Combining Deep Learning with Knowledge Graph for Design Knowledge Acquisition in Conceptual Product Design

Huang, Yuexin; Yu, Suihuai; Chu, Jianjie; Su, Zhaojing; Cong, Yangfan; Wang, Hanyu; Fan, Hao

DOI
10.32604/cmes.2023.028268

Publication date
2024

Document Version
Final published version

Published in
CMES - Computer Modeling in Engineering and Sciences

Citation (APA)
Huang, Y., Yu, S., Chu, J., Su, Z., Cong, Y., Wang, H., & Fan, H. (2024). Combining Deep Learning with Knowledge Graph for Design Knowledge Acquisition in Conceptual Product Design. *CMES - Computer Modeling in Engineering and Sciences*, *138*(1), 167-200. https://doi.org/10.32604/cmes.2023.028268

Computer Modeling in Engineering & Sciences

Tech Science Press

DOI: 10.32604/cmes.2023.028268

ARTICLE

Combining Deep Learning with Knowledge Graph for Design Knowledge Acquisition in Conceptual Product Design

Yuexin Huang[1,2], Suihuai Yu[1], Jianjie Chu[1,*], Zhaojing Su[1,3], Yangfan Cong[1], Hanyu Wang[1] and Hao Fan[4]

[1]Key Laboratory of Industrial Design and Ergonomics, Ministry of Industry and Information Technology, Northwestern Polytechnical University, Xi'an, 710072, China

[2]School of Industrial Design Engineering, Delft University of Technology, Delft, 2628 CE, The Netherlands

[3]Department of Industrial Design, College of Arts, Shandong University of Science and Technology, Tsingtao, 266590, China

[4]College of Computer Science and Technology, Zhejiang University, Hangzhou, 310027, China

*Corresponding Author: Jianjie Chu. Email: cjj@nwpu.edu.cn

Received: 08 December 2022 Accepted: 31 March 2023 Published: 22 September 2023

ABSTRACT

The acquisition of valuable design knowledge from massive fragmentary data is challenging for designers in conceptual product design. This study proposes a novel method for acquiring design knowledge by combining deep learning with knowledge graph. Specifically, the design knowledge acquisition method utilises the knowledge extraction model to extract design-related entities and relations from fragmentary data, and further constructs the knowledge graph to support design knowledge acquisition for conceptual product design. Moreover, the knowledge extraction model introduces ALBERT to solve memory limitation and communication overhead in the entity extraction module, and uses multi-granularity information to overcome segmentation errors and polysemy ambiguity in the relation extraction module. Experimental comparison verified the effectiveness and accuracy of the proposed knowledge extraction model. The case study demonstrated the feasibility of the knowledge graph construction with real fragmentary porcelain data and showed the capability to provide designers with interconnected and visualised design knowledge.

KEYWORDS

Conceptual product design; design knowledge acquisition; knowledge graph; entity extraction; relation extraction

1 Introduction

Conceptual product design plays an essential role in product development and affects approximately 80% of manufacturing costs [1]. Conceptual product design is also a knowledge-intensive activity requiring design-related knowledge for inspiration generation, design analysis, design comparison and other tasks. Designers have to deal with large amounts of fragmentary, unstructured and disorganised data and make decisions based on limited knowledge. However, there is a distance between fragmentary data and the required design knowledge. As a result, it is important to provide designers with useful, reasoning and visual design knowledge. Historical research focuses on design

databases and designs knowledge management frameworks for design knowledge acquisition. For example, the Advanced Industrial Research Institute of Japan created the Global Fashion Style Culture Database [2], and the Chinese Academy of Arts constructed the Intangible Cultural Heritage Database [3]. Furthermore, many researchers have proposed design knowledge management frameworks to help designers acquire design knowledge, such as Extensible markup language (XML) [4], Resource description framework (RDF) [5] and Web ontology language (OWL) [6].

Despite significant contributions made by various researchers towards developing design databases, these databases are often constructed manually, which restricts their scalability. Conventional design knowledge management techniques need more automation, visualisation and reasoning capabilities, making it challenging for designers to obtain valuable design knowledge from large, fragmented data effectively. Therefore, it is critical to present design knowledge in a visual, intuitive and automatic way to facilitate the acquisition of valuable design knowledge for designers.

A knowledge graph is a semantic network that interconnects diverse data by representing entities and the relations between them [7,8]. The knowledge graph creates connections between disparate data sources, integrates isolated data sources, bridges structured and unstructured data, and provides a visual representation of information flow. Data storage and management using the knowledge graph are becoming increasingly popular to get the required valuable information from fragmentary data. It has been applied in electronic commerce [9], CAD models [10], smart manufacturing [11] and other areas. In the research area of conceptual product design, specific tasks such as product attribute acquisition [12], design knowledge management [13,14], and unified process information modelling [15] have been applied knowledge graph. The knowledge graph can also be applied in design information visualization [16], design workflow arrangement [17], and modelling process execution sequence [18]. Nevertheless, there are few studies on obtaining design knowledge from massive fragmentary data for conceptual product design. Compared with conventional methods for acquiring design knowledge, such as XML [4], RDF [5], and OWL [6], knowledge graph has significant advantages in automation, visualisation [16] and reasoning [12]. Therefore, this study proposes a design knowledge acquisition method using the knowledge graph for conceptual product design.

The process of constructing a knowledge graph is centred around several steps, including domain ontology construction, knowledge extraction, knowledge fusion, knowledge reasoning, and graph application. Among these steps, entity extraction and relation extraction are considered the core steps in constructing a knowledge graph [15]. As for the entity extraction, statistical methods and machine learning models are utilised to extract entities from unstructured text in the early stage, such as hidden Markov [19], support vector machines [20,21] and conditional random fields (CRF) [22,23]. Then, deep learning models have been widely adopted in entity extraction by using sequence labelling tasks, and accuracy has reached over 70%. For example, Li et al. [24] used Bidirectional LSTM CRF (BiLSTM-LSTM-CRF) to extract medical entities from electronic medical records. Dou et al. [25] proposed a method for extracting entities from intangible cultural heritage using the ID-CNN-CRF algorithm. Afterwards, various tasks in natural language processing have been improved by large-scale pre-training models, such as BERT [26], ERNIE [27], UniLM [28], and XLNet [29]. These models have the advantage of high accuracy, while large parameters of pre-training models require a big memory capacity and longer training hours. Model parameters also have a significant impact on communication overhead. Thus, using pre-training models with large parameters becomes challenging due to memory limitations and communication overhead.

To extract relations, semantic rules and templates are initially used, followed by deep learning algorithms. There have been numerous studies on how to improve the performance of relation

extraction models, with a focus on designing better models or improving the representation of the input data. Some researchers have applied neural network models, such as Convolutional Neural Networks (CNNs) [30], Recurrent Neural Networks (RNNs) [31], and Long-Short-Term Memory (LSTM) networks [32], to extract relationships from text data. Additionally, different methods have been proposed for improving the representation of input sentences, such as applying positional embeddings, multi-instance learning, and tree-like structures. For example, Tai et al. [33] proposed a tree-like LSTM model. Rönnqvist et al. [34] utilised the BiLSTM model to extract relations. Xu et al. [35] established a word-level database from hundreds of Chinese articles for relation extraction. Zhang et al. [36] established a character-based and word-based model called Lattice LSTM to extract relations. However, the performance of relation extraction models is significantly impacted by the quality of the input data, especially the word segmentation and the handling of polysemous words. Most classical relation extraction methods rely on characters or word vectors as input, and the performance of the models is affected by the quality of the segmentation. In addition, the existing models do not consider the fact that input sentences contain many polysemous words, which can affect the performance of the models. To address the above issues, further research is needed to handle the input data better and to improve the representation of the input sentences in relation to extraction.

With a literature review of design knowledge acquisition methods, entity extraction models and relation extraction models, there are three main challenges in using the knowledge graph to acquire design knowledge:

(1) Establish a knowledge graph framework according to design knowledge requirements. The majority of knowledge graph research focuses on the manufacturing industry, which includes manufacturing documents [37], product attributes acquisition [12], and semantic relations identification [38]. Knowledge graph frameworks from other research areas cannot be directly applied due to different data characteristics and knowledge requirements. Thus, it is challenging to propose a generic framework for knowledge graph construction to facilitate the design knowledge acquisition for conceptual product design.

(2) Entity extraction and relation extraction models compatible with design data are challenging. It is difficult to obtain design knowledge from unstructured text because of memory limitation and communication overhead in entity extraction. In the relation extraction process, segmentation errors and polysemy words influence relation extraction. Consequently, entity extraction and relationship extraction models should be developed according to the characteristics of the design data.

(3) Specific design tasks limit the performance of the knowledge graph. Compared with other research areas that utilise knowledge graph technology, design knowledge acquisition for conceptual product design involves unique processes, such as the acquisition of design inspiration and the comparison and management of design knowledge. A knowledge graph should be constructed to address the existing problem and applied to the conceptual product design process.

To cope with these challenges, this study proposes a design knowledge acquisition method for conceptual product design combining deep learning with knowledge graph to overcome the above obstacles. Specifically, the generic knowledge graph-driven design knowledge acquisition framework is proposed to facilitate design knowledge acquisition in conceptual product design. Furthermore, the design knowledge extraction model is proposed to extract design knowledge from massive fragmentary unstructured data automatically. Finally, validating the knowledge graph in the case study allows for the acquisition of interconnected and visualised design knowledge.

The following paragraphs of this study are organised as below: Section 2 proposes the knowledge graph-driven design knowledge acquisition framework. The entity extraction and relation extraction models are developed in Section 3 for extracting design-related knowledge from massive fragmentary data. Section 4 demonstrates a case study. The discussion is presented in Section 5. Section 6 provides the conclusion of this study and an overview of future work.

2 The Framework of Knowledge Graph-Driven Design Knowledge Acquisition

This study aims to help designers acquire valuable design knowledge from massive fragmentary data. In accordance with the objectives, a framework for knowledge graph-driven design knowledge acquisition is proposed. As illustrated in Fig. 1, the framework has five layers, including the data resources layer, domain ontology layer, entity extraction layer, relation extraction layer and knowledge graph application layer.

(1) Data resources layer. The data resource layer is the fundamental resource layer with design-related data such as design inspiration, historical product data and domain expert data. Design inspiration data, such as design websites, creative cultural websites, and other design platforms, can provide designers with a wealth of inspiration for their design work. Historical product data provides designers with references and inspirations, such as product databases, design rules, heterogeneous models, and process information. Domain expert data contains multi-disciplinary data on marketing, manufacturing, and maintenance. Historical product data are presented as an example in this study. Design inspiration data, domain expert data and other design-related data can follow the same steps to construct a knowledge graph. These resources serve as a valuable source of information, helping designers to broaden their perspectives, generate new ideas, and explore creative solutions to design challenges. Through the use of the knowledge graph, design-related data can be effectively organised, analysed, and visualised, providing designers with an accessible and comprehensive view of design inspiration from multiple sources.

(2) Domain ontology layer. There is a distance between design knowledge and fragmentary factual data. Therefore it is essential to construct a unified framework of entities and relations. Domain ontology is the design of knowledge structure in the knowledge graph. In order to construct the domain ontology, the concepts are defined and should satisfy the principles of independence and minimisation. Then, the concepts are connected using a top-down approach, and the relations are defined. The domain ontology layer is described further in Section 4.1.

(3) Entity extraction layer. The entity extraction layer aims to extract design-related entities from large amounts of fragmentary text automatically. This study introduces a lite BERT (ALBERT) [26] in entity extraction task to encode input sentences. The entity extraction model is discussed further in Section 3.2.

(4) Relation extraction layer. The relation extraction layer is responsible for extracting relations between entities. Thus, entity extraction and relation extraction models result in the automation of the conversion of fragmented data into design knowledge, creating a centralised repository of design knowledge that is readily accessible, shareable and conducive to collaboration on the platform. This study proposes a multi-grained relation extraction framework to address segmentation errors and polysemy ambiguity. The relation extraction model is discussed in detail in Section 3.3.

(5) Knowledge graph application layer. The knowledge graph platform is developed to provide designers with interconnected and visualised design knowledge for idea generation, design

comparison and management. All the design-related data could follow the above steps to construct a knowledge graph. In the following, a case study is conducted using creative cultural product data. The same steps proposed above can be applied to design inspiration, domain expert data and other design-related data. The knowledge graph application layer is shown in Section 4.4.

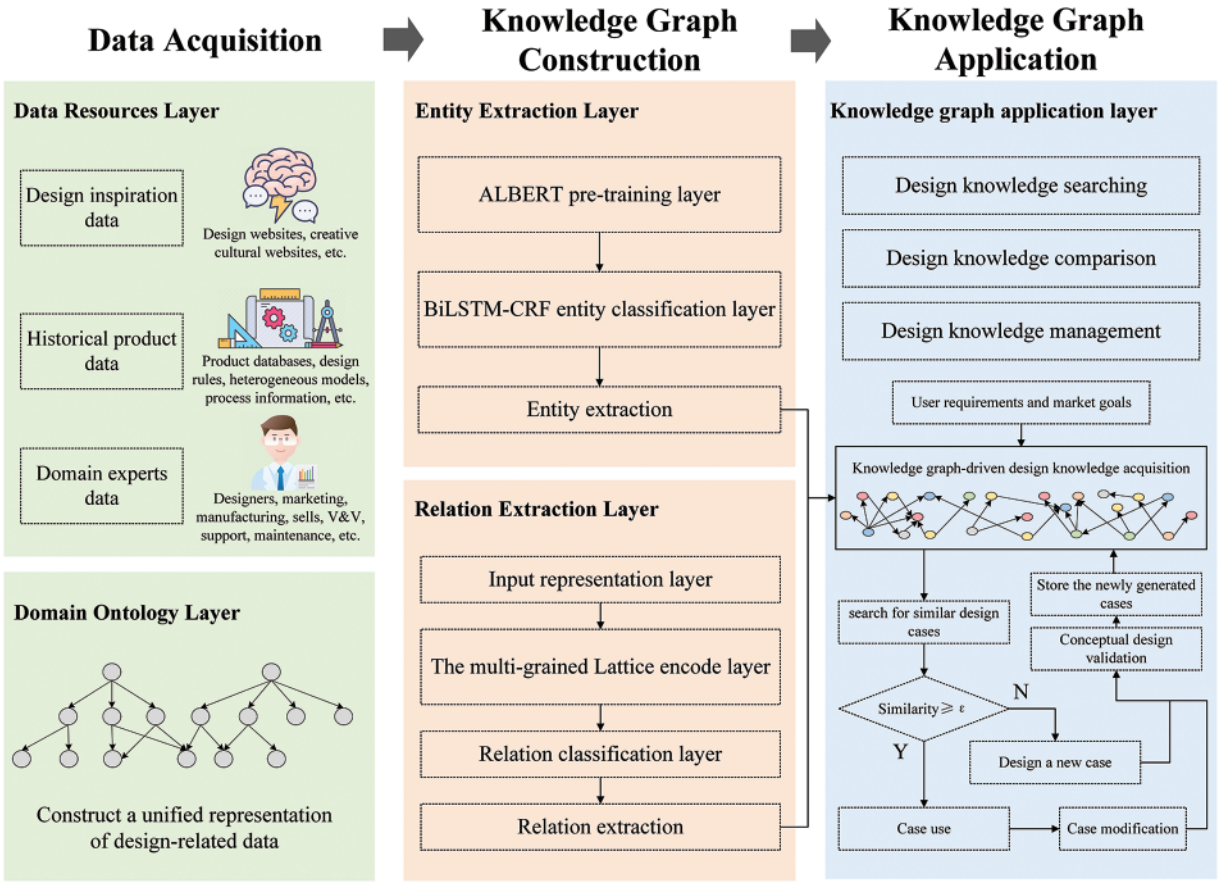

Figure 1: The framework of knowledge graph-driven design knowledge acquisition

Integrating knowledge graph and conceptual product design has many benefits for designers. Firstly, it supports several tasks of design inspiration, inspiration collection, design comparison, knowledge management and design practice in the conventional conceptual product design, as depicted in Fig. 2. Secondly, integrating design-related data and the knowledge graph addresses the gap between fragmentary design data and design knowledge by transforming data into design knowledge by applying the knowledge graph. The knowledge graph allows for a more streamlined and efficient design process, as relevant information can be quickly and easily retrieved. Thirdly, the use of deep learning algorithms for entity extraction and relation extraction from unstructured data facilitates the automation of the transformation of fragmentary data into design knowledge. It automatically forms a centralised repository of design knowledge that can be easily accessed, shared and collaborated on the platform. It reduces designers' cognitive load, allowing them to focus on creative design, and helps designers create more informed and practical design solutions. Lastly, the knowledge graph supports conceptual design methodologies such as Function-Behaviour-States (FBS), ensuring designs are

developed consistently and with clear functional requirements. By representing functions, behaviours, and states, the model can provide a structured framework for organising and representing design knowledge, making it easier for designers to access and use the information in their design process. Overall, knowledge graph-driven design knowledge acquisition for conceptual product design can significantly enhance the efficiency and effectiveness of the design process by providing a visualised framework for organising and representing design knowledge, making it easier for designers to access and use information in the design process.

