30×10^{-9}	1.00×10^{-2}

To comprehensively evaluate the practical usability and computational cost of each ontology matching method, a comparative analysis of runtime across all benchmark cases was conducted, as presented in Table 7.

Table 7. Comparison of run time and F-measure between the proposed method and baseline methods.

Method	F -Measure	Runtime (s)	F -Measure per Second
AML	0.77	120	0.0064
LogMap	0.66	194	0.0034
LogMapBio	0.54	54,439	0.0000
PhenoMF	0.01	1632	0.0000
Transformer	0.28	1132	0.0002
Ours	0.91	116	0.0078

As shown in Table 7, the proposed method achieves the highest F-measure while requiring only 116 s of execution time, which is lower than all other methods, demonstrating its superior runtime efficiency. Furthermore, to comprehensively evaluate the matching quality and processing speed of each method, we adopt an efficiency-effectiveness trade-off metric, namely the F-measure per second. As shown in the fourth column of the figure, the proposed method achieves a score of 0.0078, which is approximately 1.2 times that of AML (0.0064) and 2.3 times that of LogMap (0.0034), and considerably higher than those of LogMapBio, PhenoME, and Transformer. These results indicate that the proposed method not only achieves high alignment accuracy but also maintains strong practicality and scalability, making it well-suited for efficient matching in large-scale ontology matching tasks.

Figures 5–7 present a comparison of the RNN-based hybrid ontology matching methods across four types of heterogeneity cases in the benchmark, in terms of precision, recall, and F-measure, respectively. As shown in Figure 5, all three methods perform well under a high filtering threshold. However, since some of the relevant words for the entities are missing in WordNet, Wu-Palmer fails to provide effective similarity measurements, resulting in lower rankings across all four types of cases. Additionally, in the case of semantically ambiguous entities, all the methods often fail to achieve precise alignment. For example, according to the reference alignment, the source entity (ID: Conference, label: The location of an event, comment: An event presenting work.) is matched to the target entity (ID: Conference, label: Conference, comment: A scientific conference). However, our method performs sequential matching within the similarity matrix, first matches another target entity (ID: ScientificMeeting, label: Scientific meeting, comment: An event presenting work). Although the newly identified entity pair shares an identical comment, and the target entity's comment in the reference alignment is semantically close to the label of the newly identified target entity, the reference alignment does not include this newly identified entity pair. This makes it difficult to identify the matched pairs in the reference alignment under the sequential matching setup.

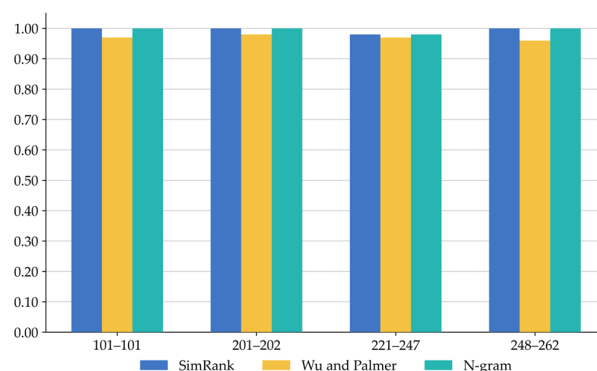


Figure 5. Comparison of precision among the three hybrid methods on four heterogeneity case types.

Figure 6 shows the comparison of recall. Due to the lack of neighboring entities for data properties and object properties in the ontologies, SimRank is unable to compute similarity values for them. In contrast, our method fully leverages the co-occurrence probability of common substrings in the entity pairs to capture the surface-level feature associations between sequence fragment information, thereby maximizing the discovery of potential alignments and improving recall.

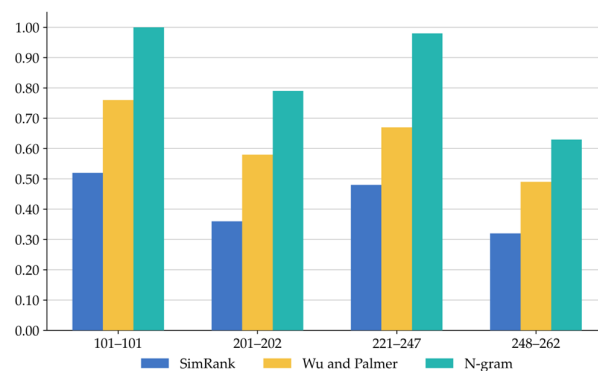


Figure 6. Comparison of recall among the three hybrid methods on four heterogeneity case types.

