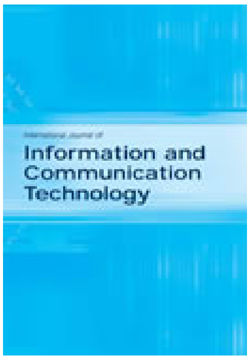


International Journal of
Information and Communication Technology

International Journal of Information and Communication Technology

ISSN online: 1741-8070 - ISSN print: 1466-6642
https://www.inderscience.com/ijict

Construction of digital art knowledge graph based on deep recurrent neural network

Huan Wang

DOI: 10.1504/IJICT.2026.10076063

Article History:

Received:	02 September 2025
Last revised:	18 November 2025
Accepted:	22 November 2025
Published online:	16 February 2026

Construction of digital art knowledge graph based on deep recurrent neural network

Huan Wang

College of Art and Design,
Shanghai Normal University Tianhua College,
Shanghai, 201815, China
Email: wh2063@sthu.edu.cn

Abstract: This study presents a method for constructing digital art knowledge graphs based on deep recurrent neural network (DRNN). A digital art knowledge graph is initially constructed by extracting visual features with ResNet50 and identifying textual entities via a CNN-BiLSTM-CRF model. Then, a DRDA model with bidirectional gated recurrent unit (GRU) and neighbour-aware attention is proposed for graph completion. Experiments on DBPedia50k and DBPedia500k show DRDA's superiority over three baselines. On DBPedia50k, DRDA improves head prediction MRR by up to 55% and achieves the lowest MR in tail prediction, though trailing slightly in Hits@10. On DBPedia500k, DRDA consistently outperforms baselines with MR reductions of 59–406 and MRR gains of 2%–19%. Further analysis identifies optimal depth and neighbour parameters, validating the model's scalability and its effectiveness in capturing complex semantic dependencies in large-scale multimodal art data.

Keywords: digital art; knowledge graph; deep recurrent neural network; DRNN.

Reference to this paper should be made as follows: Wang, H. (2026) 'Construction of digital art knowledge graph based on deep recurrent neural network', *Int. J. Information and Communication Technology*, Vol. 27, No. 11, pp.18–37.

Biographical notes: Huan Wang obtained her Master's degree from Shanghai Normal University in 2007. She is currently an Associate Professor at the College of Art and Design, Tianhua College, Shanghai Normal University. Her research interests encompass traditional painting, digital art, and artificial intelligence.

1 Introduction

Knowledge graphs are a promising tool for exploring diverse, dynamic, and large-scale datasets. They combine deductive and inductive techniques (Hogan et al., 2021) and have become a fundamental tool in the field of artificial intelligence for modelling structured information, capturing semantic relationships, and enabling cross-domain intelligent reasoning (Peng et al., 2023). By representing entities and relationships as triples in a graph structure, knowledge graphs facilitate question-answering systems, tailored recommendations, and semantic search, with broad applications in technology (Wang et al., 2023), education (Abu-Salih and Alotaibi, 2024), art (Castellano et al., 2022). About the art, knowledge graphs have demonstrated significant value in areas such as art management, automated analysis, and cultural dissemination (Huang et al., 2023).

However, with the continuous development of digital humanities and digital art, traditional knowledge graph construction methods are no longer able to adapt to the numerous characteristics of modern digital art. Specifically, conventional approaches to knowledge graph construction primarily depend on human curation, which requires domain-specific expertise and often demands significant human resources in many complex construction scenarios (Yu et al., 2024). Although artificial intelligence progress has made machine learning applicable to knowledge graph building, and initial success has been achieved in areas such as entity learning and ontology learning (Zhao et al., 2024), thereby addressing the time-consuming and labour-intensive issues of manual construction, these methods typically involve pre-processing and analysing text descriptions using specific algorithms (Chen et al., 2020; Zhong et al., 2023), they still face issues such as error propagation and a lack of guiding information during model training. Additionally, existing automatic extraction techniques often fail to adequately consider the associative information between entities and relationships, leading to incomplete or inaccurate knowledge representations. The static nature of traditional knowledge graph construction methods also cannot capture the dynamic evolution of art movements or the continuously expanding digital art ecosystem, resulting in knowledge graphs becoming quickly outdated or containing significant coverage gaps. Furthermore, digital artworks inherently possess composite multimodal attributes (Chun, 2011), and this multimodal nature requires knowledge graph construction to integrate information from diverse data sources while preserving the complex interdependencies between different modalities. Digital artworks present unique challenges for knowledge graph construction, necessitating deeper integration across different modalities.

In this context, deep recurrent neural network (DRNN) demonstrates significant advantages. Data in the field of digital art is highly heterogeneous and temporal, containing not only visual information such as images but also temporal semantic features such as descriptive text, creative background, and stylistic evolution. Conventional approaches to building knowledge graphs typically depend on structured or static text, making it difficult to effectively integrate and model these multimodal, dynamically interconnected data. DRNN with its capable of capturing temporal dependencies in input data through its recurrent structure. Compared to traditional neural networks, DRNN possesses stronger contextual modelling capabilities (Guo et al., 2019), widely applied in natural language processing and speech recognition tasks, can capture contextual dependencies and semantic progression in art work descriptions, particularly suited for handling features that evolve over time, such as style and theme. This effectively enhances the accuracy and generalisation capabilities of tasks like art work attribute prediction and relationship inference (Li et al., 2022; Ye et al., 2022). Additionally, by incorporating gating mechanisms, such as gated recurrent unit (GRU), DRNN can effectively remember long-range dependencies, thereby strengthening the connection between entities and relationships in the graph. Furthermore, DRNN structures combined with attention mechanisms can dynamically focus on key information, achieving deep associations between image features and text entities in multimodal fusion. In summary, DRNN can improve the construction accuracy of digital art knowledge graphs and is an ideal choice for addressing complex multi-modal data knowledge graph construction problems in the field of digital art.

Building upon these foundations, the present study introduces an RNN-based digital art knowledge graph construction method to enhance the recognition accuracy and processing capability of multimodal artistic data. First, ResNet50 is employed for visual feature extraction from digital art images to achieve efficient image classification and semantic understanding. Second, a CNN-BiLSTM-CRF model is used to ensure extraction precision for artistic descriptions. Additionally, an attention-enhanced bidirectional GRU encoding mechanism is introduced to processes long-span dependencies in multimodal sequential inputs, and a dynamic knowledge graph completion approach is designed to infer absent entities and relationships.

The main innovations and contributions of this work include:

1. Multimodal feature integration: to overcome the constraints of conventional knowledge graph construction approaches in processing multimodal digital art data, this paper employs ResNet50 for image feature extraction and CNN-BiLSTM-CRF (Chiu and Nichols, 2016) for textual entity recognition. The integration of visual and textual features through similarity-based alignment effectively improves the comprehensiveness of knowledge representation, resulting in substantial enhancement in multimodal data processing capability. This improvement is particularly significant in digital art environments where visual and textual information need to be jointly analysed.

2. Deep RNN-based sequence modelling: to cope with the challenges of complex semantic relationships and long-term dependencies in artistic descriptions, this paper employs deep bidirectional GRU networks with attention mechanisms. The attention-enhanced RNN architecture effectively captures contextual semantic information and reduces interference from irrelevant information, which enhances the precision of entity and relation extraction. This improvement significantly enhances the understanding capability of the system for complex artistic concepts and cultural contexts.