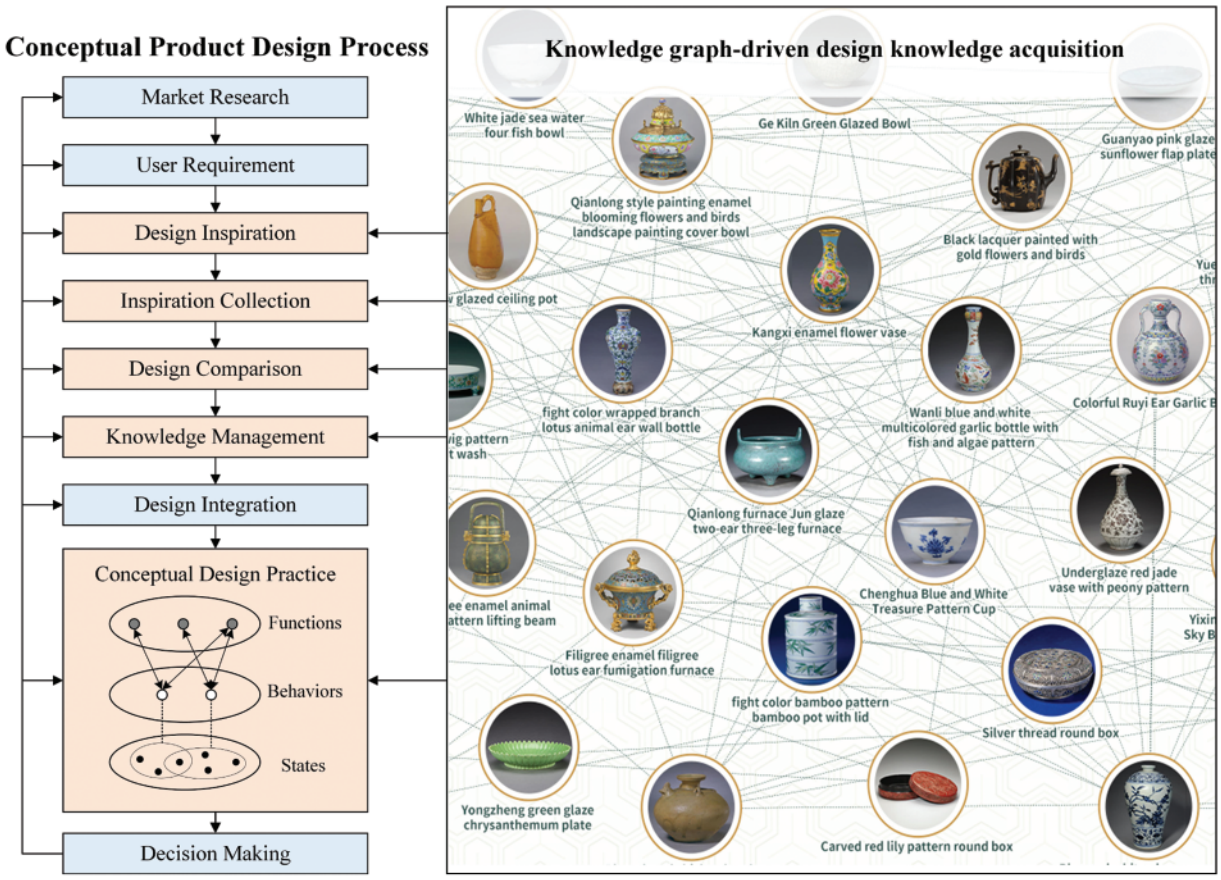

Figure 2: Knowledge graph-driven design knowledge acquisition for conceptual product design

3 The Design Knowledge Extraction Model for Knowledge Graph Construction

The extraction of design-related entities and relations from unstructured data is critical in knowledge graph construction. For example, we would like to extract design-related entities such as ‘entwined branches’, ‘peony’, ‘lotus’, and ‘blue glaze’ from the sentence ‘carved on the abdomen with entwined branches and peony, the lower abdomen with the double layer of lotus, is blue glaze’. It is also necessary to extract relations such as ‘is glazed of’, ‘has a pattern of’, and ‘is the shape of’ for knowledge graph construction. However, entity extraction suffers from memory limitations and long training time when using well-known pre-training models like BERT [26] and XLNet [29]. In addition, segmentation errors and polysemy ambiguity influences the relation extraction. As a result, this section proposes a knowledge extraction model to overcome the above obstacles.

3.1 The Design Knowledge Extraction Model

The first challenge is the memory limitation and communication overhead in the entity extraction task. Pre-training models that have lots of parameters [26,29,39] often improve performance on subsequent tasks and are crucial for achieving the state-of-art performance of entity extraction [26,40]. Considering the significance of the large model size, most pre-training models have many parameters, which may exceed a million or even a billion. Model parameters have an impact on communication overhead, and the training speed is also affected. As a result, the increasing size of the model results in memory limitations and longer training hours. The memory limitation problem has already been addressed through model parallelisation [41,42] and intelligent memory management [43,44], while these solutions do not take into account the communication overhead. To solve both memory limitation and communication overhead, some researchers proposed ALBERT [26]. ALBERT presents two parameter-reduction strategies to address the memory limitations of the hardware and the communication overhead. Parameter reduction strategies include factorised embedding parameterisation and cross-layer parameter sharing. In addition, the ALBERT approach utilises a self-supervised loss to model the inter-sentence coherence of sentences. ALBERT requires fewer parameters than XLNet [29] and RoBERTa [39] to achieve similar performance in entity extraction [26]. Thus, introducing ALBERT into entity extraction tasks can enhance the efficiency of entity extraction task.

The second challenge is segmentation errors and polysemy ambiguity in the relation extraction task. Relation extraction is the second step of design knowledge extraction, and most classical relation extraction methods use character vectors or word vectors as their input [34,35,45,46]. In character-based relation extraction methods [45], each input sentence is treated as a series of characters, which captures fewer features than word-based methods since they cannot fully leverage word-level information. The word-based relation extraction methods [36] typically perform the word segmentation, generate word sequences, and feed the word sequences into the machine learning models. Thus, these methods are considerably affected by segmentation quality. In addition, the existing relation extraction models do not consider the fact that input sentences contain many polysemous words, which limits their ability to explore the deeper semantic features of sentences. As a result, segmentation errors and polysemous words affect the results of relation extraction. To address the problem of segmentation errors and polysemy ambiguity in relation extraction, a multi-grained Lattice framework is proposed to employ internal and external information comprehensively. In order to deal with segmentation errors, the model uses a Lattice-based framework that integrates word-level features with character-based features. To address the problem of polysemy ambiguity in relation extraction, the relation extraction model utilises HowNet [47] knowledge base, which contains Chinese polysemantic words with manual annotations. Therefore, the multi-grained model with multi-granularity information and a polysemantic knowledge base can fully leverage the semantic information to enhance the effectiveness of relation extraction.

Accordingly, the study implements the design knowledge extraction model as an entity extraction module introduced by ALBERT and a relation extraction module based on the multi-grained Lattice, as shown in Fig. 3. The design knowledge extraction model overcomes memory limitation and communication overhead of entity extraction and enhances the performance of relation extraction.

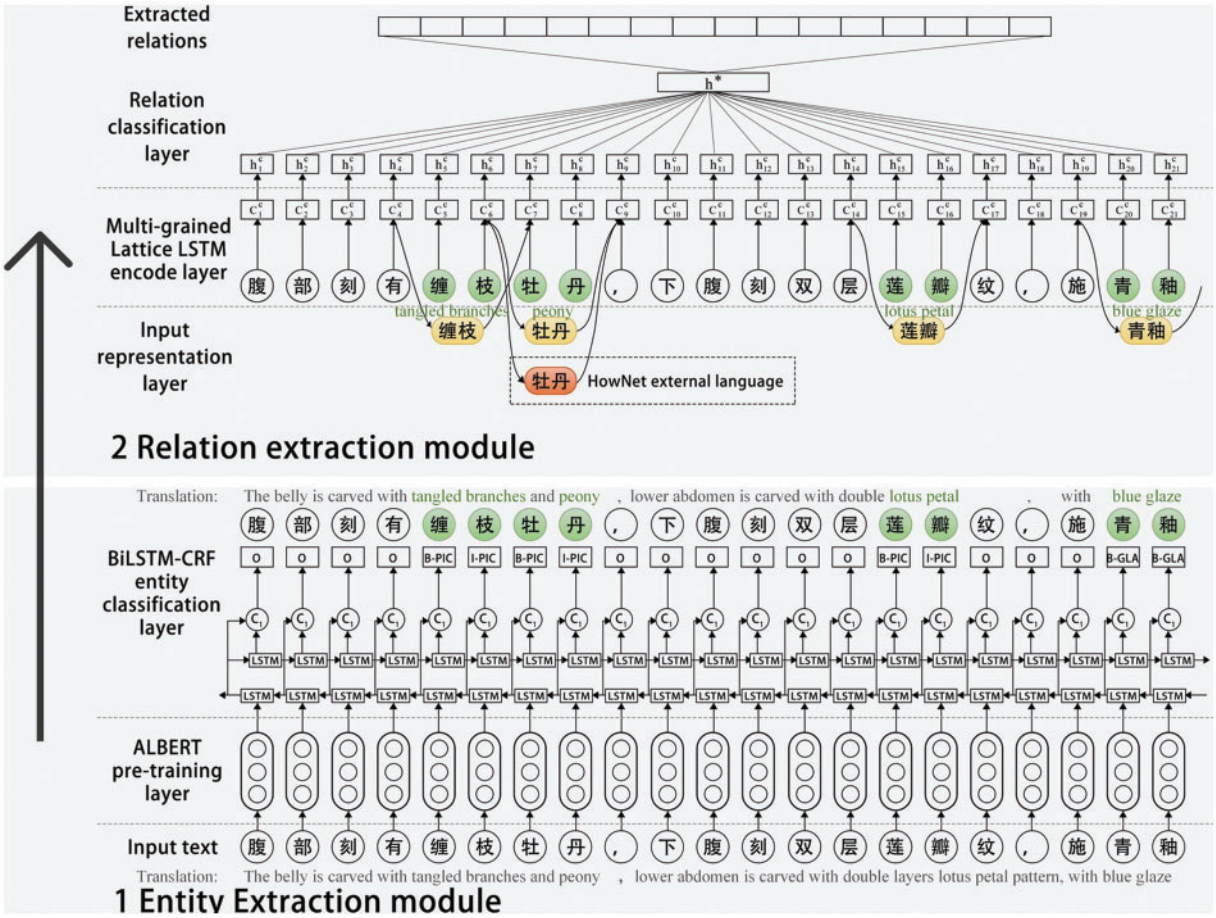

Figure 3: The design knowledge extraction model

3.2 Entity Extraction Module

The entity extraction process usually views the sentences as token sequences labelled with the BIO format. The character will be labelled as 'O' if it is not an entity, and it will be labelled as 'B' if it is the beginning of an entity. The end of the entity is labelled as 'I', followed by the entity type. For example, Fig. 4 shows an example with the BIO labels. The input sentence is 'The belly is carved with tangled branches, peony, covered with blue glaze'. The word 'tangled branches' are labelled as 'B-PAT' and 'I-PAT'.

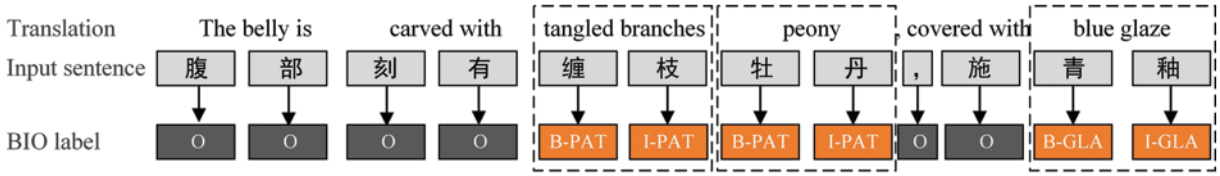

Figure 4: The BIO token sequences for entity extraction

The entity extraction module contains the ALBERT pre-training layer and BiLSTM-CRF entity classification layer. Firstly, the entity extraction module converts each input word into a word vector utilizing ALBERT. Secondly, the Bidirectional long short term memory (BiLSTM) layer is utilized to acquire input features. The Conditional random fields (CRF) layer labels the sequence of sentences. Finally, the entity extraction module repeats the above procedures until all input sentences are labelled.

3.2.1 ALBERT Pre-Training Layer

The entity extraction module uses ALBERT as the word encoding layer. The sentence containing m words is represented by $x = (x_1, x_2, \ldots, x_m)$, where x_i refers to *i-th* word of the sentence in the vocabulary, creating the one-hot vectors. Each word x_i becomes a dense vector with low-dimension $x_i \in R^d$, where d is the dimension based on the pre-training embedding matrix in the ALBERT pre-training layer.