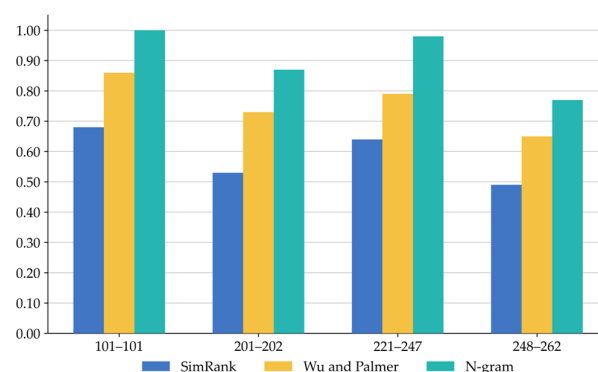


Figure 7. Comparison of F-measure among the three hybrid methods on four heterogeneity case types.

Figure 7 presents the F-measure values of the three hybrid methods across the four dataset types. The proposed method achieves the best matching results on each case set, which reflects the effectiveness and generality of the method. Combined with Figures 5 and 6, the superior F-measure performance of our method is mainly attributed to its high recall, which is enabled by N-gram's ability to identify as many entities as possible based on character-level information, while the high filtering threshold ensures precision, resulting in an overall boost in F-measure. Compared to the SimRank and Wu-Palmer hybrid methods, the proposed method improves the F-measure by 18.46%. This further confirms that syntax-based similarity measurement methods are more effective than those based on structure and semantics in handling surface-level features, thereby improving the quality of the matching results.

As shown in Figure 8, the ablation results indicate that the RNN achieves higher precision, while the N-gram effectively improves recall. These advantages stem from their respective strengths: the RNN leverages its capacity to model complex semantic dependencies in sequential data, enabling the capture of deeper semantic relationships between entities and thereby enhancing precision; in contrast, the N-gram excels at detecting local character patterns within entity texts, allowing identification of additional potential matches and a significant increase in recall. In comparison, the proposed hybrid method maintains a high precision while improving recall, resulting in the optimal overall F-measure. These findings demonstrate that combining sequence modeling with syntactic features facilitates more accurate and comprehensive ontology matching.

Furthermore, to provide an in-depth assessment of the computational efficiency of the standalone RNN model and the proposed method (RNN+N-gram), two commonly used lightweight variants of RNN, namely LSTM and GRU, are included in the comparison. Both their standalone configurations and their integrations with the N-gram module are analyzed in terms of time and space complexity. This results in a total of six methods

being evaluated: RNN, LSTM, GRU, RNN with N-gram, LSTM with N-gram, and GRU with N-gram. In addition, the metric F-measure per second is computed for all methods to jointly reflect their performance in alignment quality and computational efficiency. A summary of the comparative results is provided in Table 8.

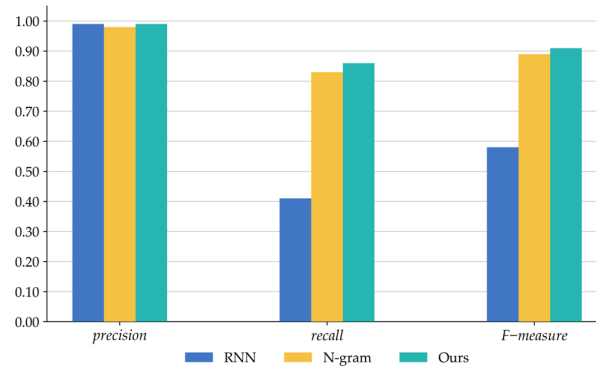


Figure 8. Ablation results for the proposed method and its core components.

Table 8. Time, space, and performance comparison of RNN, LSTM, GRU and their N-gram-enhanced variants.