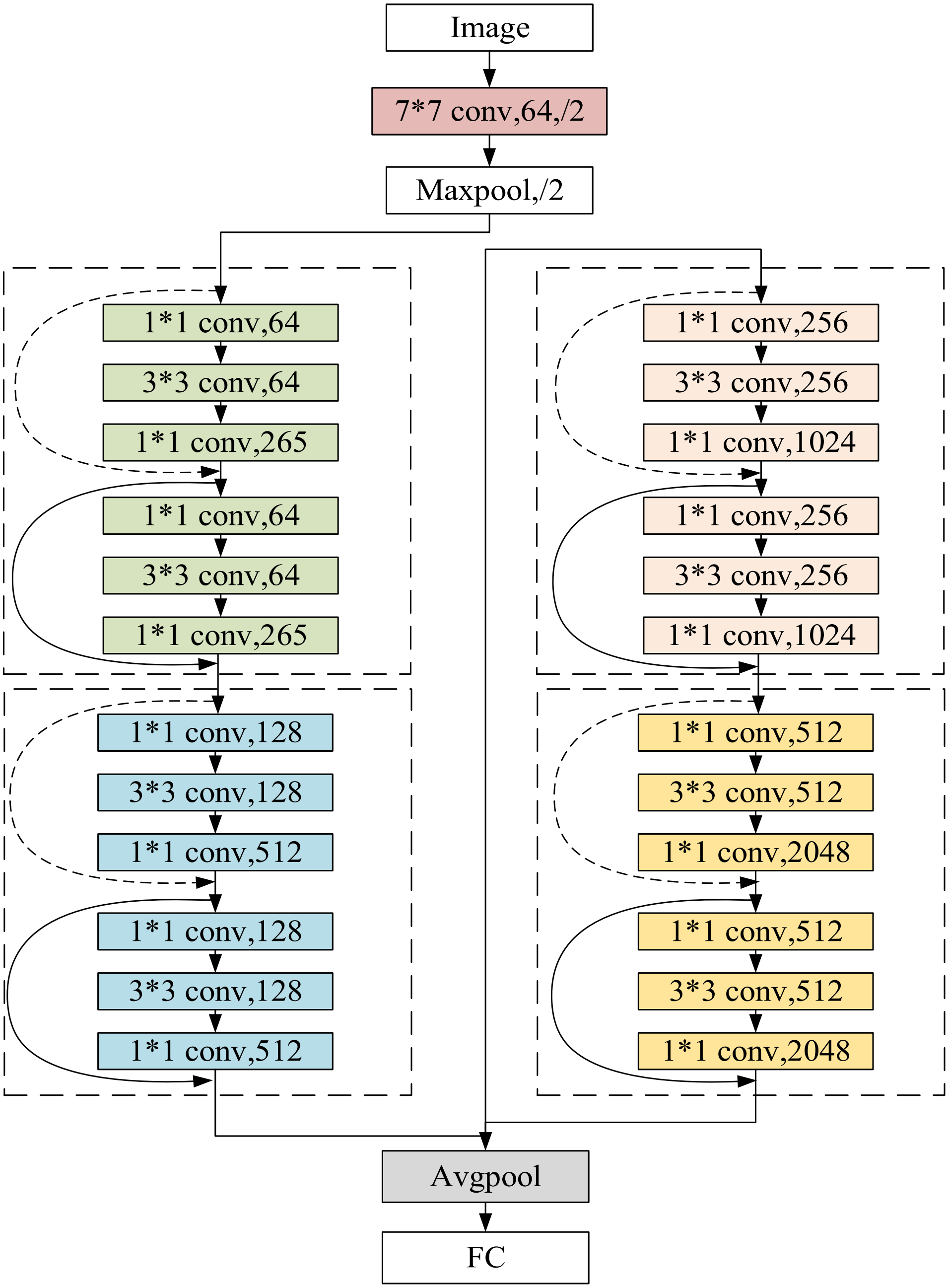
3. Dynamic knowledge completion mechanism: this paper introduces an RNN-driven dynamic completion approach based on neighbour information and translation principles to predict missing entities and relationships in the knowledge graph. The N-attention module adaptively weights entity and relation information according to structural patterns, which improves the flexibility and effectiveness of knowledge graph completion. Meanwhile, the scoring function combining similarity and translation models further reduces computational complexity and improves completion accuracy.

2 Digital art knowledge graph initial construction

2.1 Image feature extraction based on ResNet50.

The image information in the digital art knowledge graph constructed in this research is sourced from the ArtDL electronic art dataset. This paper employs the ResNet50 model for image classification. The ResNet50 model incorporates residual blocks, enabling information to flow directly from the input layer to the output layer, thereby avoiding the issue of gradient vanishing. In the ResNet50 model, each residual block contains multiple convolutional layers, including shortcut connections, enabling the network to be deeper while maintaining gradient stability. As a result, the ResNet50 model performs exceptionally well in image classification and is widely adopted.

Figure 1 ResNet50 model architecture (see online version for colours)



We first use methods such as scaling, cropping and filling, and adjusting image proportions to resize and normalise the original images, and then perform image classification based on the ResNet50 model. The residual structure of ResNet50 helps preserve visual features across layers and alleviates vanishing gradients, making it suitable for capturing fine-grained details in digital art images. The model structure diagram is shown in Figure 1. This model consists of multiple residual blocks, each containing multiple convolutional layers and batch normalisation (BN) layers. The model is primarily divided into an input layer, convolutional layers and pooling layers, residual blocks, a global average pooling layer, a fully connected layer, and an output layer.

The model is primarily divided into an input layer, convolution layers and pooling layers, residual blocks, global average pooling layers, fully connected layers, and an output layer:

1 Input layer: image data after pre-processing

2 Convolution layers and pooling layers: used for feature extraction and down sampling of image data

3 Residual blocks: the ResNet50 model has a total of 4 residual blocks, containing 3, 4, 6, and 3 residual units respectively, with a total of 50 layers. Each residual block includes multiple convolutional layers and a cross-layer connection, which directly passes the input to the output, thereby addressing the issue of gradient vanishing. The Equation for the residual block are shown in equation (1) and equation (2)

4 Global average pooling layer: use a global average pooling layer to convert feature maps into vectors, which are then used as input for the classifier

5 Fully connected layer and output layer: convert feature vectors into classification scores, then map the scores to category probabilities via a softmax output layer. The equation for the ResNet50 model is shown in equation (3).

$$y = F\left(x_0, \{W_i\}\right) + x_0 \tag{1}$$

where $F(\cdot)$ denotes the mapping function of the residual block, and W_i denotes the parameters of that residual block. $F(x_0, \{W_i\})$ denotes the mapping function from input x_0 to output y, which can be expressed as:

$$F\left(x_0, \{W_i\}\right) = W_2 \sigma\left(W_1 x + W_3\right) \tag{2}$$

where W_1 and W_2 are the weight matrices of the convolution layer, and W_3 is the bias vector.

