Automatic Skill-Oriented Question Generation and Recommendation for Intelligent Job Interviews

CHUAN QIN, Career Science Lab, BOSS Zhipin and PBC School of Finance, Tsinghua University
HENGSHU ZHU, Career Science Lab, BOSS Zhipin
DAZHONG SHEN, Shanghai Artificial Intelligence Laboratory
YING SUN, Hong Kong University of Science and Technology (Guangzhou)
KAICHUN YAO, Institute of Software,Chinese Academy of Sciences
PENG WANG, Baidu Talent Intelligence Center
HUI XIONG, Department of Computer Science and Engineering, HKUST; Thrust of Artificial Intelligence, HKUST (GZ)

Job interviews are the most widely accepted method for companies to select suitable candidates, and a critical challenge is finding the right questions to ask job candidates. Moreover, there is a lack of integrated tools for automatically generating interview questions and recommending the right questions to interviewers. To this end, in this paper, we propose an intelligent system for assisting job interviews, namely, DuerQues. To build this system, we first investigate how to automatically generate skill-oriented interview questions in a scalable way by learning external knowledge from online knowledge-sharing communities. Along this line, we develop a novel distantly supervised skill entity recognition method to identify skill entities from large-scale search queries and web page titles with less need for human annotation. Additionally, we propose a neural generative model for generating skill-oriented interview questions. In particular, we introduce a data-driven solution to create high-quality training instances and design a learning algorithm to improve the performance of question generation. Furthermore, we exploit click-through data from query logs and design a recommender system for recommending suitable questions to interviewers. Specifically, we introduce a graph-enhanced algorithm to efficiently recommend suitable questions given a set of queried skills. Finally, extensive experiments on real-world datasets demonstrate the effectiveness of our DuerQues system in terms of the quality of generated skill-oriented questions and the performance of question recommendation.

Part of the work was done when Chuan Qin was employed by Baidu Inc.
This work was partially supported by grants from the City-University Joint Funding Project of Guangzhou Science and Technology Plan (No. 2023A03J0141).
Authors' addresses: C. Qin, Career Science Lab, BOSS Zhipin and PBC School of Finance, Tsinghua University; Grandyvic Building Taiyanggong Middle Rd 6/f, Beijing, Beijing, China, 100028; email: chuanqin0426@gmail.com; H. Zhu (corresponding author), Career Science Lab, BOSS Zhipin; Grandyvic Building Taiyanggong Middle Rd 6/f, Beijing, Beijing, China, 100028; email: zhuhengshu@gmail.com; D. Shen, Shanghai Artificial Intelligence Laboratory; 701 Yunjin Road, Xuhui District, Shanghai, Shanghai, China, 201210; email: dazh.shen@gmail.com; Y. Sun, Hong Kong University of Science and Technology (Guangzhou); No. 1 Duxue street, Nansha District, Guanghzou, Guangdong, China, 511455; email: yings@hkust-gz.edu.cn; K. Yao, Institute of Software, Chinese Academy of Sciences; 4# South Fourth Street, Zhong Guan Cun, Beijing, Beijing, China, 100190; email: yaokaichun@outlook.com; P. Wang, Baidu Talent Intelligence Center; Baidu Campus, No. 10 Shangdi 10th Street, Haidian District, Beijing, Beijing, China, 100085; email: wangpeng40@baidu.com; H. Xiong (corresponding author), Department of Computer Science and Engineering, HKUST; Thrust of Artificial Intelligence, HKUST (GZ); No. 1 Duxue street, Nansha District, Guanghzou, Guangdong, China, 511455; email: xionghui@ust.hk.
Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than the author(s) must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.
© 2023 Copyright held by the owner/author(s). Publication rights licensed to ACM.
1046-8188/2023/08-ART27 $15.00
https://doi.org/10.1145/3604552

CCS Concepts: • **Information systems → Data mining;**

Additional Key Words and Phrases: Job interview assessment, question generation, question recommendation

ACM Reference format:
Chuan Qin, Hengshu Zhu, Dazhong Shen, Ying Sun, Kaichun Yao, Peng Wang, and Hui Xiong. 2023. Automatic Skill-Oriented Question Generation and Recommendation for Intelligent Job Interviews. *ACM Trans. Inf. Syst.* 42, 1, Article 27 (August 2023), 32 pages.
https://doi.org/10.1145/3604552

1 INTRODUCTION

Job interviewing is an essential process of talent acquisition, and it helps employers assess whether a candidate's capabilities align with the job requirements [62]. A successful job interview process enables a company to find the right talent efficiently and accurately, gaining an advantage in the increasingly competitive recruitment market [21, 30]. Therefore, substantial efforts have been recently made to improve the interviewing process, such as automated interview scheduling [23], automated video interview analysis [36], and intelligent interview decision-making [35, 53].

Furthermore, a key challenge for job interviewing is to find the right questions to ask job candidates [87], especially questions related to job knowledge, skills, and abilities [31, 61]. Interviewers who fail to prepare suitable questions beforehand may fail to effectively evaluate candidates [7, 70]. While considerable efforts have been devoted to recruitment data-based question recommendation [66, 74], there still exists a critical challenge in building a comprehensive skill-oriented interview question bank for interviewers [17]. Typically, the construction of question banks relies on the extensive effort of domain experts, which incurs high human and financial resource costs and requires regular updates to evaluate emerging and evolving skills [52, 54].

To this end, in this paper, we propose an intelligent system, namely, DuerQues, to achieve skill-oriented automatic interview question generation and recommendation. Specifically, the motivation is to exploit skill-related knowledge from the user-generated content of online **knowledge-sharing communities (KSCs)** and the click-through behaviors of search engine queries. As shown in Figure 1, by matching the skill keywords (e.g., "*CNN*"), it is easy to locate web pages in KSCs that contain relevant knowledge, and these pages can be used for collecting interview materials. Moreover, the click-through data from search engine queries can help us aggregate web pages and cluster correlated skills for job interviews (e.g., "*Backward Propagation*" and "*Activation Function*"). This is because users usually click on similar web pages for similar search intents. In particular, some high-quality queries are naturally good interview questions, such as "*How does backward propagation work in a CNN?*", which provide effective training samples for question generation.

Based on the above intuition, to build our system, we first investigate how to automatically generate skill-oriented interview questions in a scalable way by learning external knowledge from online KSCs. Along this line, we develop a novel distantly supervised skill entity recognition method to identify skill entities from large-scale search queries and web page titles with less need for human annotation to construct a training set. In particular, we introduce a vocabulary-based annotation method to generate a large number of distantly supervised training instances. Then, inspired by knowledge transfer [90, 109], we propose a novel skill-oriented interview question generation model. Specifically, we construct a query-web page bipartite graph and design a data-driven heuristic algorithm to collect high-quality training data. Next, we use click-through data from the query logs to develop a question recommender system based on the generated interview questions. Specifically, we design a graph-enhanced question recommendation algorithm to recommend suitable questions given a set of queried skills. Moreover, we introduce a skill entity suggestion algorithm to help interviewers more efficiently use our recommender system. Finally, we conduct extensive experiments on real-world datasets to demonstrate the effectiveness of our

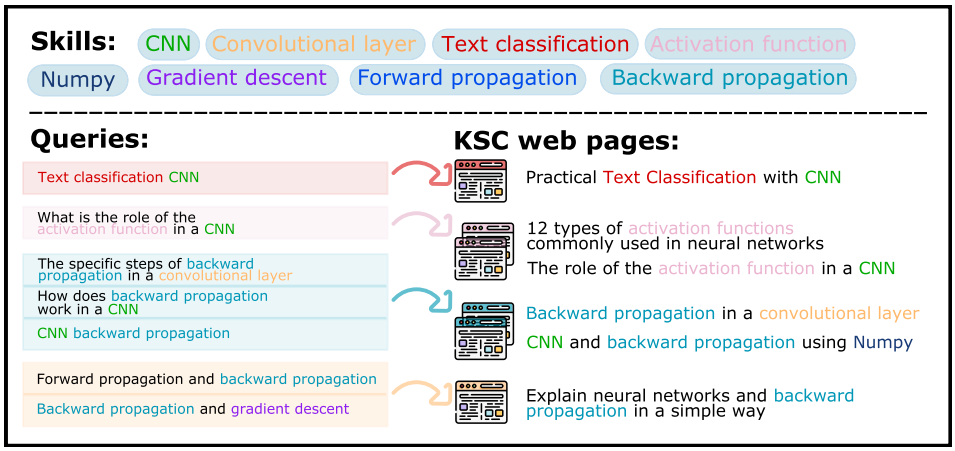

Fig. 1. Motivation example of our work.

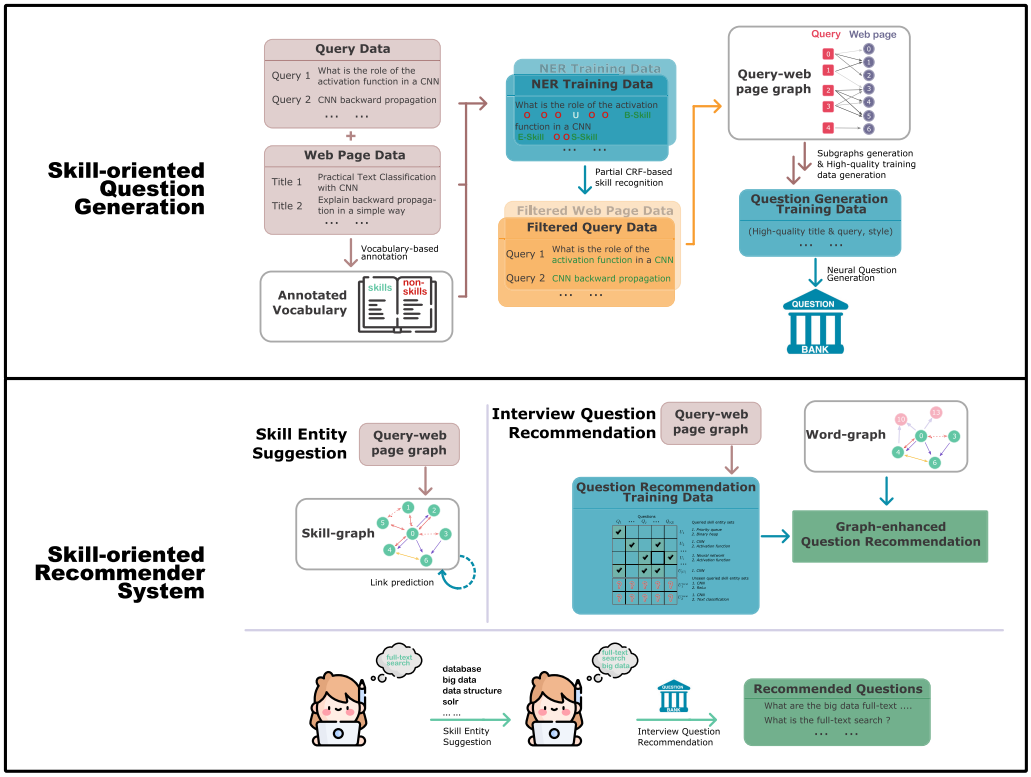

Fig. 2. A schematic diagram of our approach, which consists of skill-oriented question generation and the skill-oriented recommender system.

DuerQues system in terms of the quality of generated skill-oriented questions and the performance of question recommendation. Figure 2 shows an overview of the framework of our approach.

Overview. The rest of this paper is organized as follows. In Section 2, we briefly introduce some related work on intelligent job interviews, skill entity recognition, text generation, graph learning,

and recommender systems. In Section 3, we present our solutions for automatic skill-oriented interview question generation, which includes two steps: **Distantly supervised skill entity recognition (DuerQues-SER)** and **interview question generation (DuerQues-IQG)**. In Section 4, we introduce our skill-oriented recommender system, which consists of **skill entity suggestion (DuerQues-SES)** and **interview question recommendation (DuerQues-IQR)** components. In Section 5, we comprehensively evaluate the performance of each part of our DuerQues system. In Section 6, we provide further discussion and present case studies. Finally, we conclude this paper in Section 7.

2 RELATED WORK

Generally, the related work of this paper can be grouped into five categories, including: intelligent job interviews, skill entity recognition, text generation, graph representation learning, and recommender systems.

2.1 Intelligent Job Interviews

Job interviewing is one of the most critical processes in talent recruitment and has attracted wide attention from the AI community. Researchers have proposed many intelligent algorithms to improve the accuracy and efficiency of job interviewing from different perspectives, such as automated interview scheduling [23] and automated video interview analysis [12, 36]. For job interviewers, one of the most critical and time-consuming tasks in preparing for a successful interview is to determine the best job interview questions. For instance, Shen et al. developed a latent variable model to exploit the latent representation of historical recruitment data and recommended interview questions based on the cosine similarity between the representations of questions and a candidate's resume [74]. Then, Qin et al. constructed a skill-graph by using recruitment and web search data, and proposed a personal question recommender system by leveraging the hypernym-hyponym relations between skills [66]. Along this line, Datta et al. further formalized interview question recommendation as an integer programming problem and used a knowledge graph to select the interview questions from the question bank based on a candidate's resume [17].

Existing studies usually assume that a set of standard candidate questions are available. However, collecting or designing suitable interview questions is not a trivial task, and this assumption may not hold for quickly evolving professional roles [7]. Recently, Shi et al. introduced a two-stage deep learning model to generate interview questions, such as "Do you speak Japanese?" [75]. However, their approach cannot generate in-depth skill-oriented questions. To address these challenges, in this paper, we design an intelligent system, namely, DuerQues, which can create an interview question bank based on skill-oriented question generation and effectively assist interviewers in selecting or designing the suitable interview questions based on a skill-centered recommender system.