Method	Time Complexity	Space Complexity	F-Measure	Runtime (s)	F-Measure per Second
RNN	$O(T \bullet L(h \bullet d + h^2))$	$O(T \bullet L + V \bullet d + h \bullet d + h^2)$	0.58	113	0.0051
RNN + N-gram	$O(T \bullet L(h \bullet d + h^2) + G \bullet l \bullet k)$	$O(T \bullet L + V \bullet d + h \bullet d + h^2 + G)$	0.91	116	0.0078
LSTM	$O(T \bullet L \bullet 4(h \bullet d + h^2))$	$O(T \bullet L + V \bullet d + 4(h \bullet d + h^2))$	0.51	155	0.0033
LSTM + N-gram	$O(T \bullet L \bullet 4(h \bullet d + h^2) + G \bullet l \bullet k)$	$O(T \bullet L + V \bullet d + 4(h \bullet d + h^2) + G)$	0.89	158	0.0056
GRU	$O(T \bullet L \bullet 3(h \bullet d + h^2))$	$O(T \bullet L + V \bullet d + 3(h \bullet d + h^2))$	0.53	164	0.0032
GRU + N-gram	$O(T \bullet L \bullet 3(h \bullet d + h^2) + G \bullet l \bullet k)$	$O(T \bullet L + V \bullet d + 3(h \bullet d + h^2) + G)$	0.89	168	0.0053

As shown in Table 8, when processing a total of $T = i \times j$ entity pairs, the RNN model's primary computational cost stems from sequence embedding and sequence modeling. For each pair of entities, two textual fields (i.e., label and comment) are processed independently. Given that the maximum input sequence length is L , the embedding dimension is d , and the hidden representation dimension is h , the overall time complexity of the RNN model can be expressed as $O(T \bullet L(h \bullet d + h^2))$, where $T \bullet L \bullet h \bullet d$ accounts for input-to-hidden transformations and $T \bullet L \bullet h^2$ corresponds to hidden-to-hidden recurrent operations. In the proposed method, based on the RNN matching results, an additional N-gram comparison is introduced to measure the similarity of residual unmatched entity pairs. The time complexity of this step is $O(G \bullet l \bullet k)$, where G denotes the number of residual unmatched entity pairs, l is the average character length of each textual field (label or comment), and k is the window size used in the N-gram process. Since this module is applied only to a relatively small number of unmatched pairs (i.e., $G < T$), the additional time cost remains limited. This is further supported by the runtime comparison shown in the table, where the proposed method introduces only a 3 s increase in runtime, indicating that the computational overhead is relatively low and confirming a favorable trade-off between performance and efficiency.

Similarly, for the standalone LSTM model, each time step involves four computational submodules: the input gate, forget gate, output gate, and candidate state update. Although strictly speaking, only three of them are gating mechanisms, all four submodules involve independent matrix multiplication operations. As a result, each time step requires four matrix multiplications, making the overall time complexity approximately four times that of the RNN, i.e., $O(T \bullet L \bullet 4(h \bullet d + h^2))$. Building on this, the LSTM + N-gram method incorporates an additional character-level N-gram module to further match residual unmatched entity pairs. The time complexity of this additional step is $O(G \bullet l \bullet k)$. Although this module

introduces additional computational overhead, its impact remains limited since it also only operates on a relatively small subset of unmatched pairs (i.e., $G < T$). For the standalone GRU model, each time step consists of the update gate, reset gate, and candidate hidden state computation, which correspond to three matrix multiplications. Therefore, the time complexity of GRU is approximately three times that of the RNN, i.e., $O(T \bullet L \bullet 3(h \bullet d + h^2))$. Similarly, the GRU + N-gram method extends the GRU-based model with the same N-gram module, resulting in a total time complexity of $O(T \bullet L \bullet 3(h \bullet d + h^2)) + O(G \bullet l \bullet k)$.

In terms of space complexity, the RNN model primarily involves the storage of input sequences, the embedding matrix, and model parameters, resulting in a total space complexity of $O(T \bullet L + V \bullet d + h \bullet d + h^2)$, where V denotes the vocabulary size. The term $T \bullet L$ corresponds to the memory used for storing input sequences, $V \bullet d$ accounts for the embedding matrix, and $h \bullet d + h^2$ represents the number of weight parameters in the RNN cell, including input-to-hidden and hidden-to-hidden connections. For the proposed method, the additional space cost mainly arises from storing the unmatched entity pairs and their temporary processing results, which is approximately $O(G)$. However, since the N-gram matching process is conducted in a pairwise manner and does not require the construction or storage of large-scale matrices, this additional memory overhead is minimal and does not significantly increase the overall memory consumption. In summary, while the proposed hybrid method introduces a slight increase in computational complexity, it substantially improves the recall and overall performance of ontology alignment, demonstrating a favorable balance between effectiveness and resource cost.