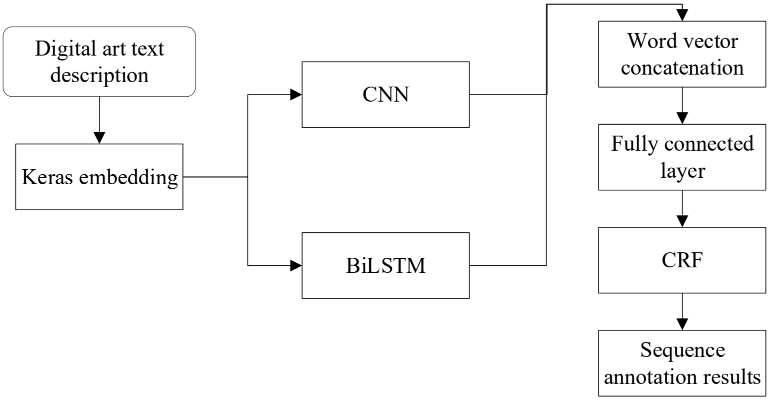
$$y = \text{Softmax}\left(W_5 g\left(W_4 f(W_3 e\left(W_2 d\left(W_1 c\left(x_0\right)\right)\right)\right)\right) \tag{3}$$

The model uses ReLU as the activation function to represent non-linear transformation. Where x_0 represents the input image data, $c(\cdot)$ represents the first convolutional layer, $d(\cdot)$ represents the second max pooling layer, $e(\cdot)$ represents the residual blocks from the third to the sixteenth layers, $f(\cdot)$ represents the global average pooling layers from the thirteenth to the sixteenth layers, and $g(\cdot)$ represents the fully connected layer. W_i represents the weight parameters for each layer.

2.2 Text information extraction based on CNN-BiLSTM-CRF

The textual description information of entities in the digital art knowledge graph constructed by this research also comes from the ArtDL electronic art dataset, using the descriptions of these works as textual knowledge for the digital art knowledge graph. This paper uses a CNN-BiLSTM-CRF model, thoroughly mines these unstructured textual data. The choice of CNN-BiLSTM-CRF architecture is motivated by the complementary strengths of its components. The CNN layer is effective in extracting local patterns such as syntactic or stylistic phrases commonly found in art descriptions. The BiLSTM captures bidirectional contextual dependencies, which is essential for understanding sequential and thematic coherence in descriptive narratives. The CRF layer ensures optimal global label consistency in the final prediction, improving the accuracy of named entity recognition. This combination is particularly suitable for digital art texts, where entity boundaries and artistic context often require joint modelling of local features and long-range dependencies.

Figure 2 CNN-BILSTM-CRF model architecture (see online version for colours)


Digital art text description
Keras embedding
CNN
BiLSTM
Word vector concatenation
Fully connected layer
CRF
Sequence annotation results

It is worth noting that some artistic terms in the dataset may present ambiguity or polysemy (e.g., ‘impression’ could refer to an art style or a specific artwork title). While the CNN-BiLSTM-CRF model captures contextual features to improve entity recognition accuracy, explicit disambiguation mechanisms (such as entity linking to external ontologies or context-based clustering) are not integrated in the current version. This remains a potential direction for future work, particularly in domains like digital art where symbolic overlaps are common.

2.2.1 BiLSTM model

The long short-term memory (LSTM) (Hochreiter and Schmidhuber, 1997) comprises several key components: input at time t, cell state, candidate cell state, hidden state, and three types of gates – forget, input (memory), and output. The model operates by selectively retaining or discarding information through these gates, enabling it to preserve relevant features across time steps. The hidden state undergoes modification and is output at every step, with the activation of gates being a function of both the immediately preceding hidden state and the current input.

Bi-directional long short-term memory (BiLSTM) extends this architecture by incorporating two parallel LSTMs: one layer processes the sequence in the forward direction, while the other handles it backward, enabling the model to incorporate both historical and future context, producing richer representations from both directions:

$$h_R = \{h_{R1}, h_{R2}, h_{R3}, \ldots, h_{Rn}\} \tag{4}$$

Then, process the sequence backwards to the LSTM to obtain the output vector:

$$h_L = \{h_{L1}, h_{L2}, h_{L3}, \ldots, h_{Ln}\} \tag{5}$$

Next, concatenate the forward and backward hidden vectors:

$$h_t = [h_{Rt} \oplus h_{Lt}] \tag{6}$$

where h_t represents the hidden layer vector of BiLSTM at time t, and finally obtains the final BiLSTM output result $h = \{h_1, h_2, h_3, \ldots, h_n\}$.

2.2.2 Model structure of CNN-BiLSTM-CRF

The CNN-BiLSTM-CRF model consists of four main components: a Keras embedding layer, a CNN layer, a BiLSTM layer, and a CRF layer. The model takes as input text descriptions related to digital art, which are aggregated and annotated to build a comprehensive dataset. The embedding layer transforms the input text into dense vector representations. Next, the CNN layer captures local contextual features, while the BiLSTM layer extracts long-range dependencies and global semantic patterns, leveraging its bidirectional structure to consider both past and future context. The outputs of the character-level and word-level representations are concatenated and passed through a fully connected layer, followed by the CRF layer for structured sequence labelling. The model outputs predicted entity labels after decoding. Finally, post-processing steps, including entity prediction and deduplication, are applied to construct the cleaned and finalised dataset.

2.3 Similarity-based data alignment

Based on the text and image modal data obtained in the preceding section, this paper employs a label alignment algorithm to preliminarily construct a digital art knowledge graph:

1 Multi-modal data label extraction: input the obtained multi-modal dataset, and perform semantic label extraction on the text and image data separately. The text data is annotated using a CNN-BiLSTM-CRF model, with the semantic labels stored in a table; the image data is extracted using ResNet50 to form an image semantic label set

2 Construction of data label sets: the extracted labels from each modality are aggregated and stored separately as text and image label sets

3 Similarity calculation: the longest common subsequence (LCS) text similarity algorithm is used to find the LCS between labels of different modalities and calculate the similarity of the label group

4 Similarity sorting: first, sort the calculated similarity scores.

Then, select the group with the highest similarity between text and image labels for matching entities in the entity set. To associate these labels with entities, link the images and text in the label group to their corresponding entities. Finally, output the corresponding data pairs.

During multimodal fusion, inconsistencies may arise when semantic labels extracted from images and text do not align. To address this, we incorporate a similarity threshold in the alignment step to filter out weak matches. Additionally, attention mechanisms in the downstream model dynamically weigh modality-specific features, reducing the influence of noisy or conflicting information. These strategies help mitigate information loss and enhance the robustness of the constructed knowledge graph in the presence of imperfect modality alignment.