The fundamental component of ALBERT is a transformer encoder [48] with GELU nonlinearities [49], which is similar to the structure of BERT. The embedding size of the vocabulary is presented as E, L represents the encoder layer number, and H refers to the hidden size in accordance with BERT notation. The size of feed-forward was set to $4H$ and attention heads were set to $H/64$ in accordance with BERT [26]. ALBERT has three significant advantages over BERT, RoBERTa and other pre-training models. An important improvement is the factorized embedding parameterization, which reduces the number of parameters. By separating the vocabulary embedding matrix into two smaller matrices, it was possible to separate the hidden layer size from the vocabulary embedding size. The separation method allows for an increase in the hidden size without substantially increasing the vocabulary embedding's parameter size. Another improvement is cross-layer parameter sharing. In this way, the parameters of the network are prevented from increasing as network depth grows. These two methods decrease the number of parameters for BERT while maintaining performance and boosting parameter effectiveness. Parameter reduction strategies mentioned above also serve as a form of regularization, which helps to stabilize the training process and facilitate generalization. ALBERT is further enhanced by introducing a self-supervised loss for sentence-order prediction. The aim of sentence-order prediction is to overcome the inefficiency of the next sentence prediction loss presented in BERT [39,50]. As a consequence, ALBERT can be scaled up to a much larger configuration with fewer parameters than BERT, and it will perform better with fewer parameters than BERT.

3.2.2 BiLSTM-CRF Entity Classification Layer

With the word embedding obtained from the ALBERT pre-training layer, the BiLSTM-CRF entity classification layer tags the input sentences and predicts entities. The BiLSTM is employed to acquire the features of the input sentences. BiLSTM uses the embedded word sequence $(x_1, x_2, \ldots, x_m)$ as an input. We combine the output sequences of the forward $\left(\overrightarrow{h_1}, \overrightarrow{h_2}, \ldots, \overrightarrow{h_m}\right)$ and backward LSTMs $\left(\overleftarrow{h_1}, \overleftarrow{h_2}, \ldots, \overleftarrow{h_m}\right)$ based on their position $h_t = \left[\overrightarrow{h_t}; \overleftarrow{h_t}\right] \in R^n$ to obtain the complete hidden state sequence $(h_1, h_2, \ldots, h_m) \in R^{m \times n}$. The subsequent linear layer maps n-dimensional hidden states into k-dimensional states, with k indicating the number of labels in the tagging scheme. Consequently, the features of each sentence are obtained and shown as the matrix $P = (p_1, p_2, \ldots, p_n) \in R^{m \times k}$.

CRF parameters can be seen in the matrix $A \in R^{(k+2)\times(k+2)}$, where A_{ij} is the score of transition from *i-th* label to *j-th* label. Considering a sequence of labels $y = (y_1, y_2, \ldots, y_m)$, the label sequence's score is calculated using the following formula:

$$\text{score}(x, y) = \sum_{i=1}^{m} P_{i,y_i} + \sum_{j=1}^{m+1} A_{y_{j-1},y_j}. \tag{1}$$

The total scores of the words in the sentence equal the score of the sentence. This score is calculated from BiLSTM's output matrix P and CRF's transition matrix A. Furthermore, the Softmax function is used to calculate the normalised probability:

$$P(y \mid x) = \frac{e^{(\text{score}(x,y))}}{\sum_{y'=1}^{k} e^{(\text{score}(x,y'))}}. \tag{2}$$

For logarithmic likelihood maximization during model training, the logarithmic likelihood of a training sample (x, y^x) is given by the following equation:

$$\log P\left(y^x \mid x\right) = \text{score}\left(x, y^x\right) - \log \sum_{y'} \exp\left(\text{score}\left(x, y'\right)\right). \tag{3}$$

During the training process, the log-likelihood function is maximized and the Viterbi algorithm is applied to determine the best route for prediction using dynamic programming. The equation is as follows:

$$y^* = \arg\max_{y'} \text{score}\left(x, y'\right). \tag{4}$$

3.3 Relation Extraction Module

Most existing relation extraction models suffer from segmentation errors and polysemy ambiguity, as discussed in Section 3.2. This study proposes the multi-grained Lattice model to incorporate the word-level information into character sequence inputs to prevent errors during the segmentation process. Moreover, the polysemous words are modelled using external linguistic knowledge to reduce polysemous ambiguity.

The relation extraction model extracts semantic relations between two target entities given an input sentence and two target entities. There are three layers in this model, as is depicted in Figs. 3 and 5. The first layer is the input representation layer. The input presentation layer represents every word and character in a sentence in which there are two target entities. The second layer is the multi-grained Lattice LSTM encode layer, which employs a Lattice LSTM network [51] to develop each input instance's distributed representation. It also introduces external knowledge to solve the polysemy ambiguity. The last layer is the relation classification layer. The character-level mechanism is modified to integrate features after learning the hidden states. A Softmax classifier is applied to forecast relations based on sentence representations.

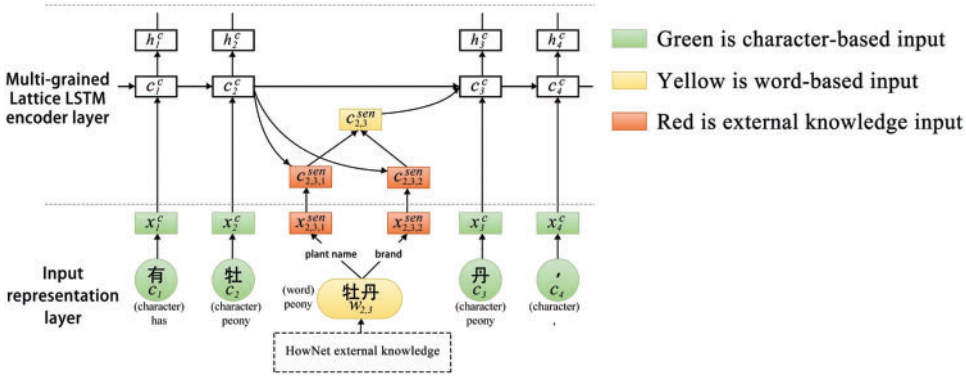

Figure 5: The multi-grained languages in the relation extraction module

3.3.1 Input Representation Layer

The input of the representation layer is a sentence s containing labelled entities. To make use of multi-grained data, both character-level and word-level representations are presented.

(1) Character-level representation

The proposed model processes each sentence as a character sequence as its direct input. Given an input sentence s consisting of M characters $s = \{c_1, \ldots, c_M\}$, each character c_i is mapped to a vector of d^c dimensions using the Skip-gram model, represented as $x_i^{ce} \in \mathbb{R}^{d^c}$. Position embeddings are also used to define specific entity pairs. The position embedding is the relative distance from the current character to the head and tail entities [52]. In particular, the relative distances between character c_i and marked entities are represented as p_i^1 and p_i^2, and the p_i^1 can be calculated as follows:

$$p_i^1 = \begin{cases} i - b^1 & i < b^1 \\ 0 & b^1 \le i \le e^1 \\ i - e^1 & i > e^1 \end{cases} . \tag{5}$$

where b^1 and e^1 represent the starting and ending indices of the head entity. The calculation of p_i^2 follows the same formula as in Eq. (5). Using the position embedding table, p_i^1 and p_i^2 are converted into two vectors, represented as $\boldsymbol{x}_i^{p1} \in \mathbb{R}^{d^p}$ and $\boldsymbol{x}_i^{p2} \in \mathbb{R}^{d^p}$.

The input representation for character c_i is $\boldsymbol{x}_i^c \in \mathbb{R}^d$ $(d = d^c + 2 \times d^p)$, is concatenated with character embedding $\boldsymbol{x}_i^{ce}$, position embeddings $\boldsymbol{x}_i^{p1}$ and $\boldsymbol{x}_i^{p2}$. Hence, $\boldsymbol{x}_i^c = \left[\boldsymbol{x}_i^{ce}; \boldsymbol{x}_i^{p1}; \boldsymbol{x}_i^{p2}\right]$. Then, the encoding of characters $\boldsymbol{x}^c = \left\{\boldsymbol{x}_1^c, \ldots, \boldsymbol{x}_M^c\right\}$ will be input directly into the model.

(2) Word-level representation

In addition to using character sequences as inputs, the model also utilizes word-level information to capture word-level characteristics accurately. Nevertheless, there are a number of segmentation errors and polysemy ambiguity in many words. A polysemy language base called HowNet is introduced to represent word senses to address this issue. Given the word $w_{b,e}$, obtaining all K senses from HowNet, $Sense\left(w_{b,e}\right)$ is used to represent the senses set of $w_{b,e}$, SAT model [53] is used to convert each sense $sen_k^{(w_{b,e})} \in Sense\left(w_{b,e}\right)$ into a real-valued vector $\boldsymbol{x}_{b,e,k}^{sen} \in \mathbb{R}^{d^{sen}}$. The SAT model is based on the Skip-gram, which is able to simultaneously learn word and sense representations. $w_{b,e}$ is represented as a vector set with the notation $x_{b,e}^{sen} = \left\{\boldsymbol{x}_{b,e,1}^{sen}, \ldots, \boldsymbol{x}_{b,e,K}^{sen}\right\}$.

3.3.2 The Multi-Grained Lattice LSTM Encode Layer

Each character sequence provides direct input to the multi-grained Lattice LSTM encode layer, and all possible words in the lexicon $\mathbb{D}$ are considered together. The encode layer outputs the hidden state vectors $\boldsymbol{h}$ after training. This layer is composed of the basic Lattice LSTM encoder and the multi-grained Lattice LSTM encoder.

(1) Basic lattice LSTM encoder

In general, the LSTM unit [54] consists of four basic gates. Current cell state $\boldsymbol{c}_j$ keeps track of the flow of all historical information up to the present moment. The character-based LSTM functions are described as follows:

$$\begin{cases} \boldsymbol{i}_j^c = \sigma\left(W_i \boldsymbol{x}_j^c + U_i \boldsymbol{h}_{j-1}^c + \boldsymbol{b}_i\right), \\ \boldsymbol{o}_j^c = \sigma\left(W_o \boldsymbol{x}_j^c + U_o \boldsymbol{h}_{j-1}^c + \boldsymbol{b}_o\right), \\ \boldsymbol{f}_j^c = \sigma\left(W_f \boldsymbol{x}_j^c + U_f \boldsymbol{h}_{j-1}^c + \boldsymbol{b}_f\right), \\ \tilde{\boldsymbol{c}}_j^c = \tanh\left(W_c \boldsymbol{x}_j^c + U_c \boldsymbol{h}_{j-1}^c + \boldsymbol{b}_c\right). \end{cases} \tag{6}$$

$$\boldsymbol{c}_j^c = \boldsymbol{f}_j^c \odot \boldsymbol{c}_{j-1}^c + \boldsymbol{i}_j^c \odot \tilde{\boldsymbol{c}}_j^c. \tag{7}$$

$$\boldsymbol{h}_j^c = \boldsymbol{o}_j^c \odot \tanh\left(\boldsymbol{c}_j^c\right). \tag{8}$$

where $\sigma()$ represents the sigmoid function. Consequently, the state of the current cell $\boldsymbol{c}_j$ is determined by computing the weighted sum based on the past and current cell states [55].

The representation of the word $w_{b,e}$ that fits the external lexicon $\mathbb{D}$ is as follows:

$$\boldsymbol{x}_{b,e}^w = e^w\left(w_{b,e}\right). \tag{9}$$

where b and e represent the starting and ending of the word, respectively, and e^w is the lookup table. The computation of $\boldsymbol{c}_j^c$ utilises the word-level representation $\boldsymbol{x}_{b,e}^w$ to build the basic Lattice LSTM encoder. Moreover, the state of the memory cell $\boldsymbol{x}_{b,e}^w$ can be described by the word cell $\boldsymbol{c}_{b,e}^w$. The formula for computing $\boldsymbol{c}_{b,e}^w$ is:

$$\begin{cases} \boldsymbol{i}_{b,e}^w = \sigma\left(W_i \boldsymbol{x}_{b,e}^w + U_i \boldsymbol{h}_b^c + \boldsymbol{b}_i\right), \\ \boldsymbol{f}_{b,e}^w = \sigma\left(W_f \boldsymbol{x}_{b,e}^w + U_f \boldsymbol{h}_b^c + \boldsymbol{b}_f\right), \\ \tilde{\boldsymbol{c}}_{b,e}^w = \tanh\left(W_c \boldsymbol{x}_{b,e}^w + U_c \boldsymbol{h}_b^c + \boldsymbol{b}_c\right). \end{cases} \tag{10}$$

$$\boldsymbol{c}_{b,e}^w = \boldsymbol{f}_{b,e}^w \odot \boldsymbol{c}_b^c + \boldsymbol{i}_{b,e}^w \odot \tilde{\boldsymbol{c}}_{b,e}^w. \tag{11}$$

where $\boldsymbol{i}_{b,e}^w$ and $\boldsymbol{f}_{b,e}^w$ act as input and forget gates.

In order to compute the state of the *e-th* character, we must use all the words ending in index e, which is $b \in \left\{b' \mid w_{b',e} \in \mathbb{D}\right\}$. It is necessary to regulate each word's contribution, an additional $\boldsymbol{i}_{b,e}^c$ gate is utilised:

$$\boldsymbol{i}_{b,e}^c = \sigma\left(W \boldsymbol{x}_e^c + U \boldsymbol{c}_{b,e}^w + \boldsymbol{b}^l\right). \tag{12}$$

As a result, *e-th* character's cell value is calculated as follows:

$$\boldsymbol{c}_e^c = \sum_{b \in \left\{b' | w_{b',e} \in \mathbb{D}\right\}} \boldsymbol{\alpha}_{b,e}^c \odot \boldsymbol{c}_{b,e}^w + \boldsymbol{\alpha}_e^c \odot \tilde{\boldsymbol{c}}_e^c. \tag{13}$$

where $\boldsymbol{\alpha}_{b,e}^c$ and $\boldsymbol{\alpha}_e^c$ are the normalization factors, and the sum of the factors equals 1. Therefore, the vectors of the final hidden state $\boldsymbol{h}_j^c$ corresponding to each character are computed using Eq. (8).

$$\boldsymbol{\alpha}_{b,e}^c = \frac{\exp\left(\boldsymbol{i}_{b,e}^c\right)}{\exp\left(\boldsymbol{i}_e^c\right) + \sum_{b' \in \left\{b'' | w_{b'',e} \in \mathbb{D}\right\}} \exp\left(\boldsymbol{i}_{b',e}^c\right)}. \tag{14}$$

$$\boldsymbol{\alpha}_e^c = \frac{\exp\left(\boldsymbol{i}_e^c\right)}{\exp\left(\boldsymbol{i}_e^c\right) + \sum_{b' \in \left\{b'' | w_{b'',e} \in \mathbb{D}\right\}} \exp\left(\boldsymbol{i}_{b',e}^c\right)}. \tag{15}$$

(2) Multi-grained Lattice LSTM encoder

Even though the Lattice encoder can directly utilize word-level and character-level information, it is unable to completely account for ambiguity. This model is improved by including an external ambiguity language base to overcome the ambiguity problem. Consequently, a more comprehensive vocabulary is established. $\boldsymbol{x}_{b,e,k}^{sen}$ represents the word $w_{b,e}$'s *k-th* sense. All of its sense representations will be incorporated into the computation for each word $w_{b,e}$ that matches the lexicon $\mathbb{D}$. The calculation for the *k-th* sense of word $w_{b,e}$ is as follows:

$$\begin{cases} \boldsymbol{i}_{b,e,k}^{sen} = \sigma\left(W_i \boldsymbol{x}_{b,e,k}^{sen} + U_i \boldsymbol{h}_b^c + \boldsymbol{b}_i\right), \\ \boldsymbol{f}_{b,e,k}^{sen} = \sigma\left(W_f \boldsymbol{x}_{b,e,k}^{sen} + U_f \boldsymbol{h}_b^c + \boldsymbol{b}_f\right), \\ \tilde{\boldsymbol{c}}_{b,e,k}^{sen} = \tanh\left(W_c \boldsymbol{x}_{b,e,k}^{sen} + U_c \boldsymbol{h}_b^c + \boldsymbol{b}_c\right). \end{cases} \tag{16}$$

$$\boldsymbol{c}_{b,e,k}^{sen} = \boldsymbol{f}_{b,e,k}^{sen} \odot \boldsymbol{c}_b^c + \boldsymbol{i}_{b,e,k}^{sen} \odot \tilde{\boldsymbol{c}}_{b,e,k}^{sen}. \tag{17}$$

where $\boldsymbol{c}_{b,e,k}^{sen}$ refers to the memory cell associated with the *k-th* sense of the word $w_{b,e}$. In this case, all the senses are combined to compute $w_{b,e}$'s memory cell, and the result is expressed as $\boldsymbol{c}_{b,e}^{sen}$:

$$\boldsymbol{c}_{b,e}^{sen} = \sum_k \boldsymbol{\alpha}_{b,e,k}^{sen} \odot \boldsymbol{c}_{b,e,k}^{sen}. \tag{18}$$

$$\boldsymbol{\alpha}_{b,e,k}^{sen} = \frac{\exp\left(\boldsymbol{i}_{b,e,k}^{sen}\right)}{\sum_{k'}^{K} \exp\left(\boldsymbol{i}_{b,e,k'}^{sen}\right)}. \tag{19}$$

where $\boldsymbol{i}_{b,e,k}^{sen}$ represents the contribution of the *k-th* sense, and it follows in the same manner as Eq. (12).