In comparison, for the standalone LSTM model, the space complexity increases due to the additional parameters required by the gating mechanisms, which include the input gate, forget gate, output gate, and the candidate state update. Each of these submodules requires distinct sets of weight matrices for input and recurrent connections, along with corresponding biases. As a result, the overall space complexity becomes $O(T \bullet L + V \bullet d + 4(h \bullet d + h^2))$, which is approximately four times the parameter-related memory of the RNN model. In the LSTM + N-gram configuration, the added N-gram component incurs a temporary memory cost of approximately $O(G)$, similar to the RNN + N-gram setting. However, since the matching is restricted to a small number of unmatched entity pairs and no large-scale matrices are constructed, the memory overhead remains minor. For the GRU model, which combines the reset gate, update gate, and candidate hidden state into a more compact structure, the space complexity is moderately reduced compared to LSTM. Specifically, it requires three sets of input and recurrent weight matrices and biases, yielding a total space complexity of $O(T \bullet L + V \bullet d + 3(h \bullet d + h^2))$. As with the LSTM + N-gram, the GRU + N-gram method introduces an additional $O(G)$ memory cost due to the character-level N-gram computation.

As shown in the fourth column of Table 8, the F-measure comparison on the benchmark dataset demonstrates that the RNN model outperforms its two lightweight variants, LSTM and GRU, in terms of alignment quality. This indicates that RNN is more effective at capturing medium-range semantic dependencies, thereby enhancing matching performance. The fifth column presents the runtime comparison, where the RNN model exhibits significantly lower processing time than LSTM and GRU, which corroborates the above analysis on time complexity, further confirming its computational efficiency during the matching process. A closer examination of the runtime of the three hybrid methods reveals that the additional time cost introduced by the N-gram module is approximately 3 to 4 s across all variants, suggesting that the primary runtime differences are attributed to the complexity of the underlying neural models. This further supports the conclusion that RNN can efficiently model semantic features with lower computational overhead. The final column reports the F-measure per second metric, which jointly reflects the effectiveness

and efficiency of each method. It can be observed that the proposed method achieves a 39.29% improvement over the second-best hybrid variant (i.e., LSTM + N-gram), indicating that it not only maintains high alignment quality but also offers superior computational efficiency, making it highly practical and scalable for ontology matching tasks.

6. Conclusions

To address ontology heterogeneity problems in 6G-enabled Cyber Twin-driven IoE systems, this paper proposes a hybrid ontology matching method that integrates deep learning and syntax-based similarity. The method combines RNN's sequence modeling capabilities with N-gram's sensitivity to local character patterns, enabling both semantic feature extraction and fine-grained alignment. Experimental results demonstrate that the proposed method significantly outperforms the compared methods, improving the F-measure by 18.18% over the representative methods from the OAEI and Transformer-based method, and by 18.46% over RNN-based hybrid methods with SimRank and Wu-Palmer. In the comparison of time and space complexity with lightweight RNN variants, the proposed method achieves a 39.29% improvement in F-measure per second over the second-best hybrid variant. These results highlight the effectiveness and efficiency of combining deep learning techniques and traditional similarity measurement methods and provide a promising direction for enhancing semantic interoperability in heterogeneous IoE environments.

Although the proposed method has achieved effective matching results on the heterogeneous benchmark dataset with diverse types of heterogeneity, which suggests its potential applicability and transferability to domain ontologies with similar heterogeneous characteristics, its practical validation on domain-specific ontologies remains to be further explored, particularly in scenarios involving dense specialized terminology or complex semantic relationships. In the future, generalization is planned to be realized through fine-tuning of domain embeddings or the incorporation of auxiliary knowledge bases. Furthermore, the method exhibits certain performance bottlenecks in handling scenarios where target entity labels are scrambled or comments are missing, leading to insufficient semantic information. This is primarily due to the model's reliance on meaningful annotation properties during the matching process, which underscores a dependency on high-quality input data and limits its applicability to ontologies with incomplete attributes. In the future, to address this issue, we intend to focus on target entities and their contexts by employing graph neural networks supplemented with adaptive ensemble weights. This approach aims to maximally mine effective semantic features and achieve fused embeddings, thereby constructing feature representations with enhanced semantic expressiveness.