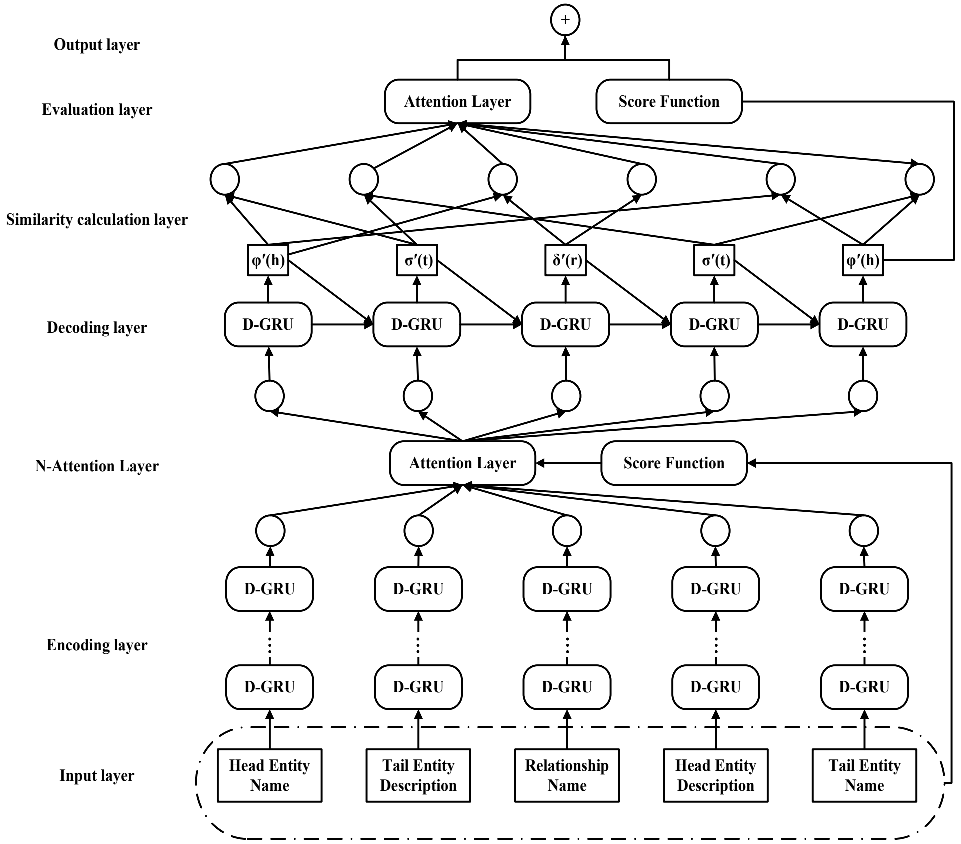
2.4 Data integration and storage

Neo4j graph database offers a flexible approach to data modelling through its graph-based structure. In contrast to conventional relational databases, it does not require significant time and effort to redefine the data structure and type of tables, and it allows for the creation of entirely new tables. The final statistics for digital art data include the annotation of 57 types of entities, 26 types of relationships, and 16 types of attributes, with a total of 6,378 annotated nodes and over 19,000 entity relationships. After extracting the digital art entities and relationships, they were organised, deduplicated, converted to CSV format, and uploaded to the Neo4j graph database in UTF-8 encoding format, facilitating subsequent research on digital art knowledge graphs, including inference and retrieval.

3 Deep RNN-based knowledge graph completion

Based on the above, we propose a knowledge graph completion model based on deep RNN coding and double attention mechanism (DRDA), as shown in the Figure 3: It embeds the semantic feature information extracted from entity description text into the structured information of the original knowledge graph, thereby fully learning the internal association information of each triplet. However, ConMask does not consider the associative learning between external text feature information and internal structural information. Therefore, the DRDA model considers reconstructing the structural information of the original knowledge graph based on the semantic association information between entities and relationships, and treats each reconstructed tuple (head entity, association information between the head entity and the relationship, relationship, association information between the tail entity and the relationship, tail entity) as a short sequence representation. Then, a deep bidirectional gate recurrent unit (GRU) encoding network with different RNN units is used to learn the dependency information between external text information and internal structural information. Subsequently, an N-Attention layer is employed to enhance the weight information between entities and relationships as well as between two entities. Finally, the decoding layer of the GRU is used to decode the encoded short sequence information, restoring it to a tuple representation, and a comprehensive score is calculated using similarity and translation principles (Schuster and Paliwal, 1997).

Figure 3 Knowledge graph completion model based on deep RNN



3.1 Deep Bi-GRU encoding layer

In NLP encoding tasks, the Bi-GRU layer is effective in capturing the contextual semantics of each word within a sequence. Initially, the word vector layer transforms every word in the input into a dense vector representation. Thus, an input sequence can be represented as $W = \{w_1, \ldots, w_t, w_{i+1}, \ldots, w_n\}$, where $w_t \in \mathbb{R}^d$ denotes the d-dimensional embedding of the ttt^{th} word, and n is the total length of the sequence. After word embedding, two GRU layers operate in parallel: a forward GRU and a backward GRU. For each word w_t, it is encoded by the forward GRU layer based on the context information from w_1 to w_n, denoted as $\vec{h}_t$. Simultaneously, each word w_t is also encoded by the backward GRU layer based on the context information from w_n to w_n, denoted as $\vec{h}_t$. The detailed computational process is as follows (Schuster and Paliwal, 1997):

$$z_t = \sigma\left(W_{xz}^T x_t + W_{hz}^T h_{t-1} + b_z\right) \tag{7}$$

$$r_t = \sigma\left(W_{xr}^T x_t + W_{hr}^T h_{t-1} + b_r\right) \tag{8}$$

$$g_t = \tanh\left(W_{xg}{}^T x_t + W_{hg}{}^T \left(r_t \otimes h_{t-1}\right) + b_g\right) \tag{9}$$

$$y_t = h_t = \left(1 - z_t\right) \otimes \tanh\left(W_{xg}{}^T h_{t-1} + z_t \otimes g_t\right) \tag{10}$$

where z and r represent the update gate and reset gate, respectively. The update gate controls the extent to which previous information is retained and propagated forward, while the reset gate determines which portions of historical information should be excluded or forgotten. b_α denotes the bias value corresponding to each gate unit, ⊗ denotes the cross product, *tanh* is the activation function, y_t denotes the semantic information transmitted between multi-layer Bi-GRUs, and $W_{(A,\ B)}$ denotes the weight parameters between A and B. For each word w_t, the forward GRU layer considers the text information from w_1 to w_t during the encoding process, resulting in vector $\overrightarrow{h_t}$. The backward GRU layer considers the text information from w_t to w_t, resulting in vector $\overleftarrow{h_t}$. Finally, connecting these two vectors yields $h_t = \left[\overleftarrow{h_t}, \overrightarrow{h_t}\right]$.

3.2 N-attention module

The N-attention layer is composed of two main components: an attention layer implementing the attention mechanism, and a neighbours layer responsible for incorporating neighbourhood information. For clarity in the subsequent explanation, the encoded representation of the short sequence is denoted as $[h, h_r, r, t_r, t]$, Let n_h and n_t represent the number of direct neighbours for the head and tail entities, respectively, and let N denote the number of relation neighbours. Additional parameters include the entity neighbour threshold ϑ, the relation neighbour threshold is π, the entity neighbour parameter is δ, RN is the set of relation neighbour tuples $\{(h_{n1}, t_{n1}), (h_{n2}, t_{n2}), \ldots, (h_{ni}, t_{ni})\}$, and the relation neighbour parameter is μ.

The neighbours layer primarily utilises neighbour structure information to add additional weight information to each position in short sequence information.

1 For the position weights of the head and tail entities (h, t), if the number of direct neighbours of the head and tail entities is greater than ϑ or there are entity neighbours with the same path as the current relationship r, then the head and tail entities are considered to have high confidence and should be given sufficient attention. If the opposite is true, then they are not processed.

2 For the position weights of (h_r, t_r) in the short sequence, if the direct neighbours of the head and tail entities also appear in their respective descriptive text information, then a higher weight should be assigned. If not, then no processing is performed.

3 For relation r, if the number of neighbours of the relation (the number of triples containing the relation) is greater than τ, then the weight information for the current relation is increased. If not, no processing is performed.