As a consequence, all cell states will be included in the word representation $\boldsymbol{c}_{b,e}^{sen}$, which may more accurately reflect the polysemous word. Then, identical to Eqs. (12)–(15), all recurrent paths of words ending in index e will enter current cell $\boldsymbol{c}_e^c$:

$$\boldsymbol{c}_e^c = \sum_{b \in \left\{b' | w_{b',e}^d \in \mathbb{D}\right\}} \boldsymbol{\alpha}_{b,e}^{sen} \odot \boldsymbol{c}_{b,e}^{sen} + \boldsymbol{\alpha}_e^c \odot \tilde{\boldsymbol{c}}_e^c. \tag{20}$$

The hidden state $\boldsymbol{h}$ remains computed using Eq. (8) and subsequently transferred to the relation classification layer.

3.3.3 Relation Classification Layer

After learning the hidden state of an instance $\boldsymbol{h} \in \mathbb{R}^{d^h \times M}$, a character-level attention mechanism is used to integrate $\boldsymbol{h}$ into a sentence-level feature vector, represented as $\boldsymbol{h}^* \in \mathbb{R}^{d^h}$. d^h represents the hidden state dimension, and M represents the length of the sequence. The final representation $\boldsymbol{h}^*$ of a sentence is passed through a Softmax classifier, which determines the degree of confidence associated with each relation. The sentence representation $\boldsymbol{h}^*$ is computed by summing all character feature vectors in $\boldsymbol{h}$:

$$\boldsymbol{H} = \tanh(\boldsymbol{h}). \tag{21}$$

$$\boldsymbol{\alpha} = \text{softmax}\left(\boldsymbol{w}^T \boldsymbol{H}\right). \tag{22}$$

$$\boldsymbol{h}^* = \boldsymbol{h}\boldsymbol{\alpha}^T. \tag{23}$$

where $\boldsymbol{w} \in \mathbb{R}^{d^h}$ is the parameter that has been trained, and $\boldsymbol{\alpha} \in \mathbb{R}^M$ represents the weight vector associated with $\boldsymbol{h}$.

The conditional probability of each relation is computed by feeding the vector of the feature $\boldsymbol{h}^*$ of the sentence S into a Softmax classifier as follows:

$$\boldsymbol{o} = \boldsymbol{W}\boldsymbol{h}^* + \boldsymbol{b}. \tag{24}$$

$$p(\boldsymbol{y} \mid S) = \text{softmax}(\boldsymbol{o}). \tag{25}$$

where $\boldsymbol{W} \in \mathbb{R}^{Y \times d^h}$ represents the transformation matrix and $\boldsymbol{b} \in \mathbb{R}^Y$ is the bias vector. Y represents all of the relation types, whereas $\boldsymbol{y}$ is the estimated probability of each type.

Finally, given all (T) training examples $(S^{(i)}, y^{(i)})$, cross-entropy is used to define the objective function:

$$J(\theta) = \sum_{i=1}^{T} \log p\left(y^{(i)} \mid S^{(i)}, \theta\right). \tag{26}$$

where θ is used to represent all parameters in the model. The dropout method [56] is utilised in the LSTM layer through a process of randomly eliminating feature detectors during the propagation of the forwarding to prevent co-adaptation of hidden units.

4 Case Study

To illustrate the feasibility and efficiency of the knowledge graph-driven design knowledge acquisition framework and the proposed design knowledge extraction model, a case study was conducted to acquire creative cultural design knowledge. The knowledge graph was built to assist designers in accessing valuable intangible cultural heritage and to benefit local manufacturing businesses through innovative cultural products. The critical steps of the case study are depicted in Fig. 6.

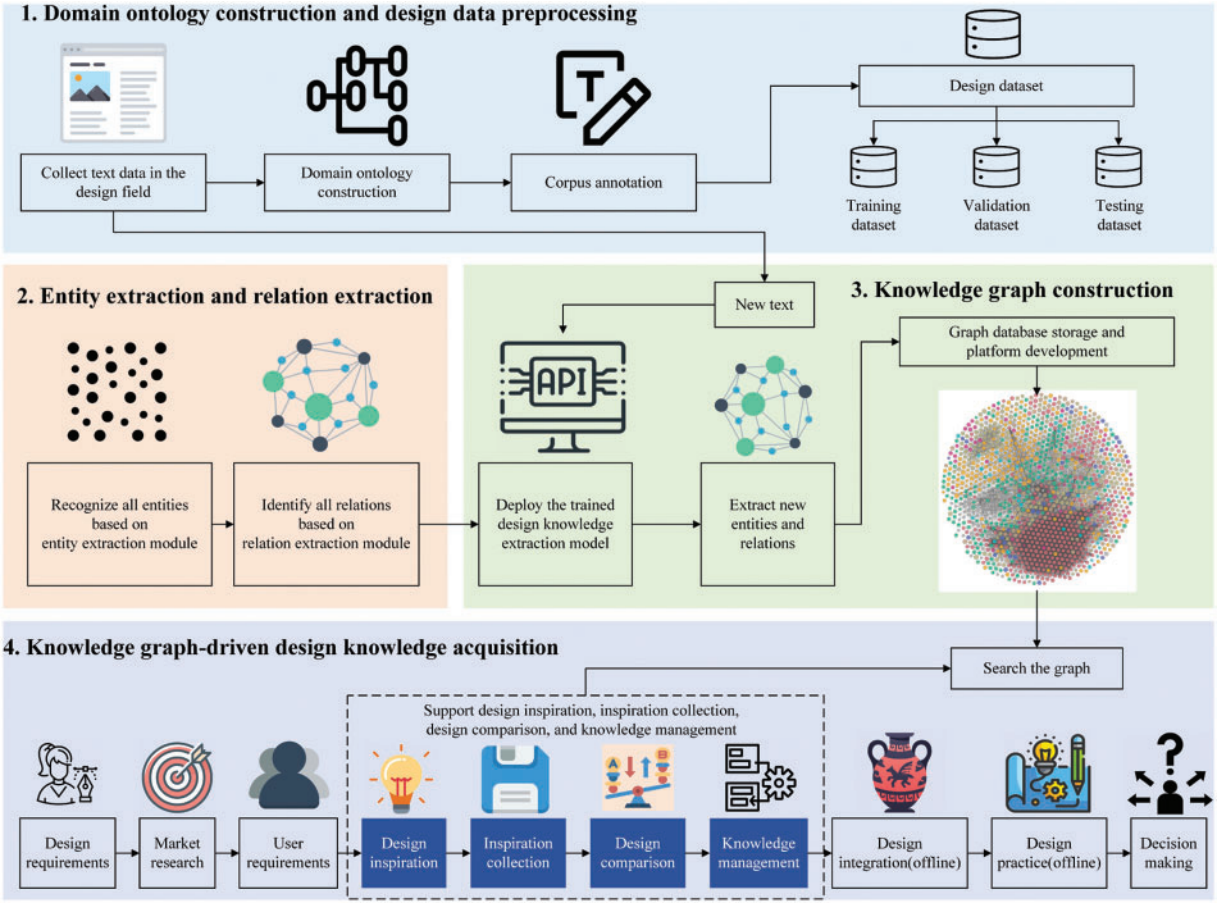

Figure 6: The key steps of the case study

4.1 Domain Ontology Construction and Design Data Preprocessing

This study employs a domain ontology to outline the structure of the knowledge graph to establish a structured and unified knowledge graph for the acquisition of creative cultural design knowledge. Data were collected from the Chinese Palace Museum website (www.dpm.org.cn/collection/ceramics.html), including 1151 porcelains, 4658 pictures, and 639,676 words. The domain ontology was then meticulously defined with the help of product designers, intangible cultural experts, and programmers. The methodology used in creating ontologies may differ. In this study, a systematic approach was followed, comprised of the steps outlined in the ontology development guideline [57]. A summary of these steps is presented in Fig. 7 and will be expounded upon in the subsequent sections of this paper.

(1) Define domain and scope. In this study, ontology development is centred around creative cultural design data, specifically porcelain data, while also considering the integration of the conceptual product design domain. Therefore, the domain scope has been expanded to include the conceptual product design.

(2) Reuse existing ontologies. As previously discussed, the scope of the proposed ontology model encompasses two domains: conceptual product design and cultural heritage. Within the domain of conceptual product design, several ontologies have been established, such as the Creative Cultural Design ontology (CCD) [58]. This ontology provides a structured and unified approach for modelling, representing, querying, and accessing Chinese creative cultural

design, which becomes a suitable ontology for reference. CCD consists of five main categories and 22 sub-categories, as illustrated in Fig. 8a. These categories include basic information, styling, functional, technical, and cultural knowledge. The basic information category contains essential details about the object, such as its classification, name, date of creation, location of origin, creator, dimensions, and history of exhibitions. The styling knowledge indicates shape, ornamentation, and colour. The functional knowledge category plays a crucial role in shaping the design, from the practical aspects of usage scenarios and purpose, to the symbolic meaning and aesthetics of the object. The technical knowledge category represents the materials and techniques used to create the object. Finally, the cultural knowledge category represents the object's cultural significance, historical context, cultural value, and related folk tales. On the other hand, the cultural heritage domain has several established ontologies, with the CIDOC Conceptual Reference Model (CIDOC CRM) [59] being the most widely accepted, as depicted in Fig. 8b. With its long and rigorous development process, CIDOC CRM has been successfully implemented in various projects and has become the only ISO standard ontology in cultural heritage fields. This study uses concepts from CIDOC CRM to develop our ontology model. At the time of writing, CIDOC CRM is in version 6.2.310 and comprises 99 classes and 188 properties, which may be too extensive for our purposes. Therefore, we select some to incorporate into our ontology structure. Therefore, this study leverages the existing and well-established ontology frameworks in both the creative cultural design and cultural heritage domains. Specifically, CCD [58] is reused from the domain of conceptual product design, while CIDOC CRM [59] is reused from the cultural heritage domain. Both frameworks have a proven track record of being standards, validated and utilised in numerous projects, having matured over time.

(3) Enumerate important terms. Identifying important terms in this study involves identifying key terms related to conceptual product design and cultural heritage. The data relating to Chinese porcelain is limited, and no central portal offers a significant amount of porcelain information, mostly kept privately by heritage organisations. There is a ray of hope in the form of an online database for the Chinese Palace Museum, which includes a list of objects with images and textual descriptions. However, the metadata for these descriptions is not standardised or well-structured. Given these limitations, this study manually selects the classes and properties required for modelling the data using CCD and CIDOC CRM ontologies. Manual data discovery is a time-consuming and labour-intensive process with its limitations. It was the only option since there was no standardised structured database.

(4) Define the concepts. The domain ontology was created by a multi-disciplinary team based on CCD and CIDOC CRM. Seven concepts were identified: name, dynasty, pattern, shape, colour, glaze, and function, as depicted in Fig. 8c. These concepts align with the principles of independence, sharing, and minimization [60]. The entity 'E1 Name' serves as a superclass in the CIDOC CRM and acts as a domain for other properties. 'E2 Dynasty' is used to define the temporal extent of the validity of instances of 'Name' and its subclasses. 'E7 Function' describes the scenario in which the object is used and its methods and forms of use. The other entities, 'E3 Pattern', 'E4 Shape', 'E5 Color', and 'E6 Glaze', originate from the CCD ontology for conceptual product design. 'E3 Pattern' refers to the lines, designs, and other decorative elements present on the object. 'E4 Shape' describes the physical appearance of the object. 'E5 Color' indicates the colour composition of the object. 'E6 Glaze' is a vitreous coating fused to the pottery body through firing and serves as the classification of porcelain.

(5) Define the properties of concepts. The properties of the concepts and the facets of the slots play a crucial role in the design of the data model. The multi-disciplinary team chose to incorporate the inverse property feature of CIDOC CRM ontology to provide flexibility in the data model

design. The object-centric nature of the model is highlighted by the fact that the object is at the centre, and other concepts are connected to it. Linking the object's name to the CCD concepts emphasises the product design aspect of the object. The relationships between the concepts were established using a top-down approach. 97 relationships were defined based on the design requirements and cultural classification.

(6) Create instances. After the ontological data model was designed and constructed, it was filled with data. Information was gathered from the official website of the Chinese Palace Museum, which was organised into specific classes, linked to other classes, and formed the knowledge graph. The concepts are represented by 'E' in the ontology diagram, while their relationships are represented by 'P'. Fig. 9 illustrates a part of the entities and relations. For example, (Dynasty, P3 has pattern of, Pattern) represents the correlation between 'dynasty' and 'pattern' (Glaze, P2 has color of, Color) indicates the connection between 'glaze' and 'colour'. This provides a visual representation of the relationships between the various concepts in the ontology, allowing for a better understanding of the data model.