2.2 Skill Entity Recognition

Skill recognition is a fundamental task in talent recruitment, which benefits many downstream applications [1, 76, 92, 97]. In our paper, we regard it as a traditional **named entity recognition (NER)** problem, which aims to locate the entities and identify their categories from the given textual content. It is an important and fundamental task in **natural language processing (NLP)** [85, 106]. NER has usually been treated as a sequence labeling problem, where traditional studies rely on high-quality hand-crafted features and employ feature-based supervised learning methods, such as the **hidden Markov model (HMM)** [106] and **conditional random fields (CRFs)** [39]. In recent years, with the development of deep learning techniques, neural-based NER approaches have become dominant and have achieved state-of-the-art performance [42] since they can eliminate the dependence on labor-intensive features and automatically learn intricate

semantic information from input data. For instance, Lample et al. proposed leveraging **bidirectional long short-term memory (BiLSTM)** networks to extract word-level information and applying a CRF layer to infer NER tags [41]. Ma and Hovy utilized a **convolutional neural network (CNN)** layer to exploit each word's character-level information and concatenated the pretrained word vectors before feeding them into BiLSTM-CRF [50]. Moreover, some studies [44, 57] have used pretrained language models, such as **bidirectional encoder representations from transformers (BERT)** [18], to further enhance the performance of neural-based NER models. Compared with English NER, Chinese NER is more difficult because the Chinese language lacks explicit word separators [89]. The above-mentioned word-level NER models result in errors in entity boundary detection without artificially labeled gold-standard segmentation. Therefore, the character-based model has been the dominant approach for Chinese NER [57, 102]. Along this line, researchers have focused on using the semantic features of words. For instance, Zhang and Yang proposed using lattice-structured LSTM to integrate latent word information by representing lexicon words from input sentences [102]. Peng et al. [57] designed the SoftLexicon feature to exploit latent word information, and this approach achieves state-of-the-art performance in Chinese NER.

The above approaches based on supervised learning require a large amount of human-annotated NER training data, which is usually expensive to obtain [11], especially for low-resource languages/domains or unusual entity categories. Some works have leveraged transfer learning techniques to handle the low-resource domain NER problem by utilizing plentiful annotations in related source domains [96]. However, they usually focus on cross-language or cross-domain NER tasks, which have trouble handling new types of entities in the target language/domain. Recently, inspired by the idea of using distantly supervised learning for the relation extraction task [99], which can automatically annotate a large amount of distantly supervised instances for training a model, there are a few works that have attempted to utilize distantly supervised learning for the NER task. However, those works either require a certain amount of gold-standard annotated data for NER [94] or rely on external knowledge bases such as Wikipedia [11]. In this paper, we introduce a novel distantly supervised neural-based NER model for recognizing skill entities, which requires only a small number of word annotations.

2.3 Text Generation

Text generation, also known as **natural language generation (NLG)**, aims to produce a natural language text from input information [48] and covers a wide range of NLP tasks [63], such as machine translation [5, 81], text summarization [43, 68, 72], question generation [19, 47], poem generation [91], and dialog systems [32, 103]. Most of the above applications can be classified as text-to-text generation tasks. Traditionally, template-based [107] and rule-based [86] methods have been leveraged for this task. Recently, neural sequence-to-sequence models have become an ideal choice for text generation; they were first introduced in [78] and quickly achieved success in a variety of text generation tasks [5, 72]. Neural sequence-to-sequence models leverage neural network architectures, such as CNNs and **recurrent neural networks (RNNs)**, with an encoder to represent the input and a decoder to generate the target text from the encoded representation. For instance, Cho et al. leveraged RNNs both as the encoder and decoder to build a neural machine translation model [14]. Gehring et al. introduced a sequence-to-sequence model entirely based on CNNs, which enables fast and accurate text generation [22]. Furthermore, some researchers have designed several variants to enhance performance. For instance, an attention mechanism allows a decoder to dynamically select parts of the input representations according to their relevance to the next generated word, which has dramatically improved the performance of neural machine translation [5]. Copy mechanisms have been widely used in text generation, which helps generate rare target words that appear in input text [25, 72].

In this paper, we focus on interview question generation, which is one specific kind of question generation task. Traditionally, question generation is formulated as generating corresponding target questions given various inputs such as raw text, a database, or semantic representation [56]. Therefore, several sequence-to-sequence approaches have been proposed to solve this problem in an end-to-end manner [19, 20]. However, existing methods are highly dependent on large-scale and high-quality training data, which are difficult to obtain in many specialized domains, such as talent recruitment. In response to this challenge, we propose a novel approach for creating training instances and utilize a sequence-to-sequence modeling solution to generate interview questions in our DuerQues system. Moreover, we adapt the original sequence-to-sequence model with a new learning method to handle the problem of insufficient high-quality interrogative training sentences.

2.4 Graph Representation Learning

Graph-structured data is ubiquitous throughout the computer sciences, from click-through graphs of web search logs to knowledge graphs [26]. Graph representation learning aims to automatically encode nodes into a low-dimensional space while conserving the graph properties and structural information [29, 101]. Great efforts have been made on this fundamental task from different perspectives [13, 37, 59, 73, 98, 100, 104]. For instance, graph factorization techniques can be used to factorize a node-node similarity matrix to learn low-dimensional graph embeddings [3, 6, 10, 55]. Specifically, Belkin and Niyogi proposed a Laplacian eigenmaps method that leverages the inner product of the embedding vectors to reproduce the weights of the graph Laplacian [6], while Ahmed et al. focused on reconstructing the weights of the adjacency matrix [3]. Similarly, Ou et al. factorized general neighborhood overlap matrices [55]. The above studies mainly optimize representations by directly reconstructing predefined matrices. In contrast, random walk techniques, such as DeepWalk [59] and node2vec [24], consider the constraint that two nodes have similar embeddings when they tend to co-occur on short random walks over a graph [29]. Recently, with the development of **graph neural networks (GNNs)**, node representations can be effectively explored from rich neighborhood information [16, 95]. For instance, a **graph convolutional network (GCN)** leverages efficient symmetric-normalized aggregation and the self-loop update approach based on a first-order approximation of spectral convolutions on graphs [37]. Veličković et al. introduced a **graph attention network (GAT)**, which employs a self-attention mechanism to dynamically aggregate node neighbors' information [82]. Moreover, Schlichtkrull et al. proposed a **relational GCN (RGCN)** to encode nodes in a heterogeneous graph [71]. Inspired by the above approaches, we design a graph-enhanced question recommendation algorithm and a skill suggestion algorithm, which take advantage of the rich structural information from two specifically designed heterogeneous graphs.

2.5 Recommender Systems

Recommender systems are a subclass of information filtering systems that seek to recommend suitable content for users based on their preferences, interests, or observed behaviors [33]. They have been widely used in a variety of applications [38, 64, 65, 83, 108], such as product recommenders for e-commerce [105], content recommenders for social media [27] and open web content recommenders [45]. Typically, recommender systems can be grouped into three categories: Collaborative filtering-based recommender systems, content-based recommender systems, and hybrid recommender systems [9]. Collaborative filtering-based methods model user preferences based on interaction data, while content-based methods leverage rich information from an item's content [26, 84]. Hybrid recommendation is proposed to overcome the overspecialization, data sparsity, and cold start problems, and it combines the advantages of the above approaches [9]. In this paper, we focus

on recommending suitable interview questions for interviewers from the generated question bank. Furthermore, several approaches have been applied to address the question recommendation task in online question-answering communities. For instance, Qu et al. applied a topic model to capture user interests based on their answering history and recommended relevant questions to them [67]. Yang et al. introduced a context-aware matrix factorization model to recommend questions under the constraints of load balancing and expertise matching [93]. Tu et al. proposed a hybrid method that jointly models implicit and explicit information for question recommendation [80]. However, most of the above approaches are mainly based on the users' historical data in the communities, which are unavailable in our scenario. In this paper, we convert the original task into the skill entity suggestion and interview question recommendation based on a queried skill entity set, to assist interviewers in selecting and designing interview questions. Moreover, we innovatively utilize search query data to construct the necessary interaction data for model training.

3 SKILL-ORIENTED QUESTION GENERATION

This section introduces how to generate skill-oriented interview questions automatically. Specifically, it mainly contains two steps, i.e., distantly supervised skill entity recognition (DuerQues-SER) and interview question generation (DuerQues-IQG). For reference, we list some important mathematical notations used throughout this paper in Table 1.

3.1 Distantly Supervised Skill Recognition

To effectively exploit the external knowledge from online KSCs and click-through data to generate skill-oriented interview questions, we first need to identify skill entities from these data sources, including queries and KSC web page titles. Moreover, skill recognition can be regarded as a NER task, which is usually formulated as a sequence labeling problem and has achieved impressive performance by utilizing several well-designed neural network models [41, 57]. However, existing NER methods are mainly based on supervised learning, which requires considerable labor to annotate training data [94]. To this end, we propose a new distantly supervised skill entity recognition method that only requires a certain number of word annotations. Additionally, we argue that our method can serve as a paradigm for solving NER tasks in other similar industrial fields. It is more efficient than labeling all or part of the NER training instances [94] and does not need to rely on additional external knowledge data [11].

Specifically, as shown in Figure 3, DuerQues-SER comprises three steps: (1) Vocabulary annotation, (2) NER training data generation, and (3) partial CRF-based neural skill entity recognition. Formally, given a set of input text sentences $\mathcal{X} = \{X_1, X_2, \ldots, X_{|\mathcal{X}|}\}$, i.e., queries or web page titles, we aim to extract all skill entities $\{s_{i,1}, s_{i,2}, \ldots, s_{i,m_i}\}$ for each input sentence, where $s_{i,j}$ and m_i denote the j-th skill entity and the number of skill entities in X_i, respectively. In addition, following recent character-based Chinese NER approaches [11, 102], we denote each input sentence X_i as a sequence of characters, i.e., $X_i = \{c_{i,1}, c_{i,2}, \ldots, c_{i,n_i}\}$, and our goal is to infer a sequence of labels $Y_i = \{y_{i,1}, y_{i,2}, \ldots, y_{i,n_i}\}$, where n_i denotes the length of the character sequence and each $y_{i,j} \in \{B, I, O, E, S\}$ indicates if $c_{i,j}$ is the *Beginning*, the *Inside*, the *Outside*, the *Ending*, or a *Singleton* of a skill entity mention.

Vocabulary Annotation. One way to eliminate the need for labeling every entity mentioned in training input is to train a NER model with automatically generated annotation data, namely, distantly supervised learning [94]. Typically, this leverages an externally typed entity dictionary to match the entity mentioned in the target text and generate training instances. Existing studies either assume that the dictionary is known [94] or rely on external knowledge such as Wikipedia [11], which leads to the limited matching coverage of the entities mentioned in specific domains, such as recruitment [66]. Here, we introduce a heuristic algorithm to directly annotate

Table 1. Mathematical Notations

Symbols	Description
$\left(X_i, \widehat{Y}_i\right)$	The i-th generated distantly supervised NER training instance;
$s_{i,j}$	The j-th skill entity in input sentence X_i;
$c_{i,j}$	The j-th character in input sentence X_i;
m_i, n_i	The numbers of skill entities and characters in X_i;
$\widehat{y}_{i,j}$	The j-th label in $\widehat{Y}_i$ corresponding to $c_{i,j}$ in X_i;
$\mathcal{V}^+$, $\mathcal{V}^-$	The labeled skill and non-skill entity dictionaries;
$x^c_{i,j}$	The j-th combination of character and character bigram semantic features in X_i;
$x^{\text{BERT}}_{i,j}$, $x^s_{i,j}$, $x^l_{i,j}$	The BERT, manual character-level, SoftLexicon features for $c_{i,j}$ in X_i;
$f^c_e(\cdot)$, $f^b_e(\cdot,\cdot)$	The character and character bigram embedding lookup tables in the neural skill entity recognition model;
$h_{i,j}$	The hidden state of $c_{i,j}$ in the neural skill recognition model;
$P_{i,j,y_{i,j}}$	The score of assigning $c_{i,j}$ with label $y_{i,j}$;
$A_{y_{i,j},y_{i,j+1}}$	The learnable transition score from label $y_{i,j}$ to $y_{i,j+1}$;
$\mathcal{Y}_i$	All the possible label sequences of input X_i;
$\widehat{\mathcal{Y}}_i$	All possible label sequences that keep the position labels in $\{B, I, O, E, S\}$ unchanged and traverse all possible labels for the unlabeled positions of input X_i;
$\mathcal{G} = (\mathcal{V}, \mathcal{E})$	The query-web page graph;
$\mathcal{V}^q$, $\mathcal{V}^u$	The query nodes and web page nodes;
$e_{i,j} \in \mathcal{E}$	There exists a click behavior between query v^q_i and web page v^u_j;
$\mathscr{G}$, $\mathcal{G}_k = (\mathcal{V}_k, \mathcal{E}_k)$	The subgraphs of $\mathcal{G}$, and k-th subgraph in $\mathscr{G}$;
$\mathcal{W}_{i,j}$, $\mathcal{W}^k_{i,j}$	The count of click behavior between v^q_i and v^u_j in $\mathcal{G}$ and $\mathcal{G}_k$;
C^u_j, C^q_i	The numbers of unique skill entities in the title of the web page v^u_j and query v^q_i;
$\widetilde{v}^{u,k}$, $\widetilde{v}^{q,k}$	The selected title of the web page and query of subgraph $\mathcal{G}_k$;
$f(\cdot)$, $g(\cdot,\cdot)$	The normalization function and string similarity function;
$w^u_{k,i}$, $w^q_{k,i}$	The i-th word in $\widetilde{v}^{u,k}$ and $\widetilde{v}^{q,k}$;
$l^u_{k,i} \in \{0, 1\}$	Whether $w^u_{k,i}$ is a skill entity;
n^u_k, n^q_k	The numbers of words in $\widetilde{v}^{u,k}$ and $\widetilde{v}^{q,k}$;
f^w_e	The word embedding lookup table in the neural question generation model;
$h^u_{k,i}$, $h^q_{k,i}$	The i-th hidden states of the encoder and decoder in the neural question generation model with given input $\widetilde{v}^{u,k}$;
$c_{k,j}$	The j-th context feature from the encoder states in the neural question generation model;
$\mathcal{G}^s = (\mathcal{V}^s, \mathcal{E}^s)$	The skill-graph $\mathcal{G}^s$ contains the skill entity nodes $\mathcal{V}^s$ and skill entity co-occurrence edges $\mathcal{E}^s$;
$\mathcal{E}^{qq}$, $\mathcal{E}^{uu}$, $\mathcal{E}^{qu}$	The skill entity co-occurrence edges of queries, the titles of web pages, and the click behaviors between queries and web pages in $\mathcal{G}^s$;
$\mathcal{G}^c = (\mathcal{V}^c, \mathcal{E}^c)$	The word-graph $\mathcal{G}^c$;
$\mathcal{V}^c = \mathcal{V}^s \bigcup \mathcal{V}^w$	$\mathcal{V}^c$ contains the skill entity nodes $\mathcal{V}^s$ and words of the selected web page titles;
$\mathcal{E}^{qw}$	The word co-occurrence edges of between the queries and selected web page titles;
$h^s_{l,i}$	The hidden state of skill entity node v^s_i in the l-th RGCN layer;
$f^s_e(\cdot)$	The skill entity embedding lookup table in the skill entity suggestion model;
$f_{dm}(\cdot,\cdot)$, W^{dm}	The DistMult factorization and its trainable parameter;
$\mathcal{U}$, U_i	The queried skill sets and i-th queried skill set;
$\mathcal{Q}$, Q_j	The question set and j-th question;
S_j	The unique skill set of web page titles in subgraph $\mathcal{G}_j$;
$v^s_{z^u_k}$, $v^s_{z^s_k}$	The z^u_k-th and z^s_k-th skill entity nodes in U_i and S_j;
$f^r_e(\cdot)$	The word embedding lookup table in the interview question recommendation model;
$x^u_{I,k}$, $x^t_{j,k}$, $x^s_{j,k}$	The word embedding vectors of the k-th words in U_i, $\widetilde{v}^{u,j}$, and S_j;
$x^{cu}_{i,k}$, $x^{ct}_{j,k}$, $x^{cs}_{j,k}$	The graph embedding vectors of the k-th words in U_i, $\widetilde{v}^{u,j}$, and S_j;
f_g	The MLP layer in the interview question recommendation model.