Author Contributions: J.L.: Conceptualization, Methodology, Software, Validation, Formal Analysis, Investigation, Resources, Data Curation, Writing—original draft, Visualization; C.Y.: Writing—review and editing, Supervision, Project administration, Funding acquisition; All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the National Natural Science Foundation of China, grant number 51165018, and by the “Innovation Star” Project of Gansu Province's Outstanding Graduate Students, grant number 2022CXZX-407.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The data that support the findings of this study are available upon reasonable request to the author, Jiawei Lu.

Acknowledgments: The authors acknowledge the valuable support of the editors and reviewers.

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

References

1. Singh, S.P.; Kumar, N.; Singh, A.; Singh, K.K.; Askar, S.S.; Abouhawwash, M. Energy efficient hybrid evolutionary algorithm for Internet of Everything (IoE)-enabled 6G. *IEEE Access* **2024**, *12*, 63839–63852. [\[CrossRef\]](#)
2. Nivetha, A.; Preetha, K.S. A review on cyber-twin in sixth generation wireless networks: Architecture, research challenges & issues. *Wirel. Pers. Commun.* **2024**, *138*, 1815–1865. [\[CrossRef\]](#)
3. Pliatsios, A.; Kotis, K.; Goumopoulos, C. A systematic review on semantic interoperability in the IoE-enabled smart cities. *Internet Things* **2023**, *22*, 100754. [\[CrossRef\]](#)
4. Patel, A.; Debnath, N.C. A comprehensive overview of ontology: Fundamental and research directions. *Curr. Mater. Sci.* **2024**, *17*, 2–20. [\[CrossRef\]](#)
5. Xue, X. Complex ontology alignment for autonomous systems via the compact co-evolutionary brain storm optimization algorithm. *ISA Trans.* **2023**, *132*, 190–198. [\[CrossRef\]](#) [\[PubMed\]](#)
6. Geng, A.; Lv, Q. A multi-objective particle swarm optimization with density and distribution-based competitive mechanism for sensor ontology meta-matching. *Complex Intell. Syst.* **2023**, *9*, 435–462. [\[CrossRef\]](#)
7. Hao, Z.; Mayer, W.; Xia, J.; Li, G.; Qin, L.; Feng, Z. Ontology alignment with semantic and structural embeddings. *J. Web Semant.* **2023**, *78*, 100798. [\[CrossRef\]](#)
8. Wu, J.; Lv, J.; Guo, H.; Ma, S. Daeom: A deep attentional embedding approach for biomedical ontology matching. *Appl. Sci.* **2020**, *10*, 7909. [\[CrossRef\]](#)
9. Xue, X.; Jiang, C.; Zhang, J.; Zhu, H.; Yang, C. Matching sensor ontologies through siamese neural networks without using reference alignment. *PeerJ Comput. Sci.* **2021**, *7*, e602. [\[CrossRef\]](#)
10. Jiang, C.; Xue, X. Matching biomedical ontologies with long short-term memory networks. In Proceedings of the 2020 IEEE International Conference on Bioinformatics and Biomedicine (BIBM), Virtual Event, 16–19 December 2020; pp. 2484–2489.
11. Iyer, V.; Agarwal, A.; Kumar, H. VeeAlign: A supervised deep learning approach to ontology alignment. In Proceedings of the Fifteenth International Workshop on Ontology Matching, Online, 2 November 2020; pp. 216–224.