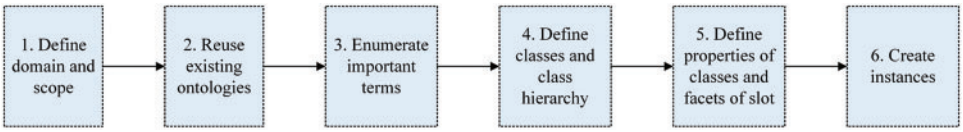

Figure 7: The process of domain ontology construction

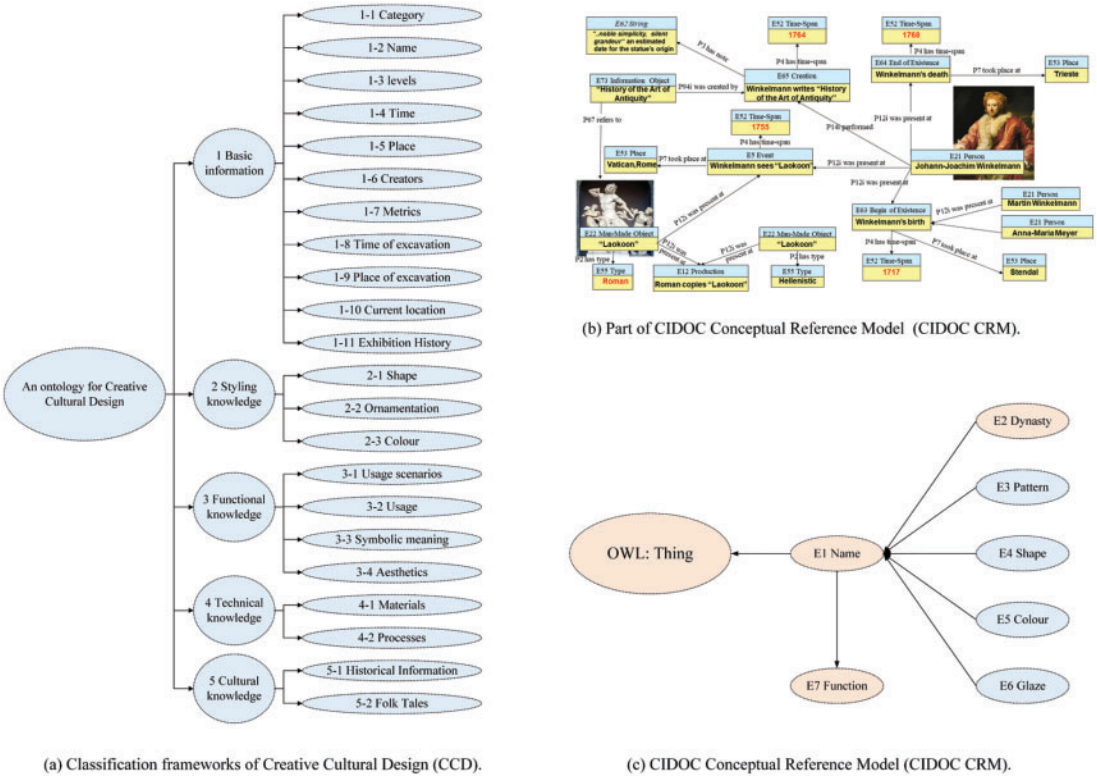

Figure 8: Reuse existing ontologies and define the concepts

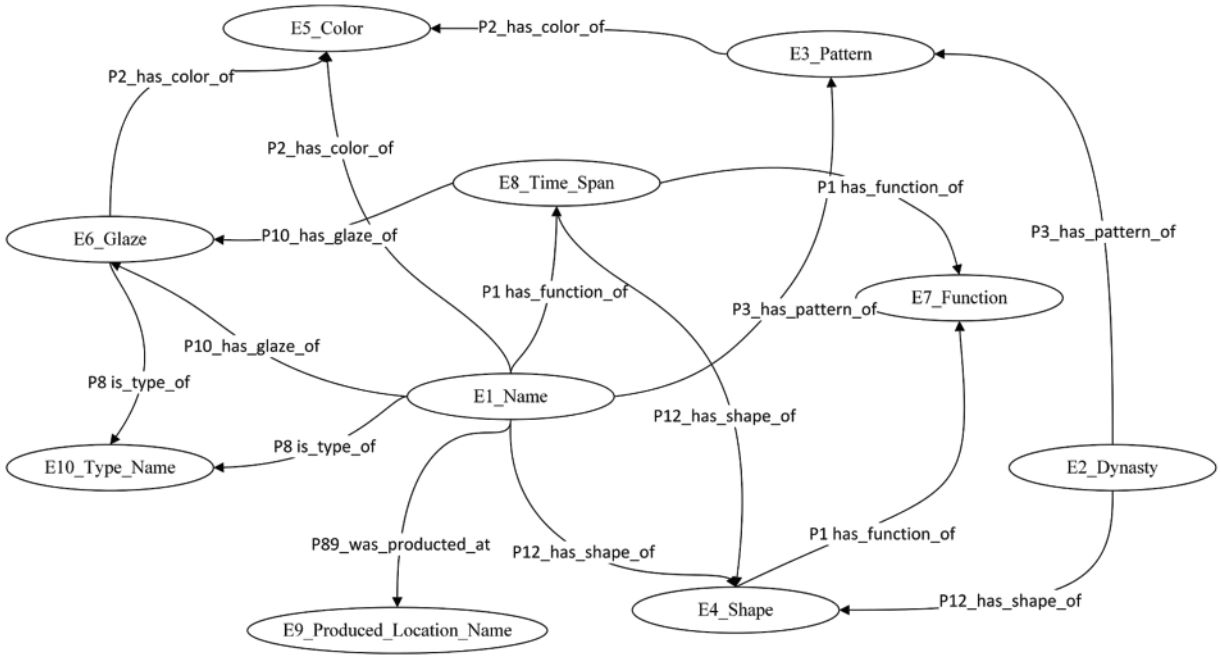

Figure 9: Instances of domain ontology

Following the domain ontology construction, the corpus is annotated, and a design dataset is created for design knowledge acquisition. The crawled text data were manually labelled using the Brat programme according to the domain ontology, as shown in Fig. A1 in Appendix A. The annotated corpus was transformed into dataset format. In the design dataset, there are 639,676 entities and 8,689 relations, which are divided into the training set, the validation set, and the testing set in the proportion of 8:1:1. After constructing the design dataset, F-values [61] is typically applied to calculate the labelling consistency of the dataset [62]. The annotation is considered consistent only if the annotated text, type label, and starting and ending positions are identical. Artstein et al. [63] suggested that 80% or more of consistency is considered acceptable. In this study, 100 sentences were selected randomly for calculation. The annotated results of A1 were used as the standard label, and the annotation consistency was calculated according to Eq. (27). The entity consistency is 96.2%, and the relation consistency is 93.5%, demonstrating the reliability of our constructed dataset. Additionally, the quantity of entities and relations has a longtail effect which may influence the results. Table 1 shows the statistics of the dataset.

$$P = \frac{\text{Number of consistent annotation results from } A_1 \text{ and } A_2}{\text{Number of annotations from } A_2},$$
$$R = \frac{\text{Number of consistent annotation results from } A_1 \text{ and } A_2}{\text{Number of annotations from } A_1}, \tag{27}$$
$$F = \frac{2 \times P \times R}{P+R}.$$

Table 1: Statistics of the dataset

Category	Design dataset
Train	7017
Valid	838
Test	834
Overall	**8689**

4.2 Experimental Results of Design Knowledge Extraction

To verify the proposed entity extraction module and the relation extraction module, the proposed models are validated on the design dataset separately through comparative experiments. This study uses Python 3.6, CUDA 10.1, PyTorch 2.7, Neo4j, and SQL Server. The operating systems are Windows 10 and Ubuntu 16.04 with the i7 7700K CPU and the 1080Ti 11G GPU.

4.2.1 Entity Extraction Experiment

To conduct the entity extraction experiment, the design dataset was converted into a BIO sequence annotation, as shown in Fig. A2 in Appendix A. The ten-fold cross-validation was applied to the entity extraction model training process. The hyperparameters are displayed in Table 2. The early stopping method was used to prevent over-fitting. The evaluation metrics did not improve in 5 rounds after 40 iterations. The loss value is significant at the beginning of the experiment, but it converges to zero in the 26 rounds, demonstrating the robustness of the model. The entity extraction model's accuracy, recall and F1-score are 94.2%, 96.7% and 95.4%.

Table 2: Parameters of entity extraction model

Parameter	Rate
Layer	12
Hidden layer	768
Parameter sharing	Yes
Maximum sequence length	128
Batch size	32
Learning rate	2e-5
Epoch	50
BiLSTM hidden layer	128

The proposed entity extraction model was compared with some well-known entity extraction models on the design dataset to verify the performance. (1) BiLSTM-CRF, the structure is BiLSTM encode layer and CRF entity classification layer. (2) BERT-BiLSTM-CRF, the structure is BERT pre-training layer, BiLSTM encode layer and CRF entity classification layer. This study uses BERT-base, which has 108M parameters, 12 layers, 768 hidden units, 768 embeddings and parameter sharing. (3) ALBERT-BiLSTM-CRF, the structure is ALBERT pre-training layer, BiLSTM encode layer and CRF

entity classification layer. This study uses ALBERT-base, which has 12M parameters, 12 layers, 768 hidden units, 128 embeddings and parameter sharing.

The result of the comparative experiment of entity extraction is shown in Fig. 10. The training process of different entity extraction models is shown in Fig. 11. The results are as follows: (1) The BiLSTM-CRF does not contain the pre-training layer, and its performance is the worst of all three comparative models. The precision is 73.8%, the recall is 77.4%, and F1-score is 75.6%. (2) BERT-BiLSTM-CRF utilises the prior knowledge in the BERT pre-training layer and brings about a 25.5% enhancement in F1-score over BiLSTM-CRF. (3) The proposed entity extraction model ALBERT-BiLSTM-CRF outperforms the baseline BiLSTM-CRF and achieves 26.2% improvement in F1-score over BiLSTM-CRF. With much fewer parameters and a quicker training speed, ALBERT-BiLSTM-CRF achieves similar results (94.2%, 96.7%, and 95.4%) as large pre-training model BERT (93.3%, 96.5%, and 94.9%).

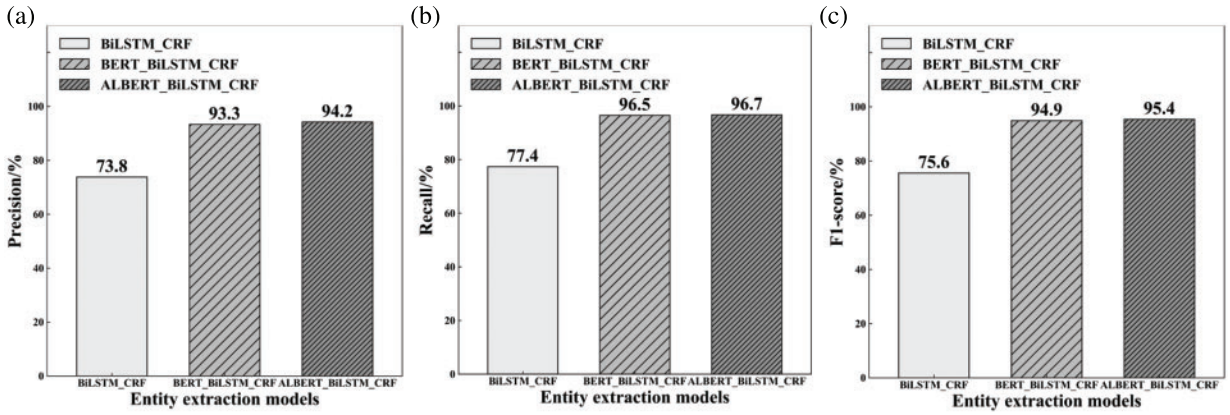

Figure 10: Comparison of entity extraction models

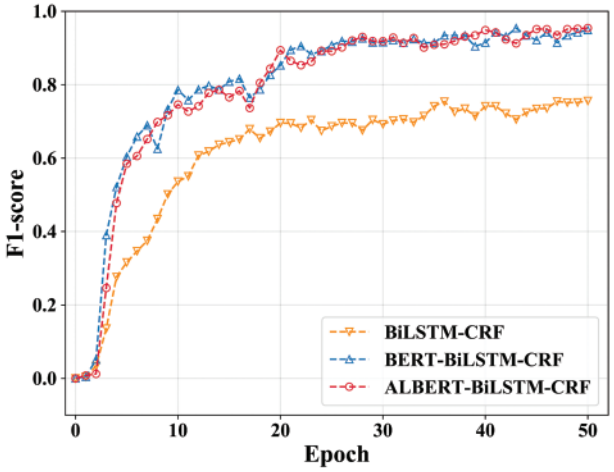

Figure 11: The training process of entity extraction models

4.2.2 Relation Extraction Experiment

In order to conduct the relation extraction experiment, the design dataset was converted into a relation ID, a training set, a validation set and a testing set, as shown in Fig. A3 in Appendix A. The parameters of the relation extraction model were set in the training process (in Table 3), and the early stopping method was adopted for the overfitting problem. The model was trained using ten-fold cross-validation and achieved the best performance after about 30 epochs. The entity extraction model's accuracy, recall and F1-score are 68.5%, 70.9% and 69.7%.

Table 3: Parameters of relation extraction model

Parameter	Rate
Learning rate	5e-4
Dropout	0.5
Char embedding dimension	100
Lattice embedding dimension	200
Position embedding dimension	5
LSTM hidden layer	200
Regularization	1e-8

The proposed relation extraction model was compared with some well-known relation extraction models on the design dataset to verify the advancement: (1) Att-BiLSTM utilises word-based embedding for the input content. The structure of Att-BiLSTM is an input representation layer, BiLSTM encode layer and relation classification layer. (2) Lattice uses word-based and character-based embedding for input representation. The structure of the Lattice is an input representation layer, lattice LSTM encode layer and relation classification layer. (3) Our proposed relation extraction model Multi-Grained-Lattice, takes the word-based, character-based and external language into account for the input content. The structure of Multi-Grained-Lattice is an input representation layer, multi-grained Lattice encode layer and relation classification layer.

The result of the comparative experiment of relation extraction is shown in Fig. 12. The training process for different relation extraction models is illustrated in Fig. 13. The results are as follows: (1) Att-BiLSTM model uses only word-based inputs, which performs worse than the other two models. The precision is 55.7%, recall is 53.1%, and F1-score is 54.4%. (2) Lattice uses both word-based and character-based input and improves about 16.5% in F1-score compared to Att-BiLSTM. The results of Lattice are 65.7%, 61.2% and 63.4%. (3) The proposed relation extraction model (Multi-Grained-Lattice) utilises the word-based, character-based and external language for the input content and performs the best among the three models. It achieves 28.1% and 10.0% improvements of the F1-score over Att-BiLSTM and Lattice, respectively. The results of Multi-Grained-Lattice are 68.5%, 70.9% and 69.7%, respectively.