part of the vocabulary of our data to construct a skill entity dictionary, which can be further used to generate high-quality training instances. In addition, the non-skill words are automatically annotated in this process, which helps us obtain the *Outside* labels in the training instances with higher confidence compared with the existing non-entity sampling method [11].

Specifically, we first use a Chinese word segmentor to split all input sentences in $\mathcal{X}$ and further construct a dictionary. Please note that although the uncertainty of the segmentation performance

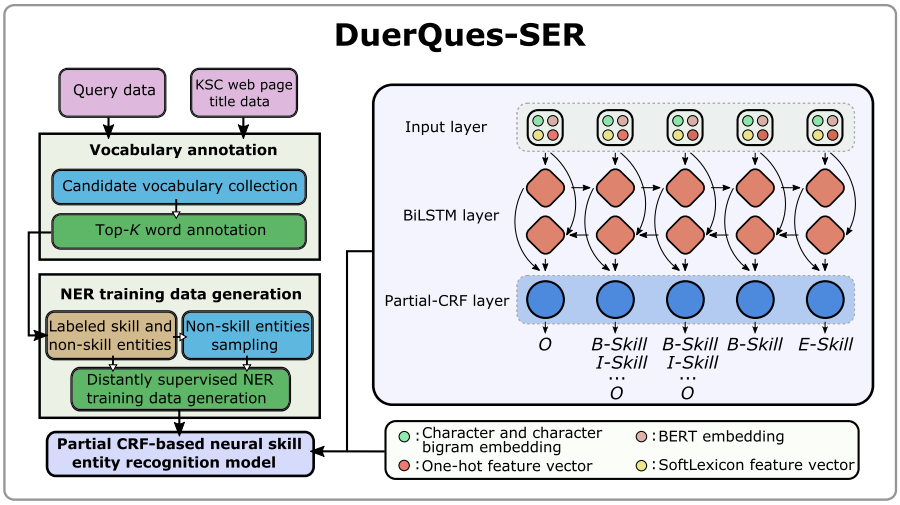

Fig. 3. A schematic diagram of distantly supervised skill entity recognition (DuerQues-SER).

makes the vocabulary unable to cover all words, the character-based NER model we introduce later can help us capture skills outside the vocabulary. Then, we leverage **part-of-speech (POS)** tagging to label each word in the input sentence and label each word as its most frequent POS tag in the vocabulary. Along this line, we add all nouns and English words into a candidate list $\mathcal{V}$ and score words in $\mathcal{V}$ according to their document frequency df in $\mathcal{X}$. Finally, we label the top-K words to construct our skill and non-skill entity dictionaries $\mathcal{V}^+$ and $\mathcal{V}^-$. In practice, we found that manual skill word annotation is much more efficient than standard NER training instance labeling. Moreover, we can leverage the existing skill dictionary from external resources to further reduce the annotation cost.

NER Training Data Generation. Now, we can generate NER training instances by leveraging the skill and non-skill entity dictionaries $\mathcal{V}^+$ and $\mathcal{V}^-$. Specifically, we first use a multi-pattern string search algorithm, i.e., the Aho-Corasick search algorithm [4], to locate all matched words $\{(start_{i,1}, end_{i,1}, \widehat{s}_{i,1}), (start_{i,2}, end_{i,2}, \widehat{s}_{i,2}), ...\}$ from $\mathcal{V}^+$ and $\mathcal{V}^-$ for each input X_i, where $start_{i,j}$ and $end_{i,j}$ denote the start index and end index of the j-th matched word $\widehat{s}_{i,j}$ in X_i, respectively.[1] Then, we remove all overlapping words $\widehat{s}_{i,j}$ that are strictly substrings of other words, i.e., we remove $\widehat{s}_{i,j}$ if there is another word $\widehat{s}_{i,k}$ such that $[start_{i,j}, end_{i,j}] \subset [start_{i,k}, end_{i,k}]$. Next, to deal with all the partially overlapping matched words, we calculate the frequency of each word in $\mathcal{V}^+$ and $\mathcal{V}^-$ occurring in $\mathcal{X}$ based on the above filtered matched word sequence and remove the matched words with lower frequency among the partially overlapping words.

Along this line, we can collect a matched word sequence without overlap and assign the distantly supervised labels corresponding to the positions of the matched words based on the *BIOES* tagging scheme. In particular, we set the unlabeled positions matching *stop word* to O for each input sequence X_i. In addition, we set the labels of all the remaining positions to U to denote that the characters in those positions are *Unlabeled*. To alleviate the reliance on the number of annotated words, i.e., K, we use a ratio α to sample words as non-skill words that do not appear in $\mathcal{V}$ based on $tf * df$, where tf denotes the term frequency in $\mathcal{X}$. Then, we transfer the label of positions from U to O that match those non-skill words. Here, we denote each generated distantly supervised NER training instance as $(X_i, \widehat{Y}_i)$, where $\widehat{Y}_i = \{\widehat{y}_{i,1}, \widehat{y}_{i,2}, \ldots, \widehat{y}_{i,n_i}\}$ and $\widehat{y}_{i,j} \in \{B, I, O, E, S, U\}$.

[1]We have excluded the scenario where a complete English word was partially matched by the words in $\mathcal{V}^+$ and $\mathcal{V}^-$.

Partial CRF-based Neural Skill Entity Recognition. Because our distantly supervised training instances contain many unlabeled positions (i.e., the positions labeled as U), we have to modify the traditional NER model. Here, we introduce a partial CRF-based neural model for skill entity recognition. Specifically, given an input character sequence $X_i = \{c_{i,1}, c_{i,2}, \ldots, c_{i,n_i}\}$, we first concatenate the character and character bigram representations to capture contextual linguistic information for each character $c_{i,j}$; that is, $x^c_{i,j} = [f^c_e(c_{i,j}); f^b_e(c_{i,j}, c_{i,j+1})]$, where $f^c_e(\cdot)$ and $f^b_e(\cdot,\cdot)$ denote the character and character bigram embedding lookup tables, respectively. Furthermore, since the pretrained BERT language model has been shown to benefit the NER task [18, 57], we also employ the BERT representation $x^{\text{BERT}}_{i,j}$ of each character to enhance the input representations. Then, we extract several character-level features (e.g., whether the character is uppercase and whether the previous/next character is a space/number/English alphabet) and construct them into a one-hot feature vector $x^s_{i,j}$. Moreover, we follow the methodology introduced by Peng et al. to produce SoftLexicon features [57], which can effectively exploit the word information. For every character $c_{i,j}$ in X_i, we denote the SoftLexicon feature vector as $x^l_{i,j}$.

Then, we concatenate all the representations/features as the input of the sequence modeling layer, i.e., a BiLSTM layer, as follows:

$$x^{all}_{i,j} = \left[x^c_{i,j}; x^{\text{BERT}}_{i,j}; x^s_{i,j}; x^l_{i,j}\right], \quad h_{i,j} = BiLSTM\left(x^{all}_{i,j}, h_{i,j-1}\right). \tag{1}$$

In the standard BiLSTM-CRF model, the CRF layer is designed to calculate the conditional probability $p(\widehat{Y}_i|X_i)$; that is:

$$p(\widehat{Y}_i|X_i) = \frac{exp(score(H_i, \widehat{Y}_i))}{\sum_{\widetilde{Y}_i \in \mathcal{Y}_i} exp(score(H_i, \widetilde{Y}_i))}, \tag{2}$$

$$P_{i,j} = W_p h_{i,j} + b_p, \quad score(H_i, Y_i) = \sum_{j=1}^{m_i} P_{i,j,y_{i,j}} + \sum_{j=0}^{m_i} A_{y_{i,j}, y_{i,j+1}}, \tag{3}$$

where $\mathcal{Y}_i$ denotes all possible label sequences of input X_i and $H_i = \{h_{i,1}, \ldots, h_{i,m_i}\}$ represents the hidden state sequence of X_i. We use $P_{i,j,y_{i,j}}$ to indicate the score for assigning label $y_{i,j}$ to the j-th character $c_{i,j}$ and let $A_{y_{i,j}, y_{i,j+1}}$ denote the learnable transition score from label $y_{i,j}$ to $y_{i,j+1}$, where W_p and b_p are trainable parameters.

Along this line, the standard BiLSTM-CRF model can be learned by maximizing the probability $p(\widehat{Y}_i|X_i)$. However, it cannot handle unlabeled positions in $\widehat{Y}_i$. Therefore, we employ a partial-CRF layer that can learn from incomplete annotations for sequence labeling tasks [11, 79]. Specifically, given an instance $(X_i, \widehat{Y}_i)$, we first keep the position labels in $\{B, I, O, E, S\}$ unchanged and traverse all possible labels for the positions labeled as U in label sequence $\widehat{Y}_i$. In this way, we can build a set of sequences $\widehat{\mathcal{Y}}_i$, in which each $\widetilde{Y}_i \in \widehat{\mathcal{Y}}_i$ is a possible label sequence of X_i based on $\widehat{Y}_i$. Then, we can compute the total probability of label sequences $\widehat{\mathcal{Y}}_i$ by:

$$p(\widehat{\mathcal{Y}}_i|X_i) = \sum_{\widetilde{Y}_i \in \widehat{\mathcal{Y}}_i} p(\widetilde{Y}_i|X_i) = \frac{\sum_{\widetilde{Y}_i \in \widehat{\mathcal{Y}}_i} exp(score(H_i, \widetilde{Y}_i))}{\sum_{\widetilde{Y}'_i \in \mathcal{Y}_i} exp(score(H_i, \widetilde{Y}'_i))}. \tag{4}$$

Finally, for training the parameters, we exploit a negative log-likelihood objective as the loss function to maximize the conditional possibility $p\left(\widehat{\mathcal{Y}}_i|X_i\right)$; that is:

$$\mathcal{L}_{SER} = -\log(p\left(\widehat{\mathcal{Y}}_i|X_i\right)). \tag{5}$$

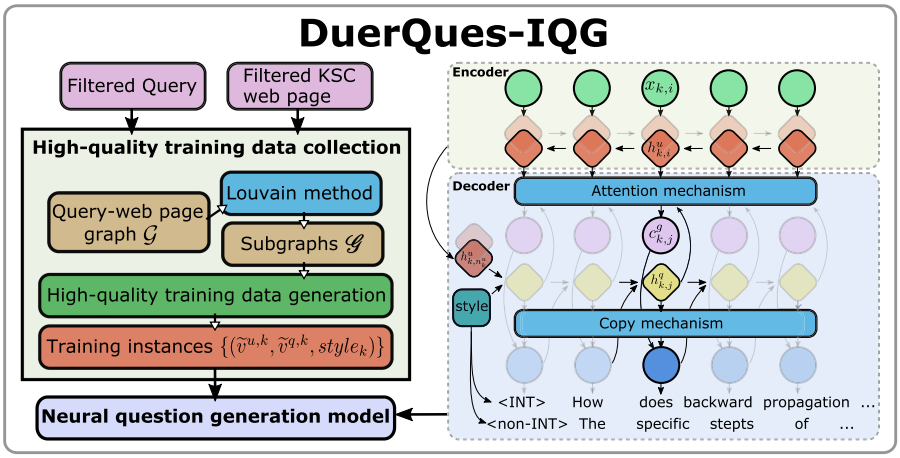

Fig. 4. A schematic diagram of interview question generation (DuerQues-IQG).