12. Wang, P.; Hu, Y. Matching biomedical ontologies via a hybrid graph attention network. *Front. Genet.* **2022**, *13*, 893409. [\[CrossRef\]](#)
13. Khoudja, M.A.; Fareh, M.; Bouarfa, H. Deep embedding learning with auto-encoder for large-scale ontology matching. *Int. J. Semant. Web Inf. Syst.* **2022**, *18*, 1–18. [\[CrossRef\]](#)
14. Bento, A.; Zouaq, A.; Gagnon, M. Ontology matching using convolutional neural networks. In Proceedings of the Twelfth Language Resources and Evaluation Conference, Marseille, France, 11–16 May 2020; pp. 5648–5653.
15. Hao, J.; Lei, C.; Efthymiou, V.; Quamar, A.; Özcan, F.; Sun, Y.; Wang, W. Medto: Medical data to ontology matching using hybrid graph neural networks. In Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, Virtual, 14–18 August 2021; pp. 2946–2954.
16. Chen, J.; Jiménez-Ruiz, E.; Horrocks, I.; Antonyrajah, D.; Hadian, A.; Lee, J. Augmenting ontology alignment by semantic embedding and distant supervision. In Proceedings of the 18th International Conference on The Semantic Web (ESWC 2021), Virtual Event, 6–10 June 2021; pp. 392–408.
17. Xue, X.; Jiang, C.; Zhang, J.; Hu, C. Biomedical ontology matching through attention-based bidirectional long short-term memory network. *J. Database Manag.* **2021**, *32*, 14–27. [\[CrossRef\]](#)
18. Xue, X.; Jiang, C.; Zhu, H. Matching ontologies through siamese neural network. In Proceedings of the International Conference on Mobile Multimedia Communications, Guiyang, China, 23–25 July 2021; pp. 715–724.
19. Xue, X.; Guo, J. Word embedding based heterogeneous entity matching on Web of Things. In Proceedings of the Companion of the Web Conference 2022, Lyon, France, 25–29 April 2022; pp. 941–947.
20. Wang, P.; Zou, S.; Liu, J.; Ke, W. Matching biomedical ontologies with GCN-based feature propagation. *Math. Biosci. Eng.* **2022**, *19*, 8479–8504. [\[CrossRef\]](#) [\[PubMed\]](#)
21. Chakraborty, J.; Bansal, S.K.; Virgili, L.; Konar, K.; Yaman, B. ONTOCONNECT: Unsupervised ontology alignment with recursive neural network. In Proceedings of the 36th ACM/SIGAPP Symposium on Applied Computing, Virtual Event, 22–26 March 2021; pp. 1874–1882.
22. He, Y.; Chen, J.; Dong, H.; Horrocks, I. Exploring large language models for ontology alignment. In Proceedings of the ISWC 2023 Posters, Demos and Industry Tracks, Athens, Greece, 6–10 November 2023; pp. 1–5.
23. Giglou, H.B.; D’Souza, J.; Engel, F.; Auer, S. LLMs4OM: Matching Ontologies with Large Language Models. In Proceedings of the Semantic Web: ESWC 2024 Satellite Events, Hersanissos, Greece, 26–30 May 2024; pp. 25–35.
24. Hertling, S.; Paulheim, H. Olala: Ontology matching with large language models. In Proceedings of the 12th Knowledge Capture Conference, Pensacola, FL, USA, 5–7 December 2023; pp. 131–139.
25. Chen, M.; Xu, Y.; Wu, N.; Pan, Y. Ontology matching method based on gated graph attention model. *Comput. Mater. Contin.* **2025**, *82*, 1–18. [\[CrossRef\]](#)