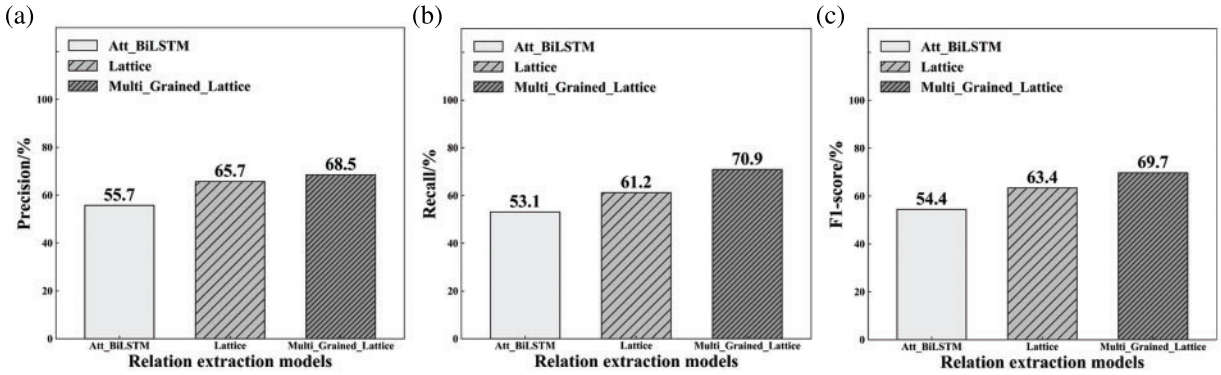

Figure 12: Comparison of relation extraction models

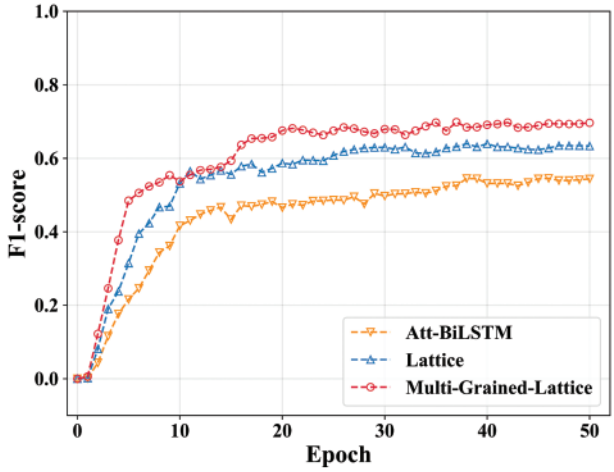

Figure 13: The training process of relation extraction models

4.3 Knowledge Graph Construction

To extract design-related entities and relations automatically, the trained model was deployed as a web application using Flask, Gunicorn and Nginx. The new text was entered into the trained model by calling the API, and the design-related entities and relations were extracted. For example, the words 'blue and white porcelain' and 'cloud and dragon pattern' are automatically extracted by entering 'blue and white porcelain vase with a cloud and dragon pattern, round foot and entwined branches', as shown in Fig. 14a. Furthermore, the relations of the entity pairs are extracted using the deployed relation extraction model, such as 'has_glaze_of', 'has_pattern_of' and 'is_shape_of', as displayed in Fig. 14b. Consequently, the entity extraction model and relation extraction model can extract entities and relations from text, allowing the automatic establishment of the knowledge graph. In this study, 15,316 entities and their relations were extracted using the proposed models and stored in the Neo4j graph database. The knowledge graph was developed with the extracted entities and relations to acquire design knowledge.

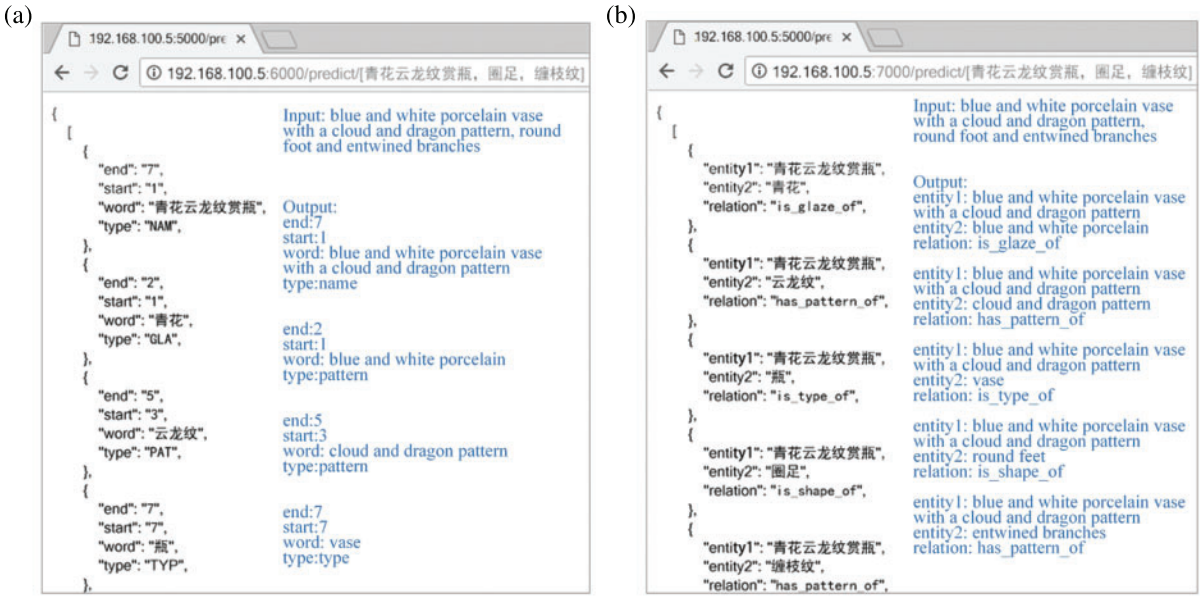

Figure 14: Model deployment and testing. (a) Testing result of entity extraction module. (b) Testing result of relation extraction module

The functional architecture of the knowledge graph is displayed in Fig. 15a, which realises knowledge graph construction, updating, management and application. The functional architecture facilitates the management of models, tools, users and data. With its comprehensive and scalable construction capabilities, the knowledge graph platform can perform many functions, such as extraction, construction and application.

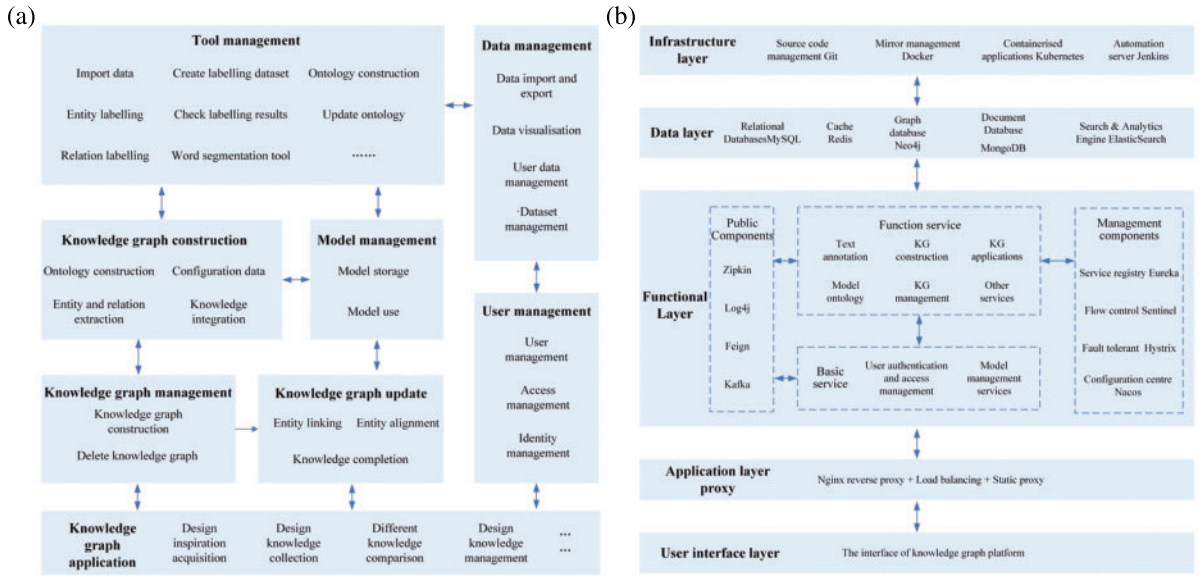

Figure 15: The architecture of the knowledge graph platform. (a) The functional architecture of the knowledge graph platform. (b) The technical architecture of the knowledge graph platform

The technical architecture of the knowledge graph is displayed in Fig. 15b, which is divided into the infrastructure layer, data layer, functional layer, application layer proxy and user interface layer. The infrastructure layer is the most basic layer and includes source code management Git, mirror management Docker, containerised applications Kubernetes and automation server Jenkins. Moreover, the data layer uses the relational database MySQL, cache database Redis, graph database Neo4j, document database MongoDB, and data search engine Elastic Search for data storage and search services. After that, the functional layer supports the construction, updating and application of the knowledge graph. The functional layer also provides the corresponding public components and management components. In addition, the application layer proxy uses the Spring cloud gateway to accept and forward external requests. It uses Nginx to implement reverse proxying, load balancing and static resource storage. Finally, the user interface layer provides designers with the knowledge graph platform and supports design knowledge acquisition for conceptual product design.

4.4 Knowledge Graph-Driven Design Knowledge Acquisition

Conceptual product design includes design goals, design analysis, and design implementation. Specifically, the process involves market research, user requirements, design inspiration, inspiration collection, design comparison, knowledge management, design integration, design practice and decision making [64], as shown in Fig. 16. In order to verify the effectiveness of the proposed knowledge graph for design knowledge acquisition, the knowledge graph platform was tested under the conceptual product design process.

(1) Market research and user requirement. As depicted in Fig. 16a, the design knowledge platform (DKG) provides a project module for designers to create a new project and fill in important details such as market goals, design requirements, and project description. For instance, designers aim to create a series of products featuring traditional dragon patterns. They would create a new project on the knowledge graph platform and fill in the project name, ‘Series of Products in Traditional Dragon Style’, the market goal, ‘modern style products with dragon patterns’, and the design requirement, ‘dragon pattern’. This project module facilitates collaboration among designers, enabling them to create projects and work together on the knowledge graph easily.

(2) Design inspiration. As shown in Fig. 16b, the knowledge graph platform searches for design inspiration in the design goal stage. It works by first identifying design-related entities and relationships from input sentences using the design knowledge extraction model. The knowledge graph platform then defines the search scope and matches the related design knowledge with a text-matching model. The result is the visualisation of relevant design knowledge and inspiration for designers. For example, if a designer inputs a query such as ‘design of bottles with a dragon pattern’, the knowledge graph platform will identify the entities ‘dragon’ and ‘bottle’ and match them with related design knowledge. The result will be visualised information and images about associated design cases, such as ‘Blue and white bottle with dragon pattern’ and ‘Cloisonne and Faience bottle with branches and dragon pattern’. The advantage of the knowledge graph platform compared to conventional design knowledge acquisition methods, such as XML and OWL, is that it provides design inspiration in a more relational, visual, and intuitive way.

(3) Inspiration collection. Fig. 16c illustrates the ability of the knowledge graph platform to search, edit, and save detailed information on design cases. For instance, the designer selects the node ‘Yaozhou celadon glaze carved lotus pattern amphora’. This triggers the search and inference operations on the node, and the node is mapped to the knowledge graph platform.

As a result, relevant nodes and design details such as 'blue and white glaze' and 'lotus petal pattern' are displayed. The editing and saving functions are performed by clicking the 'Edit' and 'Save' buttons. The utilization of a knowledge graph platform in design knowledge acquisition presents a significant improvement over traditional methods such as XML and OWL. This platform offers a more accurate and relevant acquisition of design knowledge, while also incorporating the added benefit of reasoning capabilities. The results are presented visually and have graphical interaction, providing a more intuitive and comprehensive presentation compared to traditional text-based methods.

(4) Design knowledge comparison and management. In Fig. 16d, the knowledge graph platform compares and manages design knowledge by exploring similarities and differences. A designer can select multiple design cases for comparison, and the knowledge graph platform will present the related design knowledge in a graphical format. For example, a designer may choose five porcelain bottles, and the knowledge graph platform will display that four bottles have an 'enamel' glaze and 2 of them were made using the 'rolling process'. The advantage of the knowledge graph platform over traditional design comparison and management methods is that it presents the comparative information in a more correlated and visual format.

(5) Design integration. The design pattern in Fig. 17a takes inspiration from enamel colours used in traditional Chinese porcelain and presents it in a simplified, refined and transformed form to bring a new dimension to it. The new design pattern embodies the essence of traditional Chinese porcelain enamel colour, taking inspiration from its unique aesthetic qualities. The design process involved refining, simplifying, and transforming the traditional pattern to create a modern, updated version. The result is a fresh and contemporary take on a classic design style, offering a visually appealing blend of tradition and modernity. This design pattern not only showcases the rich cultural heritage of traditional Chinese porcelain but also demonstrates the versatility and timelessness of its aesthetic principles.

(6) Conceptual design practice. As shown in Fig. 17b, the integration of the traditional pattern into contemporary products serves to reinvigorate and give new life to the timeless design style. By bringing the pattern into the present day, a broader audience can be exposed to and appreciate the beauty of traditional coloured porcelain. This not only celebrates the rich cultural heritage but also highlights the versatility and relevance of the traditional design style in modern times. By bridging the gap between the past and the present, the traditional pattern can be appreciated and celebrated in new and innovative ways, ensuring that it continues to be an integral part of our cultural heritage for generations to come.

(7) Decision making. As shown in Fig. 17c, the knowledge graph platform provides designers with a visualised comparison of the pros and cons of each design scheme, making it easier for designers to make informed decisions. The overall score is based on the designers' comments, feedback, and scores and is presented in a graphical format to assist in the final decision-making process. Using the knowledge graph platform in the decision-making stage allows for more efficient and effective collaboration between designers and helps ensure that the final design solution is of high quality and meets the design requirements.

The knowledge graph prototype platform has the potential to change the way designers approach the conceptual product design process. With its ability to extract design-related entities and relationships, match relevant knowledge, and provide visualised and correlated design cases, the knowledge graph platform can streamline the design knowledge acquisition process and improve the overall efficiency of the conceptual product design process. The results of the testing indicate a favorable

response from all participants, demonstrating the potential for the system to significantly enhance the efficiency of large-scale product design through widespread adoption. The implementation of the knowledge graph platform in other design cases can serve as a further reference for designers and potentially improve the overall efficiency of the design industry.