Along this line, we can recognize skill entities in the corpus. Moreover, we indicate that each skill entity mentioned in the following is always considered as a single word.[2]

3.2 Interview Question Generation

After recognizing the skill entities in both the query and web page titles, we remove all those data that contain no skill entity. We introduce how to leverage the filtered data to generate interview questions. As mentioned before, we consider that high-quality queries can be regarded as natural interview questions about the knowledge of some professional skills contained in the corresponding web page, such as "*How does backward propagation work in a CNN?*" in Figure 1. Along this line, we design a neural interview question generation method (DuerQues-IQG), which mainly consists of two components, namely, (1) high-quality training data collection and (2) neural question generation. Figure 4 shows an illustration of our method.

High-quality Training Data Collection. To collect high-quality training data, we first group the related queries and web pages based on their relevance with different professional knowledge. This can help to prevent generating duplicate or similar interview questions, and improve the performance of the subsequent question recommendation module. Formally, we denote all the queries and web pages as $\mathcal{V}^q = \{v_1^q, v_2^q, ..\}$ and $\mathcal{V}^u = \{v_1^u, v_2^u, ...\}$, respectively. Then, we construct an edge set $\mathcal{E} \subset \mathcal{V}^q \times \mathcal{V}^u$ based on click-through data. Each $e_{i,j} \in \mathcal{E}$ denotes that there exists a click behavior between query v_i^q and web page v_j^u. In this way, we can obtain a query-web page graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where the node set $\mathcal{V}$ contains two kinds of nodes: $\mathcal{V}^q$ and $\mathcal{V}^u$. Here, we use $\mathcal{W} \in \mathbb{N}^{|\mathcal{V}^q| \times |\mathcal{V}^u|}$ to denote the edge weights, where $|\mathcal{V}^q|$ and $|\mathcal{V}^u|$ denote the numbers of elements in $\mathcal{V}^q$ and $\mathcal{V}^u$, respectively, and $\mathcal{W}_{i,j}$ equals the count of click behavior between v_i^q and v_j^u. To group the related query and web page nodes in $\mathcal{G}$, we leverage a traditional community detection algorithm, namely, the Louvain method [40]. Then, we can collect a set of subgraphs $\mathscr{G} = \{\mathcal{G}_1, \mathcal{G}_2, ...\}$, where each $\mathcal{G}_k = (\mathcal{V}_k, \mathcal{E}_k)$ contains at least one web page node. We use $\mathcal{V}_k^q$ and $\mathcal{V}_k^u$ to denote query nodes and web page nodes in $\mathcal{V}_k$, respectively. Our purpose is to generate an interview question for each subgraph.

[2]Given any corpus that requires word segmentation, we use the Chinese word segmentor with a predefined skill dictionary to ensure that the recognized skill entities can be correctly segmented.

Afterward, we introduce a heuristic algorithm to construct high-quality training data from the obtained subgraph set $\mathscr{G}$ for the interview question generation model. Specifically, for each subgraph $\mathcal{G}_k = (\mathcal{V}_k, \mathcal{E}_k)$, we first score all the web page nodes. Here, web pages with more clicks are more likely to have high-quality content, and web pages containing more skills can be more informative. Both of these characteristics can benefit subsequent question generation and skill-oriented question recommendation. Therefore, we calculate the score of a web page node v_j^u by:

$$score\left(v_j^u\right) = \alpha_u f\left(\sum_{v_i^q \in \mathcal{V}_k^q} \mathcal{W}_{i,j}^k\right) + (1-\alpha_u) f\left(C_j^u, \left\{C_l^u\right\}_{v_l^u \in \mathcal{V}_k^u}\right), \tag{6}$$

where $\mathcal{W}^k$ denotes the weight matrix of $\mathcal{G}_k$ that is calculated similarly to $\mathcal{W}$. C_j^u denotes the number of unique skill entities in the title of a web page v_j^u. $f(x_i, x) = \frac{x_i - min(x)}{max(x) - min(x)}, x = \{x_1, x_2, ...\}$ is the normalization function. We use $\widetilde{v}^{u,k}$ to denote the web page that has the highest score in $\mathcal{G}_k$.

In addition, the title of $\widetilde{v}^{u,k}$ already contains the primary information needed to generate the interview question. Then, we aim to collect a corresponding high-quality query that can assist us in learning how to ask this question in a natural way. To this end, we score all the query nodes according to the total click count in $\mathcal{G}_k$, the number of unique skill entities in the query, and the string similarity between the query and the title of $\widetilde{v}^{u,k}$ as follows:

$$score\left(v_i^q\right) = \alpha_{q,1} f\left(\sum_{v_j^u \in \mathcal{V}_k^u} \mathcal{W}_{i,j}^k\right) + \alpha_{q,2} f\left(C_i^q, \left\{C_l^q\right\}_{v_l^q \in \mathcal{V}_k^q}\right) + (1-\alpha_{q,1}-\alpha_{q,2}) g\left(\widetilde{v}^{u,k}, v_i^q\right), \tag{7}$$

where C_i^q denotes the unique skill entity count of query v_i^q, and $g(\cdot,\cdot) \in [0,1]$ is a string similarity function. In this work, we employ the Ratcliff-Obershelp algorithm.[3] Compared with Equation (6), we further consider the content similarity between the web page title and query, which can promote the query of the top ranking samples that contain more information and reduce the difficulty of the subsequent question generation task. Next, we select the query with the highest score whose string similarity must be greater than the threshold t^q. If such a query exists, we denote it as $\widetilde{v}^{q,k}$ in $\mathcal{G}_k$. Moreover, we leverage the rule-based method to determine whether the query is an interrogative sentence. For instance, a query containing interrogative words such as "How" and "What" can be regarded as interrogative. Finally, we can collect high-quality interview question generation training data, which are represented as a set of triples $(\widetilde{v}^{u,k}, \widetilde{v}^{q,k}, style_k)$ for each subgraph $\mathcal{G}_k$, where $style_k \in \{0,1\}$ indicates whether query $\widetilde{v}^{q,k}$ is an interrogative sentence. Please note that for each subgraph $\mathcal{G}_k$, we only select $\widetilde{v}^{u,k}$ and $\widetilde{v}^{q,k}$ with the highest $score(v_\cdot^u)$ and $score(v_\cdot^q)$, respectively.

Neural Question Generation. With the collected training instances, we can formalize the interview question generation task for all $\mathcal{G}_k \in \mathscr{G}$. Specifically, given a web page title $\widetilde{v}^{u,k}$, we want to train a model $\mathcal{M}_g$ that can generate fluent and rational $\widetilde{v}^{q,k}$ in an interrogative style. Although we can directly leverage some state-of-the-art neural sequence-to-sequence models, such as [5, 72, 81], based on the training instances containing the interrogative query $\widetilde{v}^{q,k}$, i.e., $style_k = 1$, the small size of the training data will limit the robustness of $\mathcal{M}_g$. Inspired by the idea of knowledge transfer [90, 109], here we design a novel encoder-decoder model with a new learning mechanism, which can leverage the rest of the high-quality training instances, i.e., $style_k = 0$. Figure 4 shows an illustration of our model.

[3]We follow https://xlinux.nist.gov/dads/HTML/ratcliffObershelp.html to calculate $g(\cdot,\cdot)$.

To be specific, for each $(\widetilde{v}^{u,k}, \widetilde{v}^{q,k}, style_k)$, we first use the Chinese word segmentor to split $\widetilde{v}^{u,k}$ and $\widetilde{v}^{q,k}$ and ensure that the previously recognized skill entities can be segmented correctly; that is:

$$\begin{aligned} \widetilde{v}^{u,k} &= \left\{ w^u_{k,1}, w^u_{k,2}, \ldots, w^u_{k,n^u_k} \right\}, \\ \widetilde{v}^{q,k} &= \left\{ w^q_{k,1}, w^q_{k,2}, \ldots, w^q_{k,n^q_k} \right\}, \end{aligned} \tag{8}$$

where $w^u_{k,i}$ and $w^q_{k,i}$ are the i-th words in the web page title $\widetilde{v}^{u,k}$ and query $\widetilde{v}^{q,k}$, respectively, and n^u_k and n^q_k are the corresponding word sequence lengths. Moreover, we use the skill label sequence $\{l^u_{k,1}, \ldots, l^u_{k,n^u_k}\}$ to represent whether each word $w^u_{k,i}$ is a skill entity word or not, where $l^u_{k,i} \in \{0, 1\}$. Then, we use an embedding layer to transform each word $w^u_{k,i}$ in web page title $\widetilde{v}^{u,k}$ into an embedding vector $f^w_e(w^u_{k,i})$, where $f^w_e(\cdot)$ denotes the word embedding lookup table. We concatenate the embedding vector and the one-hot feature vector of the skill label $l^u_{k,i}$ to form the input of an encoding layer in our model as follows:

$$x_{k,i} = \left[f^w_e(w^u_{k,i}); onehot(l^u_{k,i}) \right]. \tag{9}$$

Then, we employ a BiLSTM layer to encode the input sequence $\{x_{k,1}, \ldots, x_{k,n^u_k}\}$; that is:

$$h^u_{k,i} = BiLSTM\left(x_{k,i}, h^u_{k,i-1}\right). \tag{10}$$

Next, we construct a decoder to generate each word in $\widetilde{v}^{q,k}$, which is built upon a unidirectional LSTM layer; that is:

$$h^q_{k,j} = LSTM\left(f^w_e\left(w^q_{k,j-1}\right), h^q_{k,j-1} \right). \tag{11}$$

Different from traditional sequence-to-sequence models, for leveraging the training instances for both interrogative and non-interrogative queries, here we add an "interrogative" token before the head of the query word sequence; that is, $w^q_{k,0}$ = <*INT*> or <*non-INT*> for $style_k$ is 1 or 0, respectively. Furthermore, we initialize the first time step of hidden states $h^q_{k,0}$ by concatenating the encoding vector h^u_{k,n^u_k} and the one-hot feature vector of $style_k$. We also employ an attention mechanism [5] to capture the important context feature from the encoder states as follows:

$$g_{k,i,j} = v^\top \tanh\left(W^g_1 \left[h^u_{k,i}; h^q_{k,j}\right] + b^g_1\right), \quad \alpha^g_{k,i,j} = \frac{exp(g_{k,i,j})}{\sum_{l=1}^{n^u_k} exp(g_{k,l,j})}, \quad c^g_{k,j} = \sum_{l=1}^{n^u_k} \alpha^g_{k,l,j} h^u_{k,l}, \tag{12}$$

where v, W^g_1 and b^g_1 are trainable parameters. Then, we can predict the j-th word $w^q_{k,j}$ by:

$$p\left(w^q_{k,j} | w^q_{k,<j}\right) = \text{softmax}\left(W^g_2 \left[c^g_{k,j}; h^q_{k,j}\right] + b^g_2\right), \tag{13}$$

where W^g_2 and b^g_2 are trainable parameters and $w^q_{k,<j}$ denotes the sentence $\{w^q_{k,0}, w^q_{k,1}, \ldots, w^q_{k,j-1}\}$. Moreover, for enhancing the performance and handling the **out-of-vocabulary (OOV)** problem, we also employ copy and coverage mechanisms [72].

Along this line, the generation model $\mathcal{M}_g$ can be learned by minimizing the following cross-entropy loss function for each generated sentence:

$$\mathcal{L}_{QG} = -\sum_{j=0}^{n^q_k} \log\left(p\left(w^q_{k,j} | \widetilde{v}^{u,k}, style_k\right) \right). \tag{14}$$

In addition, we leverage the teacher forcing algorithm [88] for our interview question generation model during the training process. Specifically, when calculating each hidden state $h^q_{k,j}$ at each

sequence decoding time step, we use the previous word $w^q_{k,j-1}$ from the ground truth in the training process and the predicted word in the test.[4] Moreover, during the generation process in the test phase, we utilize a beam search to improve the performance. We also set $w^q_{k,0}$ to <*INT*> and $style_k$ to 1, as our purpose is to generate interview questions in an interrogative style.

Finally, we can utilize our model to generate interview question sentences for each $\mathcal{G}_k$ that failed to obtain high-quality questions in previous data collection steps. Therefore, we can construct our skill-oriented question bank.

4 SKILL-ORIENTED RECOMMENDATION

In this section, we introduce our skill-oriented recommender system, which can effectively assist an interviewer in designing and selecting interview questions. Our system consists of two components: Skill entity suggestion (DuerQues-SES) and interview question recommendation (DuerQues-IQR). Specifically, in skill entity suggestion, we recommend relevant skill entities based on the skill entities retrieved by interviewers to help the interviewers with question design and to efficiently use the question recommender. For instance, we can recommend skill entities "*CNN*" or "*RNN*" to interviewers when they want to design a question about skill entity "*Text Classification*". Furthermore, interview question recommendation is the core function of our system. When interviewers want to ask interviewees about one or more skills, our system can recommend a set of suitable candidate interview questions from our question bank.

4.1 Skill Entity Suggestion

To suggest related skills for interviewers, we first construct a skill-graph $\mathcal{G}^s = (\mathcal{V}^s, \mathcal{E}^s)$ based on the query-web page graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ and try to exploit the skill entity co-occurrences in $\mathcal{G}^s$. Specifically, each node $v^s_i \in \mathcal{V}^s$ denotes a skill entity. There are three kinds of edges in $\mathcal{G}^s$, including $\mathcal{E}^{qq}$, $\mathcal{E}^{qu}$, and $\mathcal{E}^{uu}$. Each edge $e^{qq}_{i,j} \in \mathcal{E}^{qq}$ indicates the existence of a query $v^q_m \in \mathcal{V}^q$ that contains the skill entities v^s_i and v^s_j. Similarly, each edge $e^{uu}_{i,j} \in \mathcal{E}^{uu}$ represents the existence of a web page title $v^u_n \in \mathcal{V}^u$ that contains skill entities v^s_i and v^s_j. We use $e^{qu}_{i,j} \in \mathcal{E}^{qu}$ to denote that there exists a skill entity v^s_i contained in query v^q_m and a skill entity v^s_j contained in the correspondingly clicked web page title v^u_n, i.e., $e_{m,n} \in \mathcal{E}$.