Additionally, the translation characteristic ($h + r \approx t$) is considered as a scoring factor for relation neighbours. In summary, the neighbour information gain layer primarily maintains a weight matrix φ, which is a multidimensional matrix with the same dimension as the input short sequence information. That is:

$$\varphi = \left[\delta\left(1+\frac{n_{hr}}{n_h}\right), 1+\frac{n_{td}}{n_t}, R_g, 1+\frac{n_{hd}}{n_h}, \delta\left(1+\frac{n_{tr}}{n_t}\right) \right] \tag{11}$$

$$R_g = \frac{\mu}{N} \sum_{(h_n,t_n)\in R_N} \sigma\left(\frac{1}{|h_n + r - t_n|}\right) \tag{12}$$

$$\delta = \begin{cases} 1, & n_h, n_t \le \vartheta \\ \delta, & n_h, n_t > \vartheta \end{cases}, \quad \mu = \begin{cases} 1, & N \le \tau \\ \mu, & N > \tau \end{cases} \tag{13}$$

where φ is a dynamic hyperparameter matrix and also the core processing scheme for weight enhancement achieved by obtaining neighbour information. The hyperparameter matrix φ is set through specific experiments. Initially, the hyperparameter matrix φ is initialised to 1, i.e., $\varphi = [1, 1, 1, 1, 1]$, and then the value at each position of the matrix φ is calculated using neighbour information. According to equation (13), the values of δ and μ are determined by the number of neighbours, σ is the activation function, n_{hr} and n_{tr} are the number of neighbours in the head and tail entities that share the same relationship path as r, n_{td} is the number of times the neighbouring entities in the tail entity appear in the text description of the tail entity, and n_{hd} is the number of times the neighbouring entities in the tail entity appear in the text description of the tail entity. Therefore, based on the neighbour information of the entity relationship and the description information of the entity, the value of each parameter in the hyperparameter matrix φ can be calculated.

For the attention layer in the N-attention module, given the output of the encoding layer $H = [h_1, h_2, \ldots, h_n]$ ($h_i \in R^d$), where d is the number of hidden layer neurons in the LSTM, and n is the length of the input short sequence. The attention mechanism can be used to calculate attention probabilities, thereby highlighting the importance of each part of the sequence to the overall input sequence. The specific calculation process is as follows:

$$h_{Nt} = U_a \tanh\left(U_a h_N + U_c h_t + b_a\right) \tag{14}$$

$$a_{Nt} = \frac{\exp\left(h_{Nt}\right)}{\sum_{j=1}^{m} \exp\left(h_{Nj}\right)} \tag{15}$$

$$h_t' = \varphi \sum_{i=1}^{m} a_{Nt} h_t \tag{16}$$

where U_a, U_b and U_c represent the weight matrices of the attention mechanism, b_a represents the bias vector value of the attention mechanism, and h_t' represents the new feature output of the t^{th} element. In equation (16), the feature output is multiplied by the hyperparameter matrix φ and the feature output of the conventional attention layer to obtain the new feature output.

3.3 Bi-GRU decoding layer and scoring function

Use the GRU network to decode and generate a decoding sequence. When detecting the label of word w_t, the input to the decoding layer is: h'_{t+1} calculated through the attention mechanism, the previous label prediction vector T_{t-1}, and the hidden layer vector h_{t-1} from the previous decoding layer. The specific calculation process is as follows (Schuster and Paliwal, 1997):

$$z_t^{(2)} = \sigma\left(W_{wz}^{(2)} h'_{t-1} + W_{hz}^{(2)} h_{t-1}^{(2)} + W_{tz} T_{t-1} + b_z^{(2)}\right) \tag{17}$$

$$r_t^{(2)} = \sigma\left(W_{wr}^{(2)} h'_{t+1} + W_{hr}^{(2)} h_{t-1}^{(2)} + W_{tr} T_{t-1} + b_r^{(2)}\right) \tag{18}$$

$$g_t^{(2)} = \tanh\left(W_{wg}^T h'_{t-1} + W_{hg}{}^T (r_t \otimes h_{t-1}^{(2)}) + W_{tg}^T T_{t-1} + b_g^{(2)}\right) \tag{19}$$

$$h'_t = \sigma\left(W_{pw}{}^T h'_{t+1} + W_{pn}^T T_{t-1} + W_{pm}{}^T \left(r_t \otimes h_{t-1}^{(2)}\right) + b_p^{(2)}\right) \tag{20}$$

$$T_t = \tanh\left(W_{ts} h'_t + b_{ts}\right) \tag{21}$$

$$y_t = h_t^{(2)} = \left(1 - z_t\right) \otimes \tanh\left(W_{xg}^T h_{t-1} + W_{xk}^T T_{t-1} + z_t \otimes g_t\right) + T_t \tag{22}$$

Subsequently, at time step t, the output from the decoding layer is represented as ($\varphi'(h)$, $\sigma'(t)$, δ' (r), $\sigma'(h)$, $\varphi'(t)$, which correspond to the transformed sequences of (h, h_r, r, t_r, t) after decoding. These outputs are reconstructed into tuples for pairwise comparison, and their similarity is evaluated using the cosine similarity metric. This process yields a new set of sequence features $h'_t = \{h_1, h_2, \ldots, a_n\}$. where h_n denotes the mean vector computed by averaging the first n vectors. Taking the final similarity calculation output as an example, the vector w_n is fed into the model's encoding layer to obtain the corresponding feature representation h_n. The attention probability of the t^{th} element for the N^{th} element is a_{Nt}, calculated as follows:

$$h_{Nt} = U_a \tanh\left(U_a h_N + U_c h_t + b_a\right) \tag{23}$$

$$a_{Nt} = \frac{\exp\left(h_{Nt}\right)}{\sum_{j=1}^{m} \exp\left(h_{Nj}\right)} \tag{24}$$

where U_a, U_b, U_c represent the weight matrices of the attention mechanism, and b_a represents the bias vector value of the attention mechanism. Then, the average score of each element at time t can be obtained as follows:

$$s_t = \frac{1}{m} \sum_{i=1}^{m} a_{Nt} \tag{24}$$

In addition to calculating similarity scores, a new scoring function based on translation models will also be used:

$$f\left(h, h_r, r, t_r, t\right) = \left\|\left(h + h_r + r\right) - \left(t_r + t\right)\right\|_2^2 \tag{25}$$

Therefore, combining the two scoring mechanisms yields the final output layer result:

$$o_t = f_t\left(\varphi'(h), \sigma'(t), \delta'(r), \sigma'(h), \varphi'(t)\right) + s_t \tag{26}$$

where *f* is the scoring function based on the translation model, *st* is the scoring result based on similarity at time *t*, and O_t is the overall output score of the DRDA model. A bidirectional GRU network is used for association modelling of short sequence tuples, thereby learning internal dependencies and deep dependencies within the sequence. The reason for choosing GRU is primarily because it has a faster convergence rate than LSTM, making it highly effective for training deep RNNs.