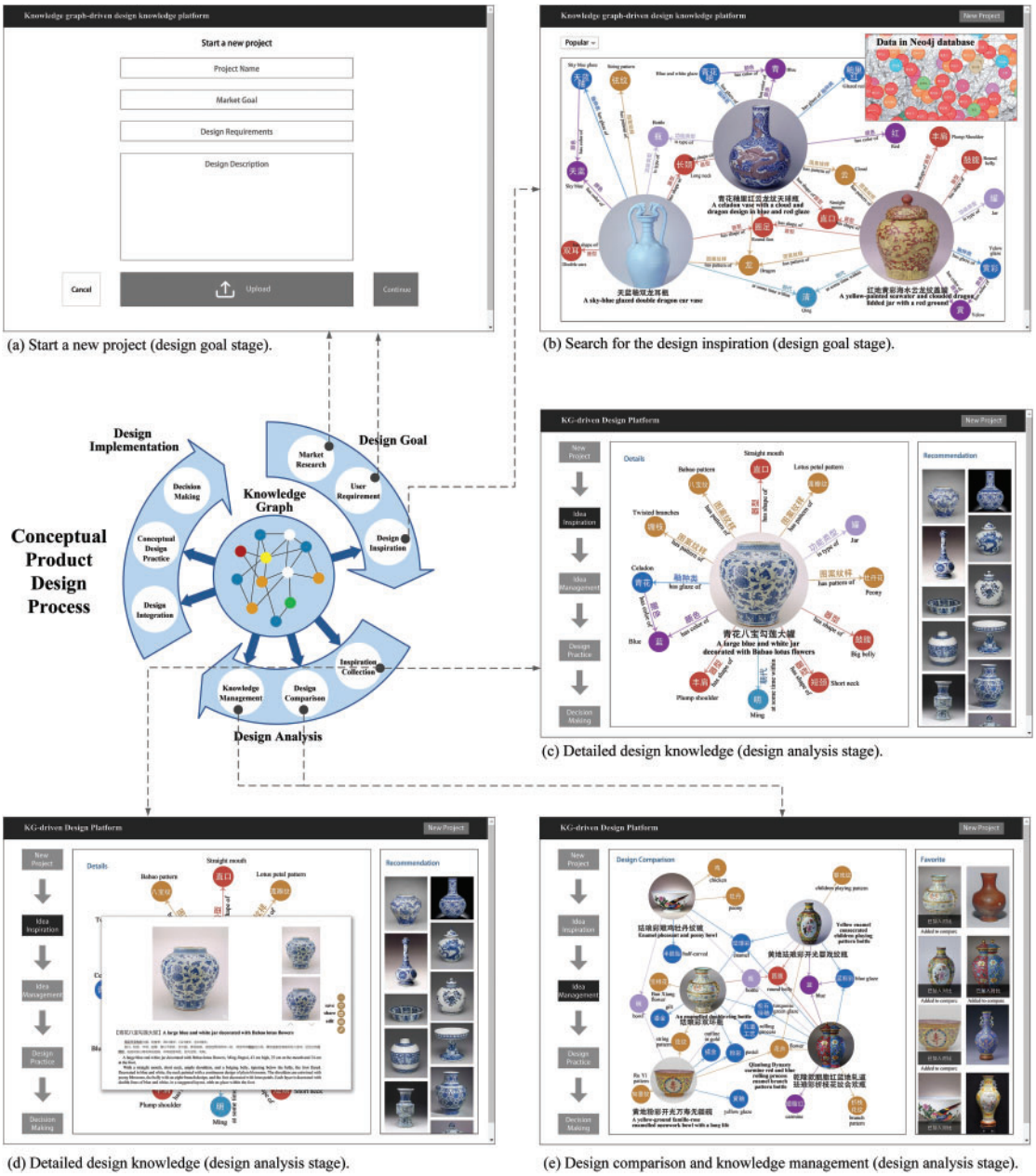

Figure 16: The illustration of knowledge graph-driven design knowledge acquisition (translation)

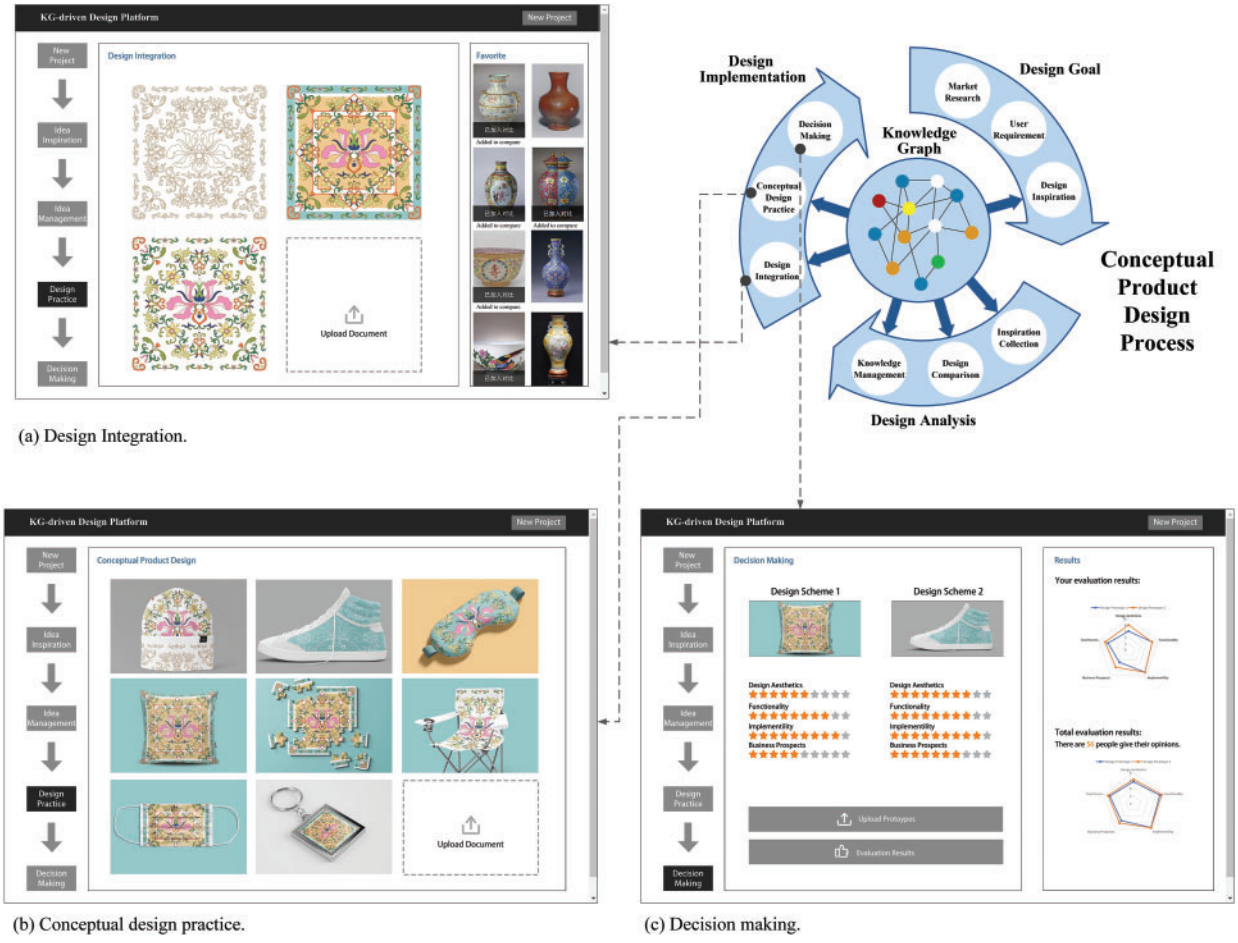

Figure 17: The illustration of knowledge graph-driven design knowledge acquisition (translation)

5 Discussion

This section discusses the design knowledge extraction model, application and limitations to analyse the proposed methods.

5.1 Design Knowledge Extraction Model

As for the entity extraction module, the F1-score of the BiLSTM-CRF model is approximately 25% lower than the other two models, which proves the effectiveness of good performance of pre-training layers on entity extraction tasks. In addition, BERT and ALBERT have similarities in the training process and entity extraction results. Due to the fact that ALBERT presents two parameter-reduction strategies to address the memory limitation and the communication overhead. As a result, the model can achieve effective entity extraction performance with fewer parameters by introducing the ALBERT pre-training layer. In conclusion, the comparative experiments demonstrate the high accuracy of ALBERT-BiLSTM-CRF.

Regarding the relation extraction module, three comparison models have a similar encode layer and the relation classification layer. The input representations lead to different results. The Att-BiLSTM uses character-based input representation, so the result of Att-BiLSTM is the worst among the three models. The Lattice utilises character-based and word-based input representation,

so its F1-score is approximately 16% better than Att-BiLSTM. Our proposed model Multi-Grained-Lattice has the fastest training speed, and its F1-score tends to stabilise at 70% after 18 epochs. The proposed model fully leverages the semantic information, including character-based, word-based and external knowledge-based, to enhance the capability of relation extraction. It solves the problems of segmentation errors and polysemy ambiguity in relation extraction. Therefore, our proposed model shows the capability of multi-granularity language, including characters, words, and external knowledge for relation extraction, and it outperforms the other two models.

5.2 Applications and Limitations

To demonstrate the innovation of knowledge graph application in design knowledge acquisition, we compared the knowledge graph-based method with the conventional design knowledge acquisition methods, such as XML-based [4] and OWL-based [6] methods. The XML-based design knowledge acquisition method describes the structural information between design knowledge, while it is insufficient in the semantic description, reasoning and visualisation. The OWL-based design knowledge acquisition methods perform better in reasoning than XML-based methods. The knowledge graph-based methods are better than XML and OWL in the semantic description, reasoning and visualisation, which proves the advancement of the proposed design knowledge acquisition framework. Furthermore, the case study demonstrates that the proposed design knowledge extraction model can effectively extract design-related entities and relations from unstructured text, gathering massive fragmentary data automatically. The proposed knowledge graph facilitates idea generation, design knowledge management and knowledge comparison stages in conceptual product design.

The proposed knowledge graph construction method and design knowledge extraction model can be applied to design-related websites, such as Dribble, Huaban and Behance, to help designers acquire valuable and visualised design knowledge. Moreover, the proposed knowledge graph method can also become a module of the enterprise collaborative design platform to facilitate the conceptual product design. The knowledge graph could also be used in other fields, such as intangible cultural heritage, complex product models and industry transformation.

It is noticed that the knowledge graph construction requires large amounts of design-related data, such as design inspiration, historical product data, and domain expert data. Due to the length of the manuscript, this study only validated historical product data. Therefore, the effectiveness of the knowledge graph, including design inspiration and domain expert data, still need to be verified in the future. The relation extraction task requires further improvement due to error accumulation, polysemous words, and overlapping relations. Furthermore, the process of collecting and labelling data is laborious, which makes the construction of knowledge graph challenging. Lastly, the current knowledge graph for design knowledge acquisition lacks the capability of requirement analysis and knowledge recommendation. Designers have to manually search for design knowledge, which can be a time-consuming and inefficient process.

6 Conclusion

Despite the importance of design knowledge for conceptual product design, designers have difficulty acquiring valuable design knowledge from large amounts of fragmentary data. This study proposes a design knowledge acquisition method combining deep learning with a knowledge graph. Specifically, the knowledge graph-driven design knowledge acquisition framework is proposed to acquire visualised and interconnected design knowledge. Secondly, the design knowledge extraction model is presented to extract entities and relations for knowledge graph construction automatically.

The design knowledge extraction model introduces ALBERT in the entity extraction module and utilises multi-granularity information in the relation extraction module. It solves the problems of memory limitation, long training time and multi-granularity in the design knowledge extraction. Comparison experiments show that the proposed entity extraction model ALBERT-BiLSTM-CRF outperforms BiLSTM-CRF by 25% and performs similarly to BERT with fewer parameters in the pre-training layer. The proposed relation extraction model Multi-Grained-Lattice represents 28.1% and 10.0% improvements over Att-BiLSTM and Lattice model. Lastly, the case study demonstrated that designers can access visualised and interconnected creative cultural design knowledge. The proposed method improves the acquisition of large amounts of fragmentary design knowledge by combining deep learning with the knowledge graph. The practical value is that the knowledge graph-driven design knowledge acquisition method can be applied to collaborative design websites to help designers acquire valuable visualised design knowledge, and further facilitate the conceptual product design.

Future works will explore multi-modal design knowledge, such as images, videos and audio. In order to improve the accuracy of relation extraction tasks, error accumulation, polysemous words, and overlapping relations should be considered. Knowledge reasoning and recommendation will also be included in the knowledge graph construction for design knowledge acquisition.

Acknowledgement: The authors wish to express sincere appreciation to the reviewers for their valuable comments, which significantly improved this paper.

Funding Statement: This research is supported by the Chinese Special Projects of the National Key Research and Development Plan (2019YFB1405702).

Conflicts of Interest: The authors declare that they have no conflicts of interest to report regarding the present study.

References

1. Vila, C., Albiñana, J. C. (2016). An approach to conceptual and embodiment design within a new product development lifecycle framework. *International Journal of Production Research, 54(10),* 2856–2874. https://doi.org/10.1080/00207543.2015.1110632
2. Kataoka, H., Satoh, Y., Abe, K., Minoguchi, M., Nakamura, A. (2019). Ten-million-order human database for world-wide fashion culture analysis. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pp. 4321–4328. Long Beach.
3. Chinese National Academy of Arts (2018). China national academy of arts intangible cultural heritage database. https://www.ihchina.cn/tuji_list.html
4. Bloomfield, R., Mazhari, E., Hawkins, J., Son, Y. J. (2012). Interoperability of manufacturing applications using the core manufacturing simulation data (CMSD) standard information model. *Computers & Industrial Engineering, 62(4),* 1065–1079. https://doi.org/10.1016/j.cie.2011.12.034
5. Wu, Z., Liao, J., Song, W., Mao, H., Huang, Z. et al. (2018). Semantic hyper-graph-based knowledge representation architecture for complex product development. *Computers in Industry, 100(2),* 43–56. https://doi.org/10.1016/j.compind.2018.04.008
6. Hu, H., Liu, Y., Lu, W. F., Guo, X. (2022). A knowledge-based approach toward representation and archiving of aesthetic information for product conceptual design. *Journal of Computing and Information Science in Engineering, 22(4),* 041011. https://doi.org/10.1115/1.4053674
7. Bi, Z., Wang, S., Chen, Y., Li, Y., Kim, J. Y. (2021). A knowledge-enhanced dialogue model based on multi-hop information with graph attention. *Computer Modeling in Engineering & Sciences, 128(2),* 403–426. https://doi.org/10.32604/cmes.2021.016729

8. Liang, C., Wu, Z., Huang, W., Giles, C. L. (2015). Measuring prerequisite relations among concepts. *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pp. 1668–1674. Lisbon.
9. Lai, B., Zhao, W., Yu, Z., Guo, X., Zhang, K. (2022). A multi-domain knowledge transfer method for conceptual design combine with FBS and knowledge graph. *International Design Engineering Technical Conferences and Computers and Information in Engineering Conference*, St. Louis, Missouri, USA.
10. Bharadwaj, A. G., Starly, B. (2022). Knowledge graph construction for product designs from large CAD model repositories. *Advanced Engineering Informatics, 53(2),* 101680.
11. Lyu, M., Li, X., Chen, C. H. (2022). Achieving knowledge-as-a-service in IIoT-driven smart manufacturing: A crowdsourcing-based continuous enrichment method for industrial knowledge graph. *Advanced Engineering Informatics, 51(11),* 101494. https://doi.org/10.1016/j.aei.2021.101494
12. Zhao, X., Liu, Y., Xu, Y., Yang, Y., Luo, X. et al. (2022). Heterogeneous star graph attention network for product attributes prediction. *Advanced Engineering Informatics, 51(23),* 101447. https://doi.org/10.1016/j.aei.2021.101447
13. Li, X., Chen, C. H., Zheng, P., Wang, Z., Jiang, Z. et al. (2020). A knowledge graph-aided concept–knowledge approach for evolutionary smart product–service system development. *Journal of Mechanical Design, 142(10),* 101403. https://doi.org/10.1115/1.4046807
14. Huang, Y., Yu, S., Chu, J., Su, Z., Zhu, Y. et al. (2023). Design knowledge graph-aided conceptual product design approach based on joint entity and relation extraction. *Journal of Intelligent & Fuzzy Systems, 44(3),* 5333–5355. https://doi.org/10.3233/JIFS-223100
15. Bao, Q., Zhao, G., Yu, Y., Zheng, P. (2022). A node2vec-based graph embedding approach for unified assembly process information modeling and workstep execution time prediction. *Computers & Industrial Engineering, 163,* 107864. https://doi.org/10.1016/j.cie.2021.107864
16. Li, H., Wang, Y., Zhang, S., Song, Y., Qu, H. (2021). KG4Vis: A knowledge graph-based approach for visualization recommendation. *IEEE Transactions on Visualization and Computer Graphics, 28(1),* 195–205. https://doi.org/10.1109/TVCG.2021.3114863
17. Ahn, K., Park, J. (2019). Idle vehicle rebalancing in semiconductor fabrication using factorized graph neural network reinforcement learning. *Proceedings of the IEEE 58th Conference on Decision and Control (CDC)*, pp. 132–138. Nice.
18. Dong, J., Jing, X., Lu, X., Liu, J. F., Li, H. et al. (2021). Process knowledge graph modeling techniques and application methods for ship heterogeneous models. *Scientific Reports,* 2911. https://doi.org/10.1038/s41598-022-06940-y
19. Rabiner, L., Juang, B. (1986). An introduction to hidden markov models. *IEEE ASSP Magazine, 3(1),* 4–16. https://doi.org/10.1109/MASSP.1986.1165342
20. Hearst, M. A., Dumais, S. T., Osuna, E., Platt, J., Scholkopf, B. (1998). Support vector machines. *IEEE Intelligent Systems and Their Applications, 13(4),* 18–28. https://doi.org/10.1109/5254.708428
21. Steinwart, I., Christmann, A. (2008). *Support vector machines*. Springer Science & Business Media https://doi.org/10.1007/978-0-387-77242-4
22. Wallach, H. M. (2004). Conditional random fields: An introduction. *Technical Reports (CIS)*. https://repository.upenn.edu/cgi/viewcontent.cgi?article=1011&context=cis_reports
23. Huang, Z., Xu, W., Yu, K. (2015). Bidirectional LSTM-CRF models for sequence tagging. https://doi.org/10.48550/arXiv.1508.01991
24. Li, L., Wang, P., Yan, J., Wang, Y., Li, S. et al. (2020). Real-world data medical knowledge graph: Construction and applications. *Artificial Intelligence in Medicine, 103(19),* 101817.
25. Dou, J., Qin, J., Jin, Z., Li, Z. (2018). Knowledge graph based on domain ontology and natural language processing technology for Chinese intangible cultural heritage. *Journal of Visual Languages & Computing, 48(4),* 19–28. https://doi.org/10.1016/j.jvlc.2018.06.005