Intuitively, when an interviewer searches for a skill entity v^s_i, we can directly recommend a set of skill entities that co-occur with it in the queries, that is, v^s_i's first-order neighbors based on $\mathcal{E}^{qq}$. However, considering the small amount of query data about some niche skill entities and the possible errors in the skill entity recognition, the edges $\mathcal{E}^{qq}$ cannot cover all possible related skill entity pairs. To this end, we formalize a link prediction task. That is, given a skill-graph $\mathcal{G}^s$ and a skill entity v^s_i, we want to obtain all the possible edges between v^s_i and other skill entity nodes, which should belong to $\mathcal{E}^{qq}$. Here, we employ a RGCN [71], which is a recently proposed neural graph model for modeling heterogeneous graphs. Specifically, we first use an embedding layer to obtain the input features for each skill entity node, that is, $h^s_{0,i} = f^s_e(v^s_i)$, where $f^s_e(\cdot)$ is the lookup table of the embedding layer initialized with pretrained word embedding from our query and title textual data. Then, the graph embedding for each node is calculated by messages passed from other skill entity nodes at each RGCN layer l; that is:

$$h^s_{l+1,i} = \sigma\left(\sum_{r\in\{qq,qu,uu\}}\sum_{j\in\mathcal{N}^r_i}\frac{1}{c^s_{i,r}}W^s_{r,l}h^s_{l,j} + W^s_{0,l}h^s_{l,i}\right), \tag{15}$$

[4]Except for the first word, which is the manually added token, i.e., <*INT*>.

where r is the edge type, which corresponds to $\mathcal{E}^{qq}$, $\mathcal{E}^{qu}$ and $\mathcal{E}^{uu}$. $\mathcal{N}_i^r$ denotes the neighbors of v_i^s on the edge of category r, and $c_{i,r}^s$ is the normalization factor of v_i^s and r, which can be learned or chosen in advance. $W_{r,l}^s$ and $W_{0,l}^s$ are the trainable parameters of the l-th RGCN layer with edge type r and a self-loop, respectively. σ is the activation function, such as $\mathrm{ReLU}(\cdot) = \max(0, \cdot)$. We employ two RGCN layers to propagate more information from the whole graph, not just the direct neighbors.

Finally, we follow the solution in [71] to use the DistMult factorization as the scoring function:

$$f_{dm}\left(v_i^s, v_j^s\right) = {h_{l,i}^s}^\top W^{dm} h_{l,j}^s, \tag{16}$$

where W^{dm} is a trainable parameter. We leverage a cross-entropy loss to train our model; that is:

$$\mathcal{L}_{SS} = -\frac{1}{(1+\beta)|\mathcal{E}^{qq}|} \sum_{(v_i^s, v_j^s, y_{i,j}^s) \in \mathcal{T}} y_{i,j}^s \log f_{sig}\left(f_{dm}(v_i^s, v_j^s)\right) + (1 - y_{i,j}^s) \log\left(1 - f_{sig}\left(f_{dm}(v_i^s, v_j^s)\right)\right), \tag{17}$$

where β is the ratio of negative sampling, $y_{i,j}^s$ denotes whether there exists an edge in $\mathcal{E}^{qq}$ between v_i^s and v_j^s, $\mathcal{T}$ denotes all the triples collected from user behavior and negative sampling, and f_{sig} is the sigmoid function.

4.2 Interview Question Recommendation

In question recommendation, we focus on how to recommend relevant interview questions to interviewers from our question bank when they query a set of skill entities. Formally, we define that the queried skill entity set is $U_i = \{v_{z_1^u}^s, v_{z_2^u}^s, \ldots, v_{z_{|U_i|}^u}^s\}$, where $\{z_1^u, \ldots, z_{|U_i|}^u\}$ is an index sequence of the skill-graph nodes $\mathcal{V}^s$ and $|U_i|$ denotes the number of unique skill entities in U_i. Each interview question collected or generated from a query-web page subgraph $\mathcal{G}_j$ in our question bank is denoted as Q_j. We denote the observed interaction matrix based on the query-web page graph $\mathcal{G}$ and subgraphs $\mathcal{G}$ between the queried skill entity sets and questions as $R \in \mathbb{R}^{|\mathcal{U}| \times |\mathcal{Q}|}$. Specifically, $R_{i,j} = 1$ denotes that there exists an edge $e_{m,n} \in \mathcal{E}$ such that (1) the corresponding recognized unique skill entity set of query v_m^q equals U_i and (2) v_n^u belongs to subgraph $\mathcal{G}_j$. Otherwise, $R_{i,j} = 0$. $\mathcal{U}$ and $\mathcal{Q}$ denote all the obtained queried skill entity sets and the questions from the original query-web page graph $\mathcal{G}$, respectively.

Intuitively, we can recommend interview questions based on R when the given U_i can be found in $\mathcal{U}$. However, because query data cannot contain all possible skill entity sets, this is insufficient to recommend questions to an unseen $U_i \notin \mathcal{U}$. To this end, we define an interview question recommendation task; that is, *given the historical queried skill entity sets $\mathcal{U}$, all the interview questions $\mathcal{Q}$, and the obtained interaction matrix R, our goal is to learn a model $\mathcal{M}_r$ that can recommend suitable questions for an unseen $U_i \notin \mathcal{U}$.*

To achieve this task, we propose a graph-enhanced recommendation algorithm. Figure 5 shows an illustration of our method. Specifically, we first exploit two kinds of features for each question Q_j. Here, we collect the corresponding top-1 web page title $\widetilde{v}^{u,j} = \{w_{j,1}^u, \ldots, w_{j,n_j^u}^u\}$ from the query-web page subgraph $\mathcal{G}_j$, as mentioned in Section 3.2, where n_j^u denotes the sequence length of $\widetilde{v}^{u,j}$. We also denote the combined unique skill entity sets as $S_j = \{v_{z_1^s}^s, \ldots, v_{z_{|S_j|}^s}^s\}$, and the entities are recognized from all the web page titles in $\mathcal{G}_j$, where $\{z_1^s, \ldots, z_{|S_j|}^s\}$ is an index sequence of $\mathcal{V}^s$ and $|S_j|$ denotes the number of unique skill entities in S_j. Next, for extracting the semantic features for U_i, we employ a word embedding layer for each word $v_{z_k^u}^s$; i.e., $x_{i,k}^u = f_e^r(v_{z_k^u}^s)$, where $f_e^r(\cdot)$ is the lookup table of the embedding layer. Similarly, we can obtain the word vector $x_{j,k}^t = f_e^r(w_{j,k}^u)$ for each word $w_{j,k}^u$ in $\widetilde{v}^{u,j}$ and $x_{j,k}^s = f_e^r(v_{z_k^s}^s)$ for each word $v_{z_k^s}^s$ in S_j.

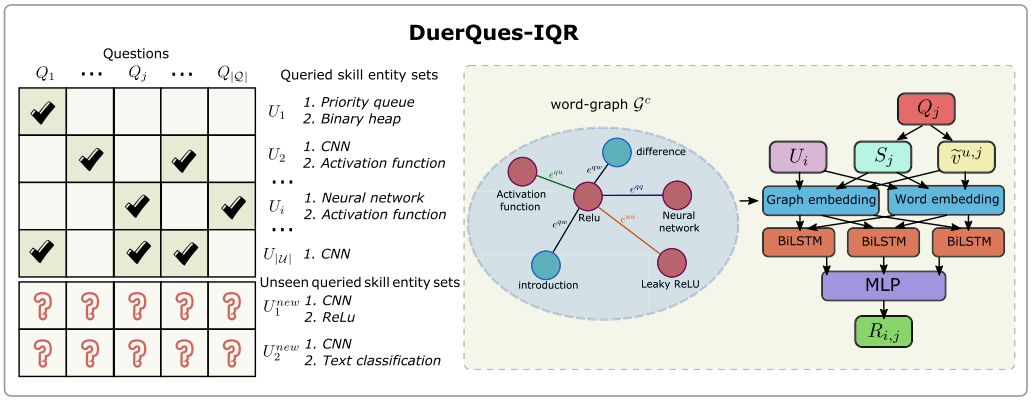

Fig. 5. A schematic diagram of interview question recommendation (DuerQues-IQR).

Then, we leverage an RGCN to model the word-graph structure information based on the word co-occurrence relationship, which enhances the representations of U_i, $\widetilde{v}^{u,j}$ and S_j. Different from the skill-graph $\mathcal{G}^s$ introduced above, we expand the original node set $\mathcal{V}^s$ to cover the words $\mathcal{V}^w$ in web page titles $\{\widetilde{v}^{u,1}, ...\}$; i.e., $\mathcal{V}^c = \mathcal{V}^s \bigcup \mathcal{V}^w$, where $\mathcal{V}^w$ is the set of word nodes appearing in $\{\widetilde{v}^{u,1}, ...\}$. We also add $\mathcal{E}^{qw}$, where edge $e^{qw}_{i,j} \in \mathcal{E}^{qw}$ if we can find an edge $e_{m,n} \in \mathcal{E}$ so that the query $v^q_m \in \mathcal{V}^q$ contains skill entity v^s_i and the web page title v^u_n contains word v^w_j. Therefore, we can collect a word-graph $\mathcal{G}^c = (\mathcal{V}^c, \mathcal{E}^c)$, where $\mathcal{E}^c$ contains four kinds of edges, i.e., $\mathcal{E}^{qq}$, $\mathcal{E}^{qu}$, $\mathcal{E}^{uu}$, and $\mathcal{E}^{qw}$. Then, similar to Equation (17) ($r \in \{qq, qu, uu, qw\}$), we can obtain the graph embedding $h^c_{l,i}$ for each node in $\mathcal{G}^r$ at the l-th RGCN layer.

Then, we employ three BiLSTM layers to obtain the final representation of each word in U_i, $\widetilde{v}^{u,j}$ and S_j; that is:

$$\begin{aligned} h^{cu}_{i,k} &= BiLSTM\left([x^u_{i,k}; x^{cu}_{i,k}], h^{cu}_{i,k-1}\right), \\ h^{ct}_{j,k} &= BiLSTM\left([x^t_{j,k}; x^{ct}_{i,k}], h^{ct}_{j,k-1}\right), \\ h^{cs}_{j,k} &= BiLSTM\left([x^s_{j,k}; x^{cs}_{j,k}], h^{cs}_{j,k-1}\right), \end{aligned} \tag{18}$$

where $x^{cu}_{i,k}$, $x^{ct}_{j,k}$, and $x^{cs}_{j,k}$ denote the graph embedding of the k-th words in U_i, $\widetilde{v}^{u,j}$, and S_j, respectively, by leveraging the $h^c_{l,.}$. Next, we use the hidden states in the last time step of sequences U_i, $\widetilde{v}^{u,j}$ and S_j to obtain their corresponding features and concatenate them as the input of a **multi-layer perceptron network (MLP)** to obtain the score of a queried skill entity set U_i and question Q_j pair; that is:

$$f_g(U_i, Q_j) = MLP\left([h^{cu}_{i,|U_i|}; h^{ct}_{j,n^u_j}; h^{cs}_{j,|S_j|}]\right), \tag{19}$$

where the activation functions for the nodes of each fully connected layer, except those of the last layer, are ReLU activation functions, and the last layer is connected with a sigmoid function.

Finally, we leverage a cross-entropy loss to train our model:

$$\mathcal{L}_{QR} = -\frac{1}{(1+\beta')\sum_{i,j} R_{i,j}} \sum_{(U_i, Q_j, R_{i,j}) \in \mathcal{T}'} R_{i,j} \log f_g(U_i, Q_j) + (1 - R_{i,j}) \log\Big(1 - f_g(U_i, Q_j)\Big), \tag{20}$$

where β' denotes the ratio of negative sampling and $\mathcal{T}$ denotes the pairs of positive and sampled negative instances.

Moreover, we use a skill-based retrieval method to initially generate a subset Q_c of questions Q during the training and test processes, which can improve the algorithm's inference efficiency when it is implemented in our system. Specifically, we build a question dictionary of the skill entity index, where we associate the skill entity v_k^s and Q_j when v_k^s is included in S_j or any U_i satisfying $R_{i,j} = 1$. In addition, we leverage the link prediction method introduced in Section 4.1 to supplement some edges in $\mathcal{E}^{qu}$ to prevent some skill entities from containing too few candidate questions. In this way, Q_c is the union of candidate questions corresponding to all the skill entities in a queried skill entity set.

5 EXPERIMENTAL RESULTS

In this section, we present the experimental results, which demonstrate the effectiveness of each component of our system.

5.1 Evaluation of DuerQues-SER

Datasets. We collected over 10 million click-through data points (queries, clicked URLs that belong to well-known KSCs, and web page titles) from the query logs of a famous Chinese search engine, ranging from July to September 2020. We obtained nearly 5.7 million unique queries and 1.3 million KSC URLs. Then, we filtered out the queries that appeared less than ten times and the KSC web pages that appeared less than three times in the collected click-through data. As a result, we collected 253,589 queries and 132,959 KSC web pages.

Based on these data, we constructed two datasets to evaluate our method, i.e., query and KSC title datasets.[5] Furthermore, we conducted experiments on a public benchmark dataset, i.e., Weibo NER [58], to demonstrate our model's effectiveness in general NER tasks. Please refer to our appendix for details. To be specific, as stated above, we first collected the sorted candidate vocabulary $\mathcal{V}$ for both datasets. In our experiment, we set $K = 10{,}000$ when annotating the top-K words in both vocabularies. Since these two vocabularies have considerable overlap, we merged the candidate words before the annotation and obtained 2,378 skill entities and 10,891 non-skill entities. Then, we generated distantly supervised training instances for both datasets. Moreover, we randomly selected 1,500 instances for both datasets and invited domain experts to label them. Afterward, we selected 1,000 instances to test the performance of our model and used the remaining 500 instances for validation and tuning parameters.