26. He, Y.; Chen, J.; Antonyrajah, D.; Horrocks, I. BERTMap: A BERT-based ontology alignment system. In Proceedings of the Thirty-Sixth AAAI Conference on Artificial Intelligence, Virtual Event, 22 February–1 March 2022; pp. 5684–5691.
27. Lama, V.; Patel, A.; Debnath, N.C.; Jain, S. IRI_Debug: An ontology evaluation tool. *New Gener. Comput.* **2024**, *42*, 177–197. [\[CrossRef\]](#)
28. Portisch, J.; Hladik, M.; Paulheim, H. Background knowledge in ontology matching: A survey. *Semant. Web* **2024**, *15*, 2639–2693. [\[CrossRef\]](#)
29. Gulić, M.; Vrdoljak, B.; Ptiček, M. CroMatcher 2.0: A comprehensive analysis of the improved ontology matching system. *IEEE Access* **2025**, *13*, 27610–27640. [\[CrossRef\]](#)
30. Mienye, I.D.; Swart, T.G.; Obaido, G. Recurrent neural networks: A comprehensive review of architectures, variants, and applications. *Information* **2024**, *15*, 517. [\[CrossRef\]](#)
31. Lin, T.; Wang, Y.; Liu, X.; Qiu, X. A survey of transformers. *AI Open* **2022**, *3*, 111–132. [\[CrossRef\]](#)
32. Ma, K.; Tian, M.; Tan, Y.; Qiu, Q.; Xie, Z.; Huang, R. Ontology-based BERT model for automated information extraction from geological hazard reports. *J. Earth Sci.* **2023**, *34*, 1390–1405. [\[CrossRef\]](#)
33. Zhong, B.; Pan, X.; Love, P.E.D.; Ding, L.; Fang, W. Deep learning and network analysis: Classifying and visualizing accident narratives in construction. *Autom. Constr.* **2020**, *113*, 103089. [\[CrossRef\]](#)
34. Wang, J.; Huang, H.; Wu, Y.; Zhang, F.; Zhang, S.; Guo, K. Open knowledge graph link prediction with semantic-aware embedding. *Expert Syst. Appl.* **2024**, *249*, 123542. [\[CrossRef\]](#)
35. Hancock, J.T.; Khoshgoftaar, T.M. Survey on categorical data for neural networks. *J. Big Data* **2020**, *7*, 1–41. [\[CrossRef\]](#)
36. Graff, M.; Moctezuma, D.; Téllez, E.S. Bag-of-word approach is not dead: A performance analysis on a myriad of text classification challenges. *Nat. Lang. Process. J.* **2025**, *11*, 100154. [\[CrossRef\]](#)
37. Faria, D.; Lima, B.; Silva, M.C.; Couto, F.M.; Pesquita, C. AML and AMLC results for OAEI 2021. In Proceedings of the 16th International Workshop on Ontology Matching co-located with the 20th International Semantic Web Conference (ISWC 2021), Online, 25 October 2021; pp. 131–136.
38. Jiménez-Ruiz, E. LogMap family participation in the OAEI 2020. In Proceedings of the 15th International Workshop on Ontology Matching Co-Located with the 19th International Semantic Web Conference (ISWC 2020), Online, 2 November 2020; pp. 201–203.
39. Jiang, C.; Xue, X. A uniform compact genetic algorithm for matching bibliographic ontologies. *Appl. Intell.* **2021**, *51*, 7517–7532. [\[CrossRef\]](#)
40. Wang, Y.; Xu, R.; Feng, Z.; Che, Y.; Chen, L.; Luo, Q.; Mao, R. Disk: A distributed framework for single-source SimRank with accuracy guarantee. *Proc. VLDB Endow.* **2020**, *14*, 351–363. [\[CrossRef\]](#)
41. Panigutti, C.; Perotti, A.; Pedreschi, D. Doctor XAI: An ontology-based approach to black-box sequential data classification explanations. In Proceedings of the 2020 Conference on Fairness, Accountability, and Transparency, Barcelona, Spain, 27–30 January 2020; pp. 629–639.
42. Wen, X.; Li, W. Time series prediction based on LSTM-attention-LSTM model. *IEEE Access* **2023**, *11*, 48322–48331. [\[CrossRef\]](#)
43. Zhou, Y.; He, X.; Montillet, J.-P.; Wang, S.; Hu, S.; Sun, X.; Huang, J.; Ma, X. An improved ICEEMDAN-MPA-GRU model for GNSS height time series prediction with weighted quality evaluation index. *GPS Solut.* **2025**, *29*, 113. [\[CrossRef\]](#)
44. Das, H.; Das, S.; Gourisaria, M.K.; Khan, S.B.; Almusharraf, A.; Alharbi, A.I.; Mahesh, T.R. Enhancing software fault prediction through feature selection with spider wasp optimization algorithm. *IEEE Access* **2024**, *12*, 105309–105325. [\[CrossRef\]](#)
45. Zamboni, P.; Junior, J.M.; Silva, J.A.; Miyoshi, G.T.; Matsubara, E.T.; Nogueira, K.; Gonçalves, W.N. Benchmarking anchor-based and anchor-free state-of-the-art deep learning methods for individual tree detection in RGB high-resolution images. *Remote Sens.* **2021**, *13*, 2482. [\[CrossRef\]](#)

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