To accelerate the training speed of the DRDA model, the following objective function is designed to achieve the experimental objectives. The optimiser uses the Adagrad algorithm proposed by Duchi et al. (2011), and the objective function is defined as:

$$\mathcal{L}(h,r,t) = \begin{cases} \sum\limits_{h_+ \in E^+} -\dfrac{\log\left(S\left(h_+, r, t, E^+ \cup E^-\right)\right)}{|E^+|}, & p_c > 0.5 \\ \sum\limits_{t_+ \in E^+} -\dfrac{\log\left(S\left(h, r, t_+, E^+ \cup E^-\right)\right)}{|E^+|}, & p_c \le 0.5 \end{cases} \tag{27}$$

where p_c denotes the blocking probability factor sampled from a uniform distribution *U*[0, 1]. If $p_c > 0.5$, the tail entity *t* of the input remains unchanged, while the positive and negative samples are generated by modifying the head entity. Conversely, if $p_c \le 0.5$, the head entity *h* of the input remains unchanged, and variations are introduced to the tail entity to form positive and negative samples. E^+ and E^- are the sets of positive and negative entities obtained from their respective target distributions P^+, P^- of positive and negative samples, respectively, and P^+, P^- are also obtained using a simple uniform distribution. When $p_c \le 0.5$, P^+ is the uniform distribution over the entity set $\{t+ \mid \langle h, r, t_+\rangle \notin \boldsymbol{T}\}$, and P^- is the uniform distribution of entities in $\{t- | \langle h, h, t-\rangle \notin \boldsymbol{T}\}$. When $p_c > 0.5$, P^+ is the uniform distribution of the entity in $\{h+ \mid \langle h+, r, t\rangle \notin \boldsymbol{T}\}$, and P^- is the uniform distribution of the entity in $\{h- \mid \langle h-, r, t\rangle \notin \boldsymbol{T}\}$. *S* represents the softmax-normalised output generated by DRDA:

$$S\left(h,r,t,E^{\pm}\right) = \begin{cases} \dfrac{\exp(\text{DKGC-DRDA}(h,r,t))}{\sum\limits_{e \in E^{\pm}} \exp(\text{DKGC-DRDA}(e,r,t))}, & p_c > 0.5 \\ \dfrac{\exp(\text{DKGC-DRDA}(h,r,t))}{\sum\limits_{e \in E^{\pm}} \exp(\text{DKGC-DRDA}(h,r,e))}, & p_c \le 0.5 \end{cases} \tag{28}$$

In fact, the main purpose of this objective function is to give correct triples a high score and incorrect triples a low score in the triplet prediction task. This allows for accurate prediction of correct entities when performing entity linking prediction tasks.

4 Experimental results and analyses

To evaluate the performance of the DRDA model, this study used the open-source dataset DBPedia as experimental data for performance assessment. Three evaluation metrics (MR, MRR, Hit@10) were used as evaluation criteria for the DRDA model. Additionally, each entity-link prediction task was run 10 times, and the average results of the model across the three metrics were calculated. Based on this, a variable analysis was conducted on the depth h of the deep bidirectional GRU network and the neighbour gain parameter p to observe the impact of these two hyperparameters on model performance.

Regarding the experimental parameter settings, the dimension of the word vectors is set to 300, and the dropout radio is set to 0.5. Each layer of the BiGRU encoding layer contains 400 nodes, and each layer of the GRU decoding layer contains 500 nodes. The number of encoding layers h is 3, the entity neighbour parameter δ is set to 1.5, the entity neighbour threshold ϑ is set to 5, the relation neighbour parameter μ is set to 1.5, and the relation neighbour threshold τ is set to 8.

To validate the comprehensive performance of DRDA, the DRDA model is compared with KBGC model (Lin et al., 2024), ConvRot model (Le et al., 2023), HPGAT model (Han et al., 2024).

Table 1 Experimental results of various models on the DBPedia50k dataset for the completion task

Model	*Head*			*Tail*		
	MR	*Hits@10*	*MRR*	*MR*	*Hits@10*	*MRR*
KBGC	134	0.66	0.69	96	0.67	0.56
ConvRot	203	0.64	0.59	138	0.65	0.49
HPGAT	225	0.62	0.56	91	0.67	0.53
DRDA	*96*	*0.31*	*0.25*	*79*	*0.54*	*0.57*

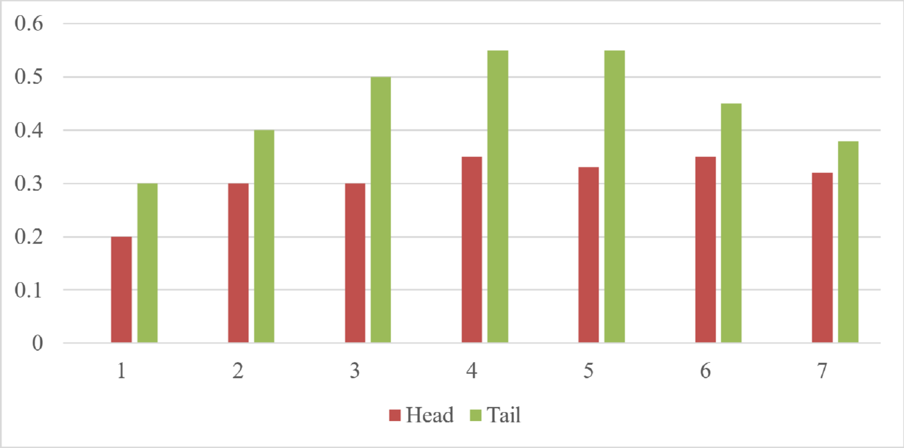
Table 2 Experimental results of various models on the DBPedia500k dataset for the completion task

Model	*Head*			*Tail*		
	MR	*Hits@10*	*MRR*	*MR*	*Hits@10*	*MRR*
KBGC	1,425	0.28	0.34	591	0.41	0.57
ConvRot	2,248	0.22	0.26	765	0.39	0.45
HPGAT	1,056	0.32	0.43	418	0.45	0.59
DRDA	*983*	*0.34*	*0.44*	*359*	*0.52*	*0.64*

Based on the experimental results in Tables 1 and 2, DRDA demonstrates performance improvements over the other three models in the head entity prediction task. Specifically, on the DBPedia50k dataset, DRDA outperformed the other models in the head prediction task by 38–1265 ranks in the MR metric, by 6%–9% in Hits@10, and by 25%–55% in MRR. These improvements can be attributed to the model's ability to integrate internal structural information with external semantic signals, thus enhancing the alignment between textual context and graph structure. However, in the tail entity prediction on the DBPedia50k dataset, DRDA's performance is more nuanced: although it achieved the lowest MR among all models, it underperformed KBGC by 13% in Hits@10 and showed a 1% higher MRR than KBGC, while outperforming ConvRot and HPGAT by 4–8% in MRR. This may suggest that in certain tail prediction cases, the sparse local subgraph structure limits the model's capacity to leverage semantic associations effectively. Alternatively, suboptimal neighbour aggregation strategies or hyperparameter configurations (e.g., in the neighbour layer) might introduce noise, impairing representation learning for tail entities.

As shown in Table 2, on the larger DBPedia500k dataset, DRDA again achieves consistent improvements in both head and tail entity prediction tasks. In head prediction, DRDA reduced the MR by 73–126 ranks, improved Hits@10 by 2%–12%, and boosted MRR by 2%–18%. In tail prediction, it outperformed the best baseline by 59–406 ranks in MR, by 7%–13% in Hits@10, and by 5%–19% in MRR. These results indicate that DRDA scales effectively with larger datasets. Unlike conventional knowledge representation models that rely solely on structured triples, DRDA benefits from jointly modelling textual semantics and local structural patterns, thereby enhancing relational reasoning over complex and expansive knowledge graphs.