Training Details. In our experiment, we used the pretrained Chinese Giga character and character bigram embeddings released by Zhang and Yang to initialize the parameters in $f_e^c(\cdot)$ and $f_e^b(\cdot)$, respectively [102]. We utilized the Chinese pretrained language model BERT-WWM released by Cui et al. to initialize $f_e^{BERT}(\cdot)$ [15]. We set the dimension of one-hot feature $x_{i,j}^s$ for each character as 13, which represents whether the previous/current/next character is an uppercase/lowercase English alphabet, number, space, or none of these. For the SoftLexicon feature $x_{i,j}^l$, we followed the settings in Peng et al. to concatenate $x_{i,j}^{l,B}$, $x_{i,j}^{l,M}$, $x_{i,j}^{l,E}$, and $x_{i,j}^{l,S}$, which are calculated as the weighted sum of lexicon word vectors in the "BMES" words sets for each character $c_{i,j}$, respectively [57]. In contrast, we used the pretrained word embedding released by Song et al. to initialize the parameters for calculating the word vectors and create the lexicon vocabulary [77]. In particular, we merged our labeled skill and non-skill entities into this lexicon. We also set the non-skill sampling ratio α to 0.4 and 0.6 for these two datasets, respectively. Moreover, we set the dimension of the BiLSTM hidden state to 200, the dropout rate to 0.3, and the batch size to 128. We adopted Adam with L2 regularization for optimization and set the learning rate to $2e^{-5}$.

[5] After calculating features $x_{i,j}^s$, we converted all data to lowercase characters.

Table 2. Performance of Skill Entity Recognition

Datasets	Methods	Precision	Recall	F1
Query	SUL	0.8855	0.9393	0.9116
	SUL+	**0.9124**	0.9444	0.9281
	DuerQues-SER	0.9060	0.9625	**0.9334**
	- w/o non-skill entity annotation	0.8975	**0.9643**	0.9297
	- w/o non-skill entity sampling	0.9004	0.9579	0.9282
	- w/o one-hot feature	0.9039	0.9573	0.9298
	- w/o SoftLexicon feature	0.8961	0.9584	0.9262
	- w/o BERT embedding	0.9067	0.9511	0.9284
KSC title	SUL	0.8784	0.8441	0.8609
	SUL+	**0.8962**	0.8510	0.8730
	DuerQues-SER	0.8732	**0.8977**	**0.8854**
	- w/o non-skill entity annotation	0.8716	0.8848	0.8782
	- w/o non-skill entity sampling	0.8799	0.8693	0.8746
	- w/o one-hot feature	0.8722	0.8849	0.8785
	- w/o SoftLexicon feature	0.8787	0.8716	0.8751
	- w/o BERT embedding	0.8863	0.8722	0.8793

Methods. Here, we chose several baselines and DuerQues-SER's variants for comparison.

- **SUL.** SUL is a recently proposed state-of-the-art Chinese NER approach, which designs a weighted word embedding feature from a lexicon feature, namely, the SoftLexicon feature [57]. To fit their training process, we transferred all labels from U to O in the training data.
- **SUL+.** We enhanced SUL using the same input embeddings and features as our proposed model for a fair comparison with our model.
- **DuerQues-SER w/o non-skill entity annotation.** This is a variant of our model that only uses non-skill entity sampling to create distantly supervised training data and applies partial-CRF to handle the unlabeled characters. Moreover, it can be regarded as a variant of the method in [11], without external knowledge from Wikipedia.
- **DuerQues-SER w/o non-skill sampling.** This is a variant of our model that only uses non-skill entities in $\mathcal{V}^-$ to obtain distantly supervised training data.
- **DuerQues-SER w/o one-hot feature.** This is a variant of our model that removes one-hot feature vector $x^s_{i,j}$ from the inputs.
- **DuerQues-SER w/o SoftLexicon feature.** This is a variant of our model that removes SoftLexicon feature vector $x^l_{i,j}$ from the inputs.
- **DuerQues-SER w/o BERT feature.** This is a variant of our model that removes BERT embedding $x^{BERT}_{i,j}$ from the inputs.

Results. Table 2 shows the overall performance of our DuerQues-SER model, its variants, and the baseline models. We can observe that our model achieves the highest F1-value for both query data and KSC web page title data since the partial-CRF layer significantly increases the recall score. Furthermore, we can observe that if we remove each kind of input feature, the F1-value performance decreases. Moreover, removing the pretrained language model, i.e., BERT, does not decrease the effect of these two datasets much, which may be due to the sufficient number of training instances. The performance can be significantly improved on other datasets with less training data by employing the BERT model. In addition, we conducted a standard Student's t-test

on our models and each baseline. We find that our model outperforms the baselines in terms of F1-value with statistical significance ($p < 0.05$). Please refer to the experiments on the Weibo NER dataset in our appendix for details. Finally, we inferred all the queries' and web page titles' skill entities and collected 218,728 queries and 85,821 web page titles by removing the instances with no skill. This produced 3,755 unique skill entities.

5.2 Evaluation of DuerQues-IQG

Training Data Collection. As mentioned in Section 3.2, we first constructed a query-web page graph. Specifically, it comprises 218,728 queries and 85,821 web pages represented as nodes. Based on the click-through data, we obtained 316,845 edges between the query and web page nodes. Then, with the Louvain method, we collected 53,998 subgraphs, each containing at least one web page node. By setting $\alpha_u = 0.7$, $\alpha_{q,1} = 0.1$, $\alpha_{q,2} = 0.1$, and the string similarity threshold $t^q = 0.7$, we collected a total of 17,605 training triples of the form $(\widetilde{v}^{u,k}, \widetilde{v}^{q,k}, style_k)$. Among them, 1,398 instances contain an interrogative query $\widetilde{v}^{q,k}$, i.e., $style_k = 1$. Moreover, to verify the performance of our interview question generation model, we randomly selected 200 instances as the test data and 100 instances for tuning parameters, where each instance satisfies $style_k = 1$.

Training Details. In DuerQues-IQG, the word embedding layer $f_e^w(\cdot)$ transforms words $w_{k,j}^u$ and $w_{k,j}^q$ into vectors of size 200. More specifically, we used the skip-gram model to pretrain the word embeddings from all the query and web page title data. Based on the word embeddings, we initialized the embedding lookup table. The encoding layer is implemented with a BiLSTM layer with a hidden size of 256 for each LSTM layer. The decoding layer is implemented by a unidirectional LSTM layer with 256 hidden units. Moreover, we set the batch size to 32 and sampled the same numbers of instances where $style_k = 0$ and $style_k = 1$ in each batch. We adopted Adam with L2 regularization for optimization and set the learning rate to $1e^{-3}$. During the generation process in the test phase, we used the beam search algorithm with a beam size of 4.

Methods and Metrics. Here, we chose several baseline methods and one DuerQues-IQG variant for comparison.

- **Seq2Seq-Attn.** It is a classic sequence-to-sequence model used to achieve neural machine translation [49]. We chose the concat-based function to calculate the attention score in our implementation.
- **PGNet.** It is a sequence-to-sequence model, and it applies the pointer network and coverage mechanism to manage the abstractive text summarization problem [72]. Here we used all the training instances with $style_k \in \{0, 1\}$ to train this model.
- **PGNet w/o filtering instances by t^q.** It is a variant of PGNet, and it does not filter the query data with string similarity less than the threshold t^q.
- **DuerQues-IQG w/o $style_k = 0$ instances.** It is a variant of our model that exclusively uses high-quality training instances containing interrogative queries.

To evaluate the proposed method, we chose standard **recall-oriented understudy for gisting evaluation (ROUGE)** metrics, which are popular metrics for many text generation tasks, such as text summarization [72] and question generation [19]. Specifically, we utilized ROUGE-1, ROUGE-2, and ROUGE-L, which measure the unigram-overlap, bigram-overlap, and **longest common subsequence (LCS)**-based statistics [46] between $\widetilde{v}^{q,k}$ and the generated question. Furthermore, we invited two human resources experts to score each generated question from 1 to 5 from two dimensions.

- **Fluency:** Does the generated result read smoothly and fluently?
- **Validity:** Is the generated result a valid and meaningful question for the corresponding web page?

Table 3. Performance of Interview Question Generation

Methods	ROUGE-1	ROUGE-2	ROUGE-L	Fluency	Validity
Seq2Seq-Attn	0.5591	0.2553	0.5427	3.71	1.69
PGNet	0.5677	0.2670	0.5456	**3.80**	1.73
- w/o filtering instances by t^q	0.4959	0.2245	0.4755	3.66	1.47
DuerQues-IQG	**0.5913**	**0.2685**	**0.5549**	3.74	**3.42**
- w/o $style_k = 0$ instances	0.5429	0.2112	0.5121	3.67	3.29

Results. The comparison results are shown in Table 3. According to the results, our DuerQues-IQG model outperforms all the baselines and the variant of the proposed model. Specifically, we found that leveraging our learning method on interrogative and non-interrogative training instances can improve ROUGE-1, ROUGE-2, and ROUGE-L by 8.92%, 27.13%, and 8.35%, respectively. Meanwhile, based on the human evaluation results, we can conclude that the questions generated by our model can be used to examine the relevant skills covered on the KSC web pages. From another perspective, the majority of these generated interview questions are answerable as they are written fluently and reference answers can be found on the KSC web pages. Furthermore, we found that our DuerQues-IQG model had slightly lower fluency than PGNet. This phenomenon may be because DuerQues-IQG samples the interrogative and non-interrogative instances during mini-batch training, resulting in a different training data distribution than PGNet. However, Seq2Seq-Attn, PGNet and its variant perform poorly in *Validity* compared to DuerQues-IQG, which indicates that these methods are not ideal for the interview question generation task.

5.3 Evaluation of DuerQues-SES

Datasets. As mentioned in Section 4.1, we constructed the skill-graph $\mathcal{G}^s$. It consists of 3,755 skill entity nodes, 33,538 edges in $\mathcal{E}^{qq}$, 43,670 edges in $\mathcal{E}^{qu}$, and 39,632 edges in $\mathcal{E}^{uu}$ based on the click-through data. Here, to evaluate the link prediction task on $\mathcal{E}^{qq}$ and $\mathcal{E}^{qu}$, we randomly selected 40%, 50%, and 10% click-through data to create three subsets of $\mathcal{E}^{qq}$ and $\mathcal{E}^{qu}$ as the training, test, and validation edge sets, respectively. Specifically, with the first 40% click-through data, we collected 20,452 edges in $\mathcal{E}^{qq}$ and 26,902 edges in $\mathcal{E}^{qu}$. With the next 50%, after removing the edges that appeared in the training set, we obtained 10,479 edges in $\mathcal{E}^{qq}$ and 13,360 edges in $\mathcal{E}^{qu}$. Similarly, with the rest of the data, we obtained 2,607 and 3,408 edges in $\mathcal{E}^{qq}$ and $\mathcal{E}^{qu}$, respectively, after removing the edges appearing in the training and test sets. Please note that we regard all web page data as observable in the training process. Therefore, the training data contain all the edges in $\mathcal{E}^{uu}$.

Training Details. In our model, we set the word embedding dimension to 200, the number of RGCN layers to 2, and the hidden state's size of RGCN to 500. Moreover, we set the dropout rate in both RGCN layers to 0.4. For model training, we used the Adam optimizer and set the learning rate to 0.01.

Results. Here we compare our DuerQues-SES model with its variants, which correspondingly remove the other types of edges. We evaluated different models based on the **mean reciprocal rank (MRR)** and Hit@K metrics. We chose several traditional link prediction methods as the baselines, including Jaccard [34], Adamic-Adar [2], and Rooted PageRank [8]. The results are shown in Table 4, which demonstrate the effectiveness of using all kinds of edges based on the click-through data on the link prediction tasks on $\mathcal{E}^{qq}$ and $\mathcal{E}^{qu}$. Therefore, we can provide skill entity suggestions and help construct candidate interview questions in subsequent question recommendations more robustly.

Table 4. Performance of Link Prediction on $\mathcal{E}^{qq}$ and $\mathcal{E}^{qu}$

Tasks	Methods	MRR	Hit@1	Hit@3	Hit@10
$\mathcal{E}^{qq}$ Link Prediction	Jaccard	0.0934	0.0539	0.0918	0.1643
	Adamic-Adar	0.1303	0.0604	0.1176	0.2690
	PageRank	0.1183	0.0546	0.1016	0.2404
	DuerQues-SES	**0.1401**	0.0606	**0.1360**	**0.3066**
	- w/o $\mathcal{E}^{qu}$	0.1272	**0.0624**	0.1227	0.2551
	- w/o $\mathcal{E}^{uu}$	0.1362	0.0584	0.1293	0.2981
	- w/o $\mathcal{E}^{uu}$, $\mathcal{E}^{qu}$	0.0805	0.0365	0.0728	0.1630
$\mathcal{E}^{qu}$ Link Prediction	Jaccard	0.1101	0.0669	0.1112	0.1879
	Adamic-Adar	0.1370	**0.0737**	0.1315	0.2898
	PageRank	0.1249	0.0614	0.1194	0.2517
	DuerQues-SES	**0.1550**	0.0715	**0.1605**	**0.3236**
	- w/o $\mathcal{E}^{qq}$	0.1349	0.0627	0.1369	0.2803
	- w/o $\mathcal{E}^{uu}$	0.1308	0.0583	0.1317	0.2734
	- w/o $\mathcal{E}^{qq}$, $\mathcal{E}^{uu}$	0.0852	0.0338	0.0793	0.1823

5.4 Evaluation of DuerQues-IQR

Datasets. As mentioned before, we constructed our skill-oriented question bank with 53,998 interview questions, i.e., $|\mathcal{Q}| = 53{,}998$. With the recognized skill entities in the queries from the click-through data, we obtained 19,383 unique queried skill sets, i.e., $|\mathcal{U}| = 19{,}383$. Afterward, we observed 88,272 interactions between $\mathcal{Q}$ and $\mathcal{U}$ based on the click-through data. To evaluate the performance of our method to recommend interview questions for unseen skill sets, we randomly selected 20% of the skill sets from $\mathcal{U}$ and the corresponding interaction data as the test set. Moreover, we selected another 10% of the skill sets from $\mathcal{U}$ for tuning the model parameters, and the rest of the data was used to train our model.