26. Devlin, J., Chang, M. W., Lee, K., Toutanova, K. (2019). BERT: Pre-training of deep bidirectional transformers for language understanding. *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 4171–4186. Minneapolis.
27. Zhang, Z., Han, X., Liu, Z., Jiang, X., Sun, M. et al. (2019). ERNIE: Enhanced language representation with informative entities. arXiv preprint arXiv:1905.07129.
28. Dong, L., Yang, N., Wang, W., Wei, F., Liu, X. et al. (2019). Unified language model pre-training for natural language understanding and generation. *Advances in Neural Information Processing Systems, 32*.
29. Yang, Z., Dai, Z., Yang, Y., Carbonell, J., Salakhutdinov, R. R. et al. (2019). XLNet: Generalized autoregressive pretraining for language understanding. *Advances in Neural Information Processing Systems, 32*.
30. Liu, C., Sun, W., Chao, W., Che, W. (2013). Convolution neural network for relation extraction. *Advanced Data Mining and Applications: 9th International Conference*, pp. 231–242. Hangzhou, China.
31. Zeng, D., Liu, K., Chen, Y., Zhao, J. (2015). Distant supervision for relation extraction via piecewise convolutional neural networks. *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pp. 1753–1762. Lisbon.
32. Lee, J., Seo, S., Choi, Y. S. (2019). Semantic relation classification via bidirectional LSTM networks with entity-aware attention using latent entity typing. *Symmetry, 11(6),* 785. https://doi.org/10.3390/sym11060785
33. Tai, K. S., Socher, R., Manning, C. D. (2015). Improved semantic representations from tree-structured long short-term memory networks. arXiv preprint arXiv:1503.00075.
34. Rönnqvist, S., Schenk, N., Chiarcos, C. (2017). A recurrent neural model with attention for the recognition of Chinese implicit discourse relations. arXiv preprint arXiv:1704.08092.
35. Xu, J., Wen, J., Sun, X., Su, Q. (2017). A discourse-level named entity recognition and relation extraction dataset for Chinese literature text. arXiv preprint arXiv:1711.07010.
36. Zhang, Y., Yang, J. (2018). Chinese NER using Lattice LSTM. arXiv preprint arXiv:1805.02023.
37. Zhou, B., Hua, B., Gu, X., Lu, Y., Peng, T. et al. (2021). An end-to-end tabular information-oriented causality event evolutionary knowledge graph for manufacturing documents. *Advanced Engineering Informatics, 50(12),* 101441. https://doi.org/10.1016/j.aei.2021.101441
38. Tran, T. K., Ta, C. D., Phan, T. T. (2022). Simplified effective method for identifying semantic relations from a knowledge graph. *Journal of Intelligent & Fuzzy Systems: Applications in Engineering and Technology, 43(2),* 1871–1876. https://doi.org/10.3233/JIFS-219288
39. Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M. et al. (2019). RoBERTa: A robustly optimized BERT pretraining approach. arXiv preprint arXiv:1907.11692.
40. Radford, A., Wu, J., Child, R., Luan, D., Amodei, D. et al. (2019). Language models are unsupervised multitask learners. *OpenAI Blog, 1(8),* 1–9.
41. Shazeer, N., Cheng, Y., Parmar, N., Tran, D., Vaswani, A. et al. (2018). Mesh-TensorFlow: Deep learning for supercomputers. *Advances in Neural Information Processing Systems, 31*. https://doi.org/10.48550/arXiv.1811.02084
42. Shoeybi, M., Patwary, M., Puri, R., LeGresley, P., Casper, J. et al. (2019). Megatron-LM: Training multi-billion parameter language models using model parallelism. arXiv preprint arXiv:1909.08053.
43. Chen, T., Xu, B., Zhang, C., Guestrin, C. (2016). Training deep nets with sublinear memory cost. arXiv preprint arXiv:1604.06174.
44. Gomez, A. N., Ren, M., Urtasun, R., Grosse, R. B. (2017). The reversible residual network: Backpropagation without storing activations. *Advances in Neural Information Processing Systems, 30*.
45. Chen, Y. J., Hsu, J. Y. J (2016). Chinese relation extraction by multiple instance learning. *Workshops at the Thirtieth AAAI Conference on Artificial Intelligence*, Phoenix, USA.

46. Zhang, Q., Chen, M., Liu, L. (2017). An effective gated recurrent unit network model for Chinese relation extraction. *DEStech Transactions on Computer Science and Engineering*. https://doi.org.10.12783/DTCSE/WCNE2017/19833
47. Dong, Z., Dong, Q. (2003). HowNet-a hybrid language and knowledge resource. *International Conference on Natural Language Processing and Knowledge Engineering*, pp. 820–824. Beijing, China.
48. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L. et al. (2017). Attention is all you need. *Advances in Neural Information Processing Systems, 30*.
49. Hendrycks, D., Gimpel, K. (2016). Gaussian error linear units (GELUs). arXiv preprint arXiv:1606.08415.
50. You, Y., Li, J., Reddi, S., Hseu, J., Kumar, S. et al. (2019). Large batch optimization for deep learning: Training BERT in 76 minutes. arXiv preprint arXiv:1904.00962.
51. Sperber, M., Neubig, G., Niehues, J., Waibel, A. (2017). Neural lattice-to-sequence models for uncertain inputs. arXiv preprint arXiv:1704.00559.
52. Zeng, D., Liu, K., Lai, S., Zhou, G., Zhao, J. (2014). Relation classification via convolutional deep neural network. *Proceedings of the 25th International Conference on Computational Linguistics*, pp. 2335–2344. Dublin.
53. Niu, Y., Xie, R., Liu, Z., Sun, M. (2017). Improved word representation learning with sememes. *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics*, pp. 2049–2058. Vancouver.
54. Hochreiter, S., Schmidhuber, J. (1997). Long short-term memory. *Neural Computation, 9(8),* 1735–1780. https://doi.org/10.1162/neco.1997.9.8.1735
55. Graves, A. (2013). Generating sequences with recurrent neural networks. arXiv preprint arXiv:1308.0850.
56. Hinton, G. E., Srivastava, N., Krizhevsky, A., Sutskever, I., Salakhutdinov, R. R. (2012). Improving neural networks by preventing co-adaptation of feature detectors. arXiv preprint arXiv:1207.0580.
57. Noy, N. F., McGuinness, D. L. (2001). Ontology development 101: A guide to creating your first ontology. https://protege.stanford.edu/publications/ontology_development/ontology101.pdf
58. Luo, S., Dong, Y. (2017). Classifying cultural artifacts knowledge for creative design. *Zhejiang Daxue Xuebao (Gongxue Ban)/Journal of Zhejiang University (Engineering Science Edition), 51(1),* 113–123.
59. Doerr, M. (2003). The CIDOC conceptual reference module: An ontological approach to semantic interoperability of metadata. *AI Magazine, 24(3),* 75.
60. Luo, S., Dong, Y. (2018). Integration and management method of cultural artifacts knowledge for cultural creative design. *Computer Integrated Manufacturing Systems, 24(4),* 964–977.
61. Hripcsak, G., Rothschild, A. S. (2005). Agreement, the F-measure, and reliability in information retrieval. *Journal of the American Medical Informatics Association, 12(3),* 296–298. https://doi.org/10.1197/jamia.M1733
62. Ogren, P., Savova, G., Chute, C. (2008). Constructing evaluation corpora for automated clinical named entity recognition. *Proceedings of the Sixth International Conference on Language Resources and Evaluation*, pp. 3143–3150. Marrakech.
63. Artstein, R., Poesio, M. (2008). Inter-coder agreement for computational linguistics. *Computational Linguistics, 34(4),* 555–596. https://doi.org/10.1162/coli.07-034-R2
64. Kataoka, H., Satoh, Y., Abe, K., Minoguchi, M., Nakamura, A. (2006). Inclusive design evaluation and the capability-demand relationship. *Designing Accessible Technology,* 177–188. https://doi.org/10.1007/1-84628-365-5_18

Appendix A. Construction of Corpus and Datasets

Figure A1: The annotated corpus

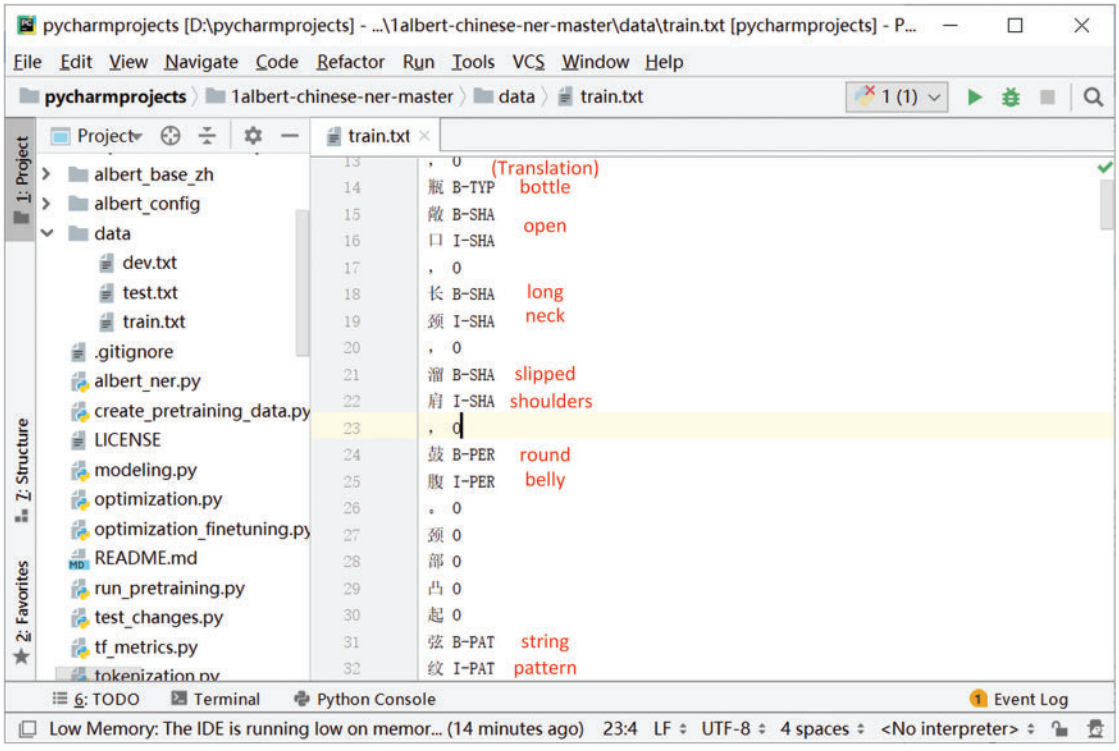

Figure A2: Design dataset for entity extraction

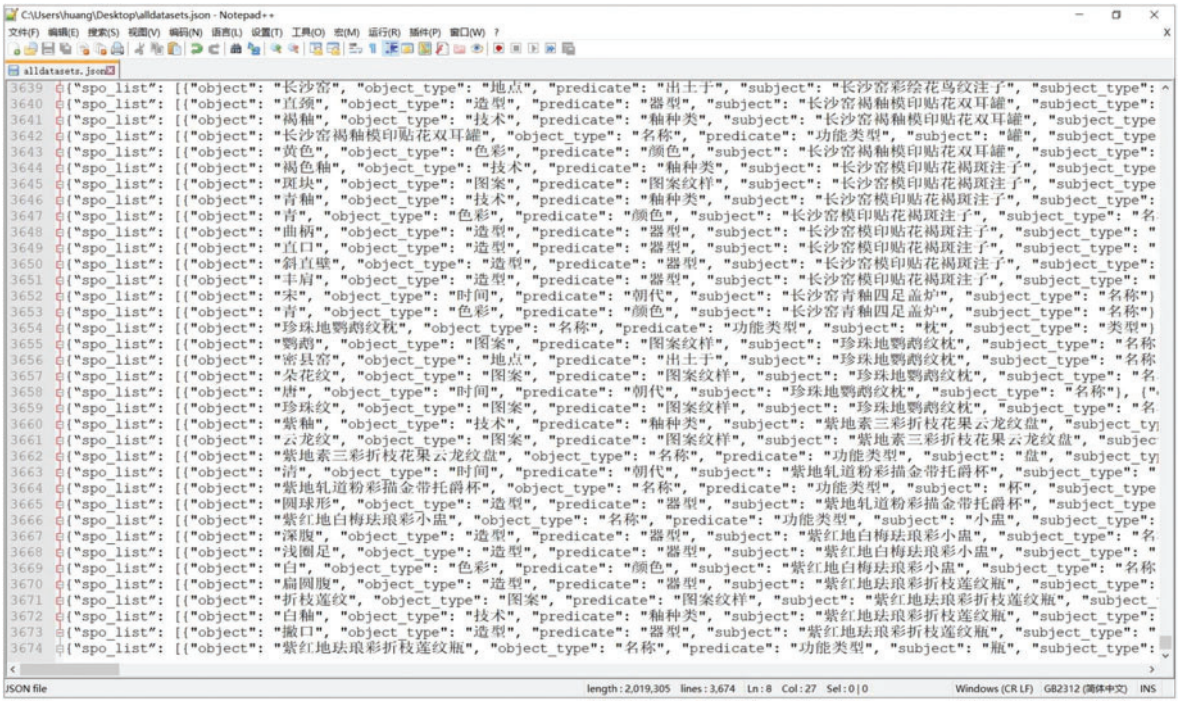

Figure A3: Design dataset for relation extraction