Figure 4 Impact of the en-depth parameter on model performance (see online version for colours)



The results show that the DRDA model performs poorly on certain metrics on the small-scale dataset DBPedia50k. Therefore, to investigate the reasons for the poor performance of DRDA on small-scale datasets, this section explores the depth (en-depth) of the Bi-GRU encoding layer. Additionally, for the large-scale dataset DBPedia500k, this section observes the hyperparameters ϑ, μ, and δ of the neighbour layer to investigate their impact on the model's overall performance. First, we explore the encoding layer depth parameter (en-depth) for the open knowledge graph completion task on the small-scale dataset (DBPedia50k), observing the impact of the en-depth parameter on the head-tail entity link prediction task. As shown in Figure 4, the prediction results for tail entities are better than those for head entities, possibly because the head entities extract a lot of semantic information about the relationships between tail entities from the text information, making the tail entity representation learning more complete. Additionally, as the number of Bi-GRU encoding layers increases, the Hit@10 metric continues to rise. When the en-depth value reaches 4, Hit@10 reaches its peak. When the en-depth value exceeds 4, the Hit@10 metric begins to decrease, indicating that the learning capability of the DRDA model starts to decline. Therefore, it can be concluded that when the en-depth value is less than 4, the model is continuously learning deep associative semantic information between entities, resulting in a continuous increase in the Hit@10 value. When the en-depth value exceeds 4, the number of encoding layers becomes excessive, and entities and relationships are encoded into higher-dimensional semantic representations, leading to inaccurate learned associative information and ultimately reduced performance.

Additionally, to investigate the impact of the three hyperparameters in the neighbours layer on model performance, we plotted line charts showing how various performance metrics change with each hyperparameter. Furthermore, since subgraphs in large-scale datasets are more densely populated, entity neighbour information is richer, making it easier to observe experimental effects. Therefore, we selected the large-scale dataset DBPedia500k for testing.

Figure 5 Curve showing the variation of the hit@10 metric of the DRDA model with respect to the ϑ parameter (see online version for colours)

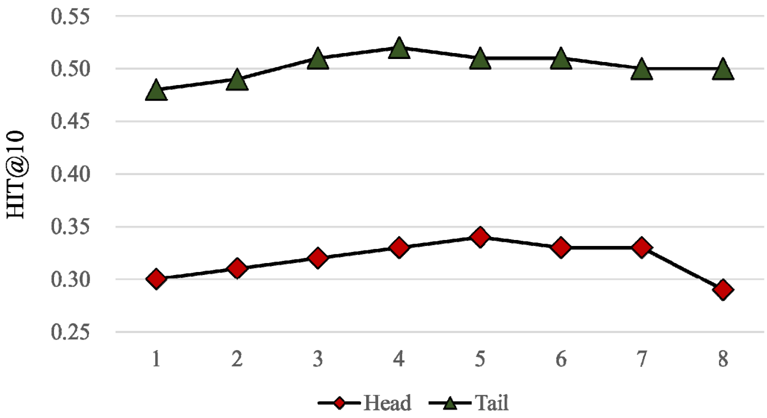
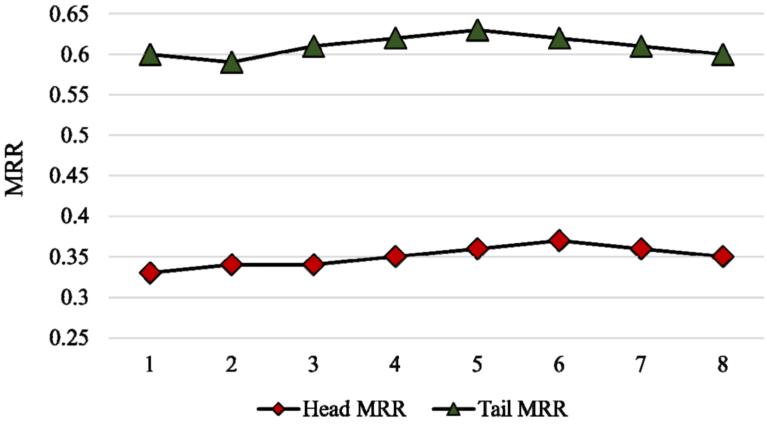


Figure 6 Curve showing the variation of the MRR metric of the DRDA model with respect to the ϑ parameter (see online version for colours)



First, we conducted experimental observations on the threshold parameter ϑ of the neighbour layer. As shown in Figures 5 and 6, (a) and (b) respectively present the curves of the Hit@10 and MRR metrics as a function of the ϑ parameter. From the figures, it can be observed that if the ϑ parameter is set too small, the performance of the Hit@10 and MRR metrics will decline. When ϑ reaches a certain value, the values of both metrics reach a peak, after which the model's performance begins to decline again. Therefore, based on the above analysis, it is concluded that setting ϑ to 4–6 is appropriate, as this yields the best model performance. Setting ϑ too high may result in the loss of neighbour information, while setting it too low may cause unreliable entities to be assigned higher weights, leading to performance degradation.

Figure 7 MRR, Hit@10, and hit@1 metrics of the DRDA model as a function of the δ parameter (see online version for colours)

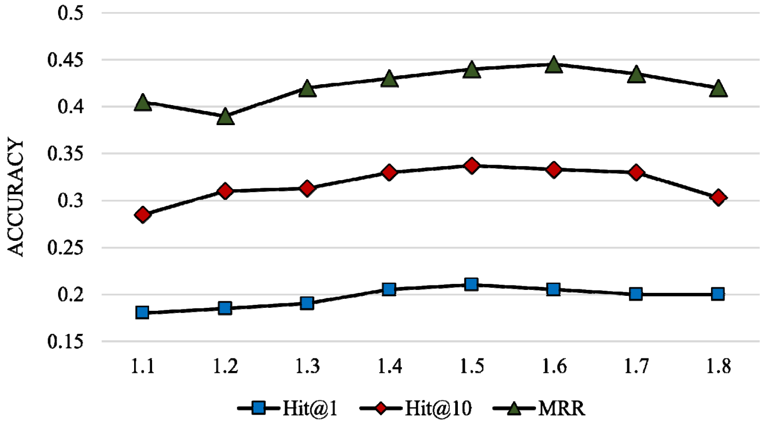
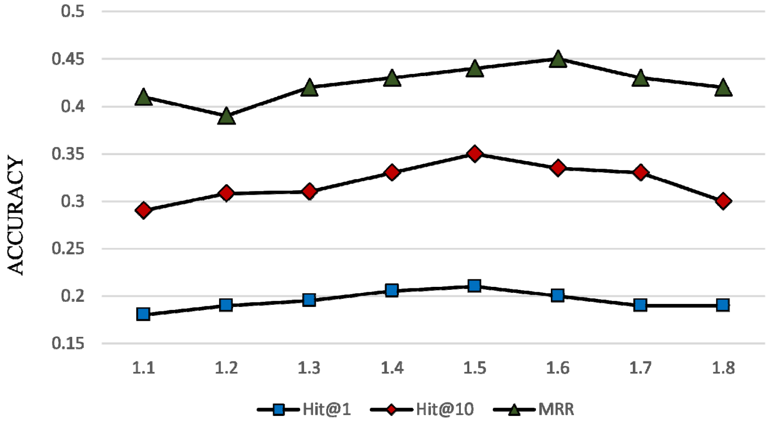


Figure 8 MRR, Hit@10, and hit@1 metrics of the DRDA model as a function of the μ parameter (see online version for colours)



The following is an experimental exploration of the hyperparameters for entities and relations. As shown in Figure 7, the three metrics (MRR, Hit@10, and Hit@1) of the DRDA model are plotted as line charts as a function of the entity neighbour parameter δ.