Training Details. We set the size of the word embedding layer to 200, and it was initialized similarly to the experimental setting of interview question generation. The number of RGCN layers was set to 2, the hidden state size in each RGCN layer was set to 128, and the size of each LSTM layer in BiLSTM was set to 128. Furthermore, we used an MLP with two hidden layers of sizes 128 and 64 and an output layer of size 2. We also set the dropout rate to 0.2 in our model. Moreover, we generated the candidate interview question set for each queried skill set based on the skill-based retrieval method introduced in Section 4.2 to help sample the negative training instances. Then, we found that the average size of candidate interview questions for each queried skill set is 1,545. Moreover, to evaluate the negative sample number sensitivity, we trained our RGCN model by varying this parameter from 1 to 15. Figure 6 shows the AUC, F1, Hit@5, and Hit@10 performance results. We found the best AUC, F1, and Hit@10 results were obtained when randomly sampling ten negative sampling questions for each positive question during training. Therefore, we used this negative sampling rate in our experiments, i.e., $\beta' = 10$. For optimization, we used the Adam optimizer. We set the learning rate to 0.001 and batch size to 256.

Methods. To demonstrate the effectiveness of the proposed DuerQues-IQR method on question recommendation task, we introduce several baseline methods and three variants of DuerQues-IQR for comparison.

- **kNN.** We employed the **k nearest neighbors (kNN)** algorithm as a baseline; it uses the similarities between queried skills and the skills in the candidate questions.

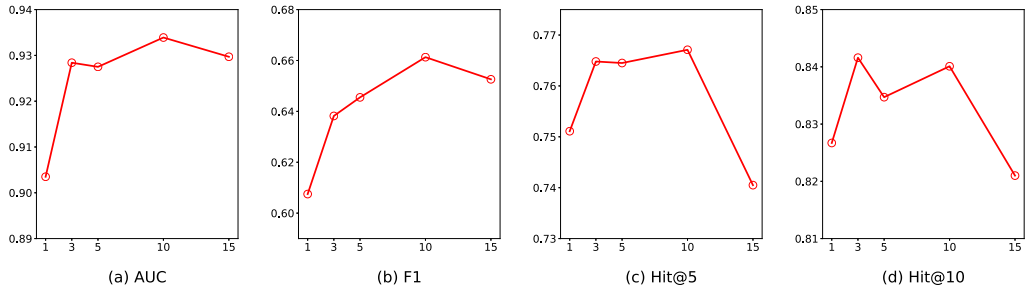

Fig. 6. Impact of negative samples on interview question recommendation.

Table 5. Performance of Interview Question Recommendation

Methods	AUC	F1	Hit@5	Hit@10
kNN	0.6508	0.3212	0.5236	0.6791
BM25	0.7637	-	0.7106	0.7454
LSTM-MLP	0.9210	0.6399	0.7472	0.8182
DuerQues-IQR-RGCN	**0.9339**	**0.6613**	**0.7671**	**0.8401**
- w/o $\mathcal{E}^{qw}$	0.9286	0.6310	0.7643	0.8354
DuerQues-IQR-GAT	0.9316	0.6400	0.7575	0.8313
- w/o $\mathcal{E}^{qw}$	0.9253	0.6271	0.7490	0.8305

- **BM25.** We selected standard Okapi BM25 [69] as a baseline; it is a popular probabilistic relevance model.
- **BiLSTM-MLP.** We used the BiLSTM models to generate the semantic representation of the queried skill set U_i, candidate web page title $\widetilde{v}^{u,j}$ and combined unique skill set S_j. Then, we leveraged a 2-layer MLP network to predict the ranking score among all candidate web pages.
- **DuerQues-IQR-RGCN, w/o $\mathcal{E}^{qw}$.** We removed the edges in $\mathcal{E}^{qw}$ when using RGCN to calculate the node embedding and only used $x^t_{j,k}$ as the input of BiLSTM to exploit the web page title feature $\widetilde{v}^{u,j}$.
- **DuerQues-IQR-GAT.** We replaced the RGCN with 2-layer GAT in our model to obtain graph embeddings, where the types of the edges are not distinguished.
- **DuerQues-IQR-GAT, w/o $\mathcal{E}^{qw}$.** We removed edges in $\mathcal{E}^{qw}$ when using GAT to calculate the node embedding, and only fed $x^t_{j,k}$ as the input of BiLSTM to exploit the web page title feature $\widetilde{v}^{u,j}$.

Results. To evaluate the performance of our method, we measured the classification correctness for each queried skill set U_i and candidate question Q_j pair. The metrics include the F1-score and **receiver operating characteristic (ROC)** AUC. We also employed the ranking metrics, i.e., Hit@K, by sorting the output score of the positive instances found in R and the negative instances sampled from the candidate interview question set. During testing, we randomly sampled 200 negative instances for each queried skill set to calculate all the metrics. The final results are shown in Table 5. We found that DuerQues-IQR-RGCN can achieve the best performance in terms of both the classification and ranking metrics. Moreover, the performance of kNN and BM25 are poor, which may be because other methods can take advantage of the inputs' semantic information. We also observed that adding graph embeddings to the input can effectively improve the performance since it can supplement some missing semantic relationships among the words compared to using only

Table 6. Performance of the Top 1, 3, and 5 Recommended Questions

Queried Skill entities	Relevancy			Validity		
	Top 1	Top 3	Top 5	Top 1	Top 3	Top 5
v_i^s	1.	1.	0.990	0.633	0.694	0.707
(v_i^s, v_{i*}^s)	0.883	0.950	0.943	0.967	0.872	0.867

word embeddings. We found that the performance of both the RGCN-based and GAT-based models decreased when we removed the $\mathcal{E}^{qw}$ information. This clearly demonstrates the effectiveness of edges $\mathcal{E}^{qw}$.

6 CASE STUDIES AND DISCUSSION

To further verify the effectiveness of our system, we invited two interviewers to evaluate the recommendation results of different queried skill entities. Specifically, we first randomly selected 30 skill entities from the skill entity vocabulary and then asked the interviewers to answer two questions about the top 5 recommended questions for each queried skill entity v_i^s.

- **Relevancy:** Does the recommended question relate to the queried skill?
- **Validity:** Can the recommended question be used in the actual interview process?

For each recommended question, the *Relevancy* and *Validity* are $\in \{0, 1\}$. Then, we collected the top 3 recommended skill entities v_{i1}^s, v_{i2}^s and v_{i3}^s by the skill entity suggestion model for each skill entity v_i^s. Then, we asked interviewers to query (v_i^s, v_{i1}^s), (v_i^s, v_{i2}^s) and (v_i^s, v_{i3}^s) and answer the *Relevancy* and *Validity* questions for the corresponding top 5 recommended questions.

Table 6 shows the average *Relevancy* and *Validity* scores for the top 1, 3, and 5 recommended questions. We can observe that our system can recommend suitable interview questions in response to the interviewers' search for specific skills. Specifically, we found that the top 5 recommendation results are highly relevant to the queried skill entities. In addition, we can observe that the recommended questions of multiple skill entities achieve higher *Validity* scores. That may be because the original click-through data query contains a single skill entity associated with some comprehensive knowledge pages, which may be too broad to generate suitable interview questions for a highly specific queried skills. Furthermore, it verifies the effectiveness of our skill suggestion module.

Moreover, we present two examples of the generated interview questions and recommended questions for the queried skill entity sets in Figures 7 and 8, respectively. Please note that we automatically translated the original generated Chinese sentences into English using the Baidu Translate Tool.

Specifically, for a selected web page $\widetilde{v}^{u,k}$ from the given subgraph $\mathcal{G}_k$, we show the generated results obtained by our model and the baseline models in Figure 7. We find that our model can generate fluent interrogative sentences that can be regarded as valid questions. Moreover, we observe that training our model without non-interrogative queries leads to the generation of less fluent questions. This may be because removing non-interrogative queries greatly reduced the amount of training data; thereby, it affects the performance of the proposed model in terms of fluency. Furthermore, it is difficult to ensure that the generated results are interrogative sentences when directly using PGNet on all the training instances. This also illustrates the importance of the proposed neural question generation model.

Then, we present an example of our question recommender system when the interviewer queries a skill set in Figure 8. Here, we can observe that, given an unseen skill set in training data, i.e., "*full-text search*" and "*big data*", our model can effectively recommend the related questions

Given $\widetilde{v}^{u,k}$:	Do THIS when the html page cannot be scrolled in web development
DuerQues-IQG:	What should we do when the html page cannot be scrolled in web development
Our model without $style_k = 0$ instances:	Why the html page cannot be scrolled
PGNet:	The html page cannot be scrolled in web development
Given $\widetilde{v}^{u,k}$:	The lock table problem in mysqldump
DuerQues-IQG:	Does mysqldump lock tables
Our model without $style_k = 0$ instances:	Is it mysqldump
PGNet:	mysqldump lock tables

Fig. 7. Case study on interview question generation.

Queried skill entity set	full-text search, big data
Recommended top-5 questions	1. What are the big data full-text retrieval technologies?
	2. What are the 5Vs of big data?
	3. What is the full-text search?
	4. What is the big data search engine ES?
	5. How to use full-text search in oracle?
Ground truth	What are the big data full-text retrieval technologies?

Fig. 8. Case study on interview question recommendation.

in our constructed skill-oriented question bank. In addition, we find that our system can not only recommend the ground truth question ($R_{i,j} = 1$) correctly but also recommend other relevant questions, such as "*What is the big data search engine ES?*" (**ES** is an abbreviation of **Elasticsearch**).
Limitations and Future Work: In this paper, limited by data usage policies, we generate skill-oriented interview questions only by learning the titles of KSC web pages and queries from query logs. Recently, some high-tech companies have gradually built internal knowledge search engines to achieve more efficient knowledge management [28, 51]. Indeed, the knowledge documents present within these internal knowledge search engines are accumulated from actual work projects of the enterprise, and the search and click logs can also provide valuable insights into the skills that employees require the most for their daily work. Therefore, an interesting research direction is to develop a more potent skill-oriented question bank that closely aligns with job requirements, using the data from these internal knowledge search engines. Additionally, exploring ways to develop a more accurate personalized question recommender system by utilizing the interviewer's preferences from the historical interview records and interactive data gathered by our system is another promising avenue for future research.

7 CONCLUSIONS

In this paper, we introduced an intelligent system for assisting job interviews, namely, DuerQues. Specifically, we first studied how to automatically generate skill-oriented interview questions in a scalable way by learning external knowledge from online KSCs. Along this line, we developed a distantly supervised skill entity recognition method to identify skill entities from large-scale search queries and web page titles with less need for human annotation. Then, we proposed a neural generative model for generating skill-oriented interview questions. Furthermore, we exploited the click-through data from the query logs and designed a recommender system for recommending the right questions to the interviewers. Specifically, we designed a graph-enhanced algorithm to recommend suitable questions efficiently given a set of queried skill entities. Finally, extensive experiments on the real-world datasets demonstrated the effectiveness of our DuerQues system in terms of the quality of generated skill-oriented questions and the performance of question recommendation.

APPENDICES

A. Performance of Distantly Supervised NER on the Weibo Dataset

To further verify the performance of the proposed distantly supervised entity recognition model, we evaluated it on a public Chinese benchmark NER dataset, namely, Weibo NER [58]. Here, we used the ground truth entities instead of human annotations as the vocabulary-based annotations and collected 514 categorized entities and 3,486 non-entity words. Then, based on our distantly supervised training data generation method, we constructed a new Weibo dataset, namely, DS-Weibo NER, where the training set is in a distantly supervised setting, while the test and validation sets are the same as the original Weibo NER sets.

The performance of our method and the baseline methods on both the DS-Weibo NER and Weibo NER datasets are shown in Table S1. Our method showed the best performance on the DS-Weibo NER dataset in terms of F1-value. After removing the BERT embeddings, the performance on both datasets dropped, which indicates that BERT embeddings can benefit the performance on the NER task, especially when the training corpus is sparse.

Table S1. Performance of Skill Entity Recognition on the Weibo and DS-Weibo NER Datasets

Datasets	Methods	Precision	Recall	F1
DS-Weibo	SUL+	**0.7561**	0.3708	0.4976
	- w/o SoftLexicon feature	0.7426	0.3589	0.4839
	- w/o BERT embedding	0.7340	0.3301	0.4554
	DuerQues-SER	0.6853	0.4689	**0.5568**
	- w/o non-skill entity annotation	0.6387	**0.4737**	0.5440
	- w/o non-skill entity sampling	0.6750	0.4522	0.5414
	- w/o SoftLexicon feature	0.6691	0.4402	0.5310
	- w/o BERT embedding	0.6741	0.3612	0.4704
Weibo	SUL+	**0.7256**	**0.6770**	**0.7005**
	- w/o SoftLexicon feature	0.6946	0.6746	0.6845
	- w/o BERT embedding	0.7160	0.5789	0.6402
	- w/o BERT embedding and SoftLexicon feature	0.6667	0.4593	0.5439

B. Interrogative Sentence Detection

In Section 3.2, we used a rule-based method to determine whether a query is an interrogative sentence. Specifically, based on studies of Chinese interrogative pronouns and interrogative sentences [60], we designed a series of regular matching rules as follows:

- $(HUI.*MA|SHI.*MA|YOU.*MA|HUI.*NE|SHI.*ME|YOU.*ME)$
- $(DUI\ MA|HAO\ MA|SHI\ MA|NENG\ MA|XING\ MA)$
- $(SHEN\ ME|YOU\ HE|NA|SHEI|SHA|ZEN\ YANG|WEI\ HE|DUO\ SHAO|ZEN\ ME|YOU\ HE)$
- $(SHI.*HAI\ SHI|GEN.*NA\ GE|NA\ GE.*BI\ JIAO|SHEI\ BI\ JIAO)$

For each query $\widetilde{v}^{q,k}$, if it contains any of the above regular patterns, we regard it as an interrogative sentence; i.e., $style_k = 1$.