From the experimental results, it can be concluded that when the entity neighbour parameter δ is less than 1.5, the model's performance shows an overall upward trend. when δ is within the range of 1.4 to 1.6, the MRR and Hit@10 metrics show a slight downward trend, but the Hit@1 metric remains stable. Therefore, it can be concluded that setting δ within the range of 1.5 to 1.7 yields the best model performance. Additionally, if the entity neighbour hyperparameter is set too high, other weight information may be ignored, leading to a decline in model performance. If the parameter is set too low, the entity's neighbour information may not be prioritised, failing to influence the model's overall performance. As shown in Figure 8, similar experiments were conducted on the relationship neighbour hyperparameters. It can be observed that the trends of the relationship hyperparameter μ and the entity hyperparameter δ are largely similar, with the optimal model performance occurring within the range of 1.5–1.7. This may be because the relationship neighbour parameters are influenced by the entity neighbour parameters, resulting in similar trends in the impact of entity neighbour hyperparameters and relationship neighbour hyperparameters on the model's overall performance.

5 Conclusions

In this paper, a deep RNN-based approach for digital art knowledge graph construction was proposed, which effectively addresses the limitations of traditional methods in handling multimodal data and capturing complex semantic relationships. By integrating ResNet50 for visual feature extraction and CNN-BiLSTM-CRF for textual entity recognition, the multimodal data processing capability is significantly improved. The attention-enhanced bidirectional GRU network is introduced to capture long-term dependencies and contextual semantic information. Additionally, a dynamic completion mechanism with N-attention module is designed to predict missing entities and relationships through neighbour information integration. Through comparative experiments, it was found that:

1 Deep RNN-based completion models demonstrate improvements in knowledge graph completion accuracy, while bidirectional GRU networks prove effective at modelling complex semantic relationships in multimodal representations.

2 The introduction of the N-attention mechanism in deep RNNs can effectively enhance the weight information between entities and relationships.

3 The optimised knowledge graph construction method based on deep RNN models demonstrates good scalability on large-scale knowledge graph datasets, validating the effectiveness and practicality of DRNNs in digital art knowledge graph construction tasks.

The experimental outcomes validate the enhanced efficiency and superiority of the suggested method when compared to traditional knowledge graph construction approaches. However, the constructed knowledge graph is only targeted at image-based digital art works, which may limit the applicability of the model to other types of artistic media such as digital animations, interactive media, and virtual reality art. Future work should consider incorporating diverse artistic forms and multimedia content to further validate the model's effectiveness in more comprehensive digital humanities scenarios.

Acknowledgements

This work is supported by the Shanghai Educational Science Research Project in 2025 (No. 202212608005).

Declarations

All authors declare that they have no conflicts of interest.

References

Abu-Salih, B. and Alotaibi, S. (2024) 'A systematic literature review of knowledge graph construction and application in education', *Heliyon*, Vol. 10, No. 3, p.e25383.

Castellano, G., Digeno, V., Sansaro, G. and Vessio, G. (2022) 'Leveraging knowledge graphs and deep learning for automatic art analysis', *Knowledge-Based Systems*, Vol. 248, p.108859.

Chen, Z., Wang, Y., Zhao, B., Cheng, J., Zhao, X. and Duan, Z. (2020) 'Knowledge graph completion: a review', *IEEE Access*, Vol. 8, pp.192435–192456.

Chiu, J.P.C. and Nichols, E. (2016) 'Named entity recognition with bidirectional LSTM-CNNs', *Transactions of the Association for Computational Linguistics*, Vol. 4, pp.357–370.

Chun, J-H. (2011) 'A review of the characteristics of digital art expressed in contemporary fashion', *International Journal of Fashion Design, Technology and Education*, Vol. 4, No. 3, pp.161–171.

Duchi, J., Hazan, E. and Singer, Y. (2011) 'Adaptive subgradient methods for online learning and stochastic optimization', *The Journal of Machine Learning Research*, Vol. 12, pp.2121–2159.

Guo, L., Zhang, Q., Hu, W., Sun, Z. and Qu, Y. (2019) 'Learning to complete knowledge graphs with deep sequential models', *Data Intelligence*, Vol. 1, No. 3, pp.289–308.

Han, W., Liu, X., Zhang, J. and Li, H. (2024) 'Hierarchical perceptual graph attention network for knowledge graph completion', *Electronics*, Vol. 13, No. 4, pp.721–732.

Hochreiter, S. and Schmidhuber, J. (1997) 'Long short-term memory', *Neural Computation*, Vol. 9, No. 8, pp.1735–1780.

Hogan, A., Blomqvist, E., Cochez, M., D'amato, C., Melo, G.D., Gutierrez, C., Kirrane, S., Gayo, J.E.L., Navigli, R., Neumaier, S., Ngomo, A-C.N., Polleres, A., Rashid, S.M., Rula, A., Schmelzeisen, L., Sequeda, J., Staab, S. and Zimmermann, A. (2021) 'Knowledge graphs', *ACM Computing Surveys*, Vol. 54, No. 4, pp.1–37.

Huang, Y.Y., Yu, S.S., Chu, J.J., Fan, H.H. and Du, B.B. (2023) 'Using knowledge graphs and deep learning algorithms to enhance digital cultural heritage management', *Heritage Science*, Vol. 11, No. 1, pp.204–229.

Le, T., Le, N. and Le, B. (2023) 'Knowledge graph embedding by relational rotation and complex convolution for link prediction', *Expert Systems with Applications*, Vol. 214, p.119122.

Li, Z., Zhao, Y., Zhang, Y. and Zhang, Z. (2022) 'Multi-relational graph attention networks for knowledge graph completion', *Knowledge-Based Systems*, Vol. 251, p.109262.

Lin, H., Bao, J., Hu, N., Zhao, Z., Bai, W. and Li, D. (2024) 'Knowledge graph completion for high-speed railway turnout switch machine maintenance based on the multi-level KBGC model', *Actuators*, Vol. 13, No. 10, pp.410–426.

Peng, C., Xia, F., Naseriparsa, M. and Osborne, F. (2023) 'Knowledge graphs: opportunities and challenges', *Artificial Intelligence Review*, Vol. 56, pp.13071–13102.

Schuster, M. and Paliwal, K.K. (1997) 'Bidirectional recurrent neural networks', *IEEE Transactions on Signal Processing*, Vol. 45, No. 11, pp.2673–2681.

Wang, L., Sun, C., Zhang, C., Nie, W. and Huang, K. (2023) 'Application of knowledge graph in software engineering field: a systematic literature review', *Information and Software Technology*, Vol. 164, p.107327.

Ye, Z., Kumar, Y.J., Sing, G.O., Song, F. and Wang, J. (2022) 'A comprehensive survey of graph neural networks for knowledge graphs', *IEEE Access*, Vol. 10, pp.75729–5741.

Yu, C-L., Wen, H-M., Ko, P-C., Shu, M-H. and Wu, Y-S. (2024) 'Automatic construction and optimization method of enterprise data asset knowledge graph based on graph attention network', *Journal of Radiation Research and Applied Sciences*, Vol. 17, No. 3, p.101023.

Zhao, Z., Luo, X., Chen, M. and Ma, L. (2024) 'A Survey of knowledge graph construction using machine learning', *Computer Modeling in Engineering and Sciences*, Vol. 139, No. 1, pp.225–257.

Zhong, L., Wu, J., Li, Q., Peng, H. and Wu, X. (2023) 'A comprehensive survey on automatic knowledge graph construction', *ACM Computing Surveys*, Vol. 56, No. 4, pp.1–62.