C. More Examples of Generated Questions

Here we show more results generated by our model, its variant, and PGNet for each given web page $\widetilde{v}^{u,k}$ in Figure S1.

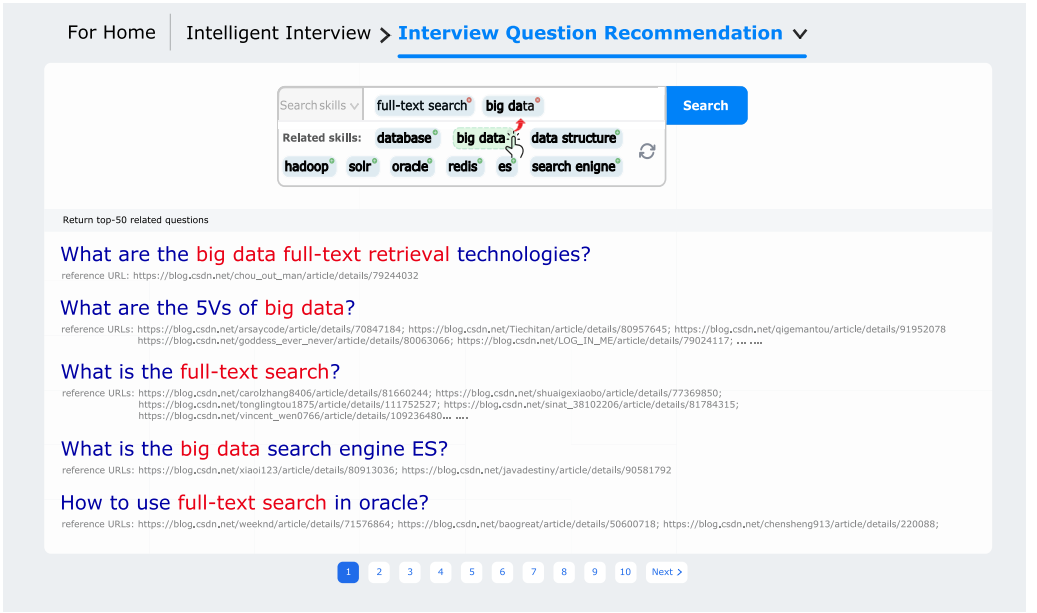

Fig. S1. More question generation examples.

D. More Examples of Recommended Questions

Here we show more examples of our interview question recommender system when the interviewer queries a skill set in Figure S2.

Queried skill entity set	nginx, load balancing
Recommended top-5 questions	1. How to configure load balancing with nginx?
	2. What are the load balancing algorithms in nginx?
	3. What is the difference between nginx reverse proxy and load balancing?
	4. What is the nginx reverse proxy?
	5. What are the load balancing strategies in ribbon?
Ground truth	How to configure load balancing with nginx? What are the load balancing algorithms in nginx?
Queried skill entity set	convolutional neural network, fully connected network, neural network
Recommended top-5 questions	1. What is the difference between a fully connected network and a convolutional neural network?
	2. What is a convolutional neural network?
	3. What is the activation function of the fully connected layer?
	4. What is pooling?
	5. What is a fully convolutional network?
Ground truth	What is the difference between a fully connected network and a convolutional neural network?

Fig. S2. More interview question recommendation examples.

E. System Demonstration

We developed a demo system as an important component of an intelligent interview product. An illustration of our system is shown in Figure S3.

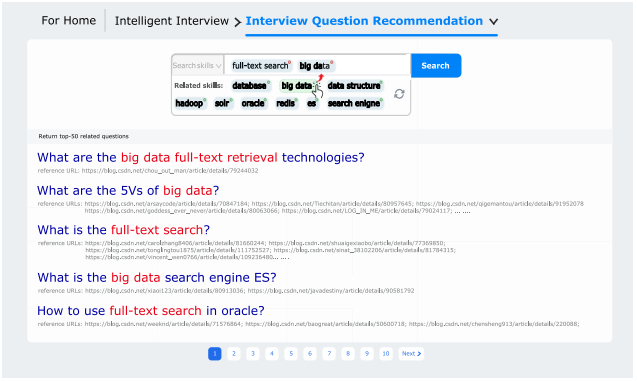

Fig. S3. Illustration of our DuerQues system. All the information has been translated into English.

REFERENCES

[1] Abdalghani Abujabal, Mohamed Yahya, Mirek Riedewald, and Gerhard Weikum. 2017. Automated template generation for question answering over knowledge graphs. In *Proceedings of the 26th International Conference on World Wide Web*. 1191–1200.

[2] Lada A. Adamic and Eytan Adar. 2003. Friends and neighbors on the web. *Social Networks* 25, 3 (2003), 211–230.

[3] Amr Ahmed, Nino Shervashidze, Shravan Narayanamurthy, Vanja Josifovski, and Alexander J. Smola. 2013. Distributed large-scale natural graph factorization. In *Proceedings of the 22nd International Conference on World Wide Web*. 37–48.

[4] Alfred V. Aho and Margaret J. Corasick. 1975. Efficient string matching: An aid to bibliographic search. *Commun. ACM* 18, 6 (1975), 333–340.

[5] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In *Proceedings of the 3rd International Conference on Learning Representations*.

[6] Mikhail Belkin and Partha Niyogi. 2001. Laplacian eigenmaps and spectral techniques for embedding and clustering. In *Proceedings of the 14th International Conference on Neural Information Processing Systems: Natural and Synthetic*. 585–591.

[7] Nikoletta Bika. 2021. A guide to interview preparation for employers. https://resources.workable.com/tutorial/preparing-conduct-interview.

[8] Sergey Brin and Lawrence Page. 2012. Reprint of: The anatomy of a large-scale hypertextual web search engine. *Computer Networks* 56, 18 (2012), 3825–3833.

[9] Robin Burke. 2002. Hybrid recommender systems: Survey and experiments. *User Modeling and User-Adapted Interaction* 12, 4 (2002), 331–370.

[10] Shaosheng Cao, Wei Lu, and Qiongkai Xu. 2015. GraRep: Learning graph representations with global structural information. In *Proceedings of the 24th ACM International on Conference on Information and Knowledge Management*. 891–900.

[11] Yixin Cao, Zikun Hu, Tat-Seng Chua, Zhiyuan Liu, and Heng Ji. 2019. Low-resource name tagging learned with weakly labeled data. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*. 261–270.

[12] Lei Chen, Gary Feng, Chee Wee Leong, Blair Lehman, Michelle Martin-Raugh, Harrison Kell, Chong Min Lee, and Su-Youn Yoon. 2016. Automated scoring of interview videos using Doc2Vec multimodal feature extraction paradigm. In *Proceedings of the 18th ACM International Conference on Multimodal Interaction*.

[13] Miao Chen, Chao Wang, Chuan Qin, Tong Xu, Jianhui Ma, Enhong Chen, and Hui Xiong. 2021. A trend-aware investment target recommendation system with heterogeneous graph. In *2021 International Joint Conference on Neural Networks (IJCNN)*. IEEE, 1–8.

[14] Kyunghyun Cho, Bart van Merrienboer, Çaglar Gülçehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning phrase representations using RNN encoder-decoder for statistical machine translation. In *EMNLP*.

[15] Yiming Cui, Wanxiang Che, Ting Liu, Bing Qin, and Ziqing Yang. 2021. Pre-training with whole word masking for Chinese BERT. *IEEE/ACM Transactions on Audio, Speech, and Language Processing* 29 (2021), 3504–3514.

[16] Le Dai, Yu Yin, Chuan Qin, Tong Xu, Xiangnan He, Enhong Chen, and Hui Xiong. 2020. Enterprise cooperation and competition analysis with a sign-oriented preference network. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 774–782.

[17] Soham Datta, Prabir Mallick, Sangameshwar Patil, Indrajit Bhattacharya, and Girish Palshikar. 2021. Generating an optimal interview question plan using a knowledge graph and integer linear programming. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. 1996–2005.

[18] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of NAACL-HLT*. 4171–4186.

[19] Xinya Du, Junru Shao, and Claire Cardie. 2017. Learning to ask: Neural question generation for reading comprehension. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. 1342–1352.

[20] Nan Duan, Duyu Tang, Peng Chen, and Ming Zhou. 2017. Question generation for question answering. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*. 866–874.

[21] Robert Gatewood, Hubert S. Feild, and Murray Barrick. 2015. *Human Resource Selection*. Cengage Learning.

[22] Jonas Gehring, Michael Auli, David Grangier, Denis Yarats, and Yann N. Dauphin. 2017. Convolutional sequence to sequence learning. In *International Conference on Machine Learning*. PMLR, 1243–1252.

[23] Goodtime. 2020. Automate interview scheduling with the top interview scheduling solution. https://www.goodtime.io/interview-scheduling.

[24] Aditya Grover and Jure Leskovec. 2016. node2vec: Scalable feature learning for networks. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. 855–864.
[25] Jiatao Gu, Zhengdong Lu, Hang Li, and Victor O. K. Li. 2016. Incorporating copying mechanism in sequence-to-sequence learning. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. 1631–1640.
[26] Qingyu Guo, Fuzhen Zhuang, Chuan Qin, Hengshu Zhu, Xing Xie, Hui Xiong, and Qing He. 2020. A survey on knowledge graph-based recommender systems. *IEEE Transactions on Knowledge and Data Engineering* (2020).
[27] Ido Guy, Naama Zwerdling, Inbal Ronen, David Carmel, and Erel Uziel. 2010. Social media recommendation based on people and tags. In *Proceedings of the 33rd International ACM SIGIR Conference on Research and Development in Information Retrieval*. 194–201.
[28] Jungpil Hahn and Mani Subramani. 2000. A framework of knowledge management systems: Issues and challenges for theory and practice. (2000).
[29] William L. Hamilton. 2020. Graph representation learning. *Synthesis Lectures on Artifical Intelligence and Machine Learning* 14, 3 (2020), 1–159.
[30] Michael M. Harris and Laurence S. Fink. 1987. A field study of applicant reactions to employment opportunities: Does the recruiter make a difference? *Personnel Psychology* 40, 4 (1987), 765–784.
[31] Christopher J. Hartwell, Clark D. Johnson, and Richard A. Posthuma. 2019. Are we asking the right questions? predictive validity comparison of four structured interview question types. *Journal of Business Research* 100 (2019), 122–129.
[32] Minlie Huang, Xiaoyan Zhu, and Jianfeng Gao. 2020. Challenges in building intelligent open-domain dialog systems. *ACM Transactions on Information Systems (TOIS)* 38 (2020), 1–32.
[33] Folasade Olubusola Isinkaye, Yetunde O. Folajimi, and Bolande Adefowoke Ojokoh. 2015. Recommendation systems: Principles, methods and evaluation. *Egyptian Informatics Journal* 16, 3 (2015), 261–273.
[34] Paul Jaccard. 1901. Étude comparative de la distribution florale dans une portion des Alpes et des Jura. *Bull. Soc. Vaudoise Sci. Nat.* 37 (1901), 547–579.
[35] Heysem Kaya and Albert Ali Salah. 2018. Multimodal personality trait analysis for explainable modeling of job interview decisions. In *Explainable and Interpretable Models in Computer Vision and Machine Learning*. Springer, 255–275.
[36] Jin-Young Kim and WanGyu Heo. 2021. Artificial intelligence video interviewing for employment: Perspectives from applicants, companies, developer and academicians. *Information Technology & People* (2021).
[37] Thomas N. Kipf and Max Welling. 2017. Semi-supervised classification with graph convolutional networks. In *International Conference on Learning Representations*.
[38] Joseph A. Konstan. 2004. Introduction to recommender systems: Algorithms and evaluation. *ACM Transactions on Information Systems (TOIS)* 22, 1 (2004), 1–4.
[39] John Lafferty, Andrew McCallum, and Fernando C. N. Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. (2001).
[40] Renaud Lambiotte, J.-C. Delvenne, and Mauricio Barahona. 2008. Laplacian dynamics and multiscale modular structure in networks. *arXiv preprint arXiv:0812.1770* (2008).
[41] Guillaume Lample, Miguel Ballesteros, Sandeep Subramanian, Kazuya Kawakami, and Chris Dyer. 2016. Neural architectures for named entity recognition. In *Proceedings of NAACL-HLT*. 260–270.
[42] Jing Li, Aixin Sun, Jianglei Han, and Chenliang Li. 2020. A survey on deep learning for named entity recognition. *IEEE Transactions on Knowledge and Data Engineering* (2020).
[43] Piji Li, Zihao Wang, Zhaochun Ren, Lidong Bing, and Wai Lam. 2017. Neural rating regression with abstractive tips generation for recommendation. In *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval*. 345–354.
[44] Xiaoya Li, Jingrong Feng, Yuxian Meng, Qinghong Han, Fei Wu, and Jiwei Li. 2020. A unified MRC framework for named entity recognition. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. 5849–5859.
[45] Jianxun Lian, Xiaohuan Zhou, Fuzheng Zhang, Zhongxia Chen, Xing Xie, and Guangzhong Sun. 2018. xDeepFM: Combining explicit and implicit feature interactions for recommender systems. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 1754–1763.
[46] Chin-Yew Lin. 2004. Rouge: A package for automatic evaluation of summaries. In *Text Summarization Branches Out*. 74–81.
[47] Bang Liu, Mingjun Zhao, Di Niu, Kunfeng Lai, Yancheng He, Haojie Wei, and Yu Xu. 2019. Learning to generate questions by learning what not to generate. In *The World Wide Web Conference*. 1106–1118.
[48] Sidi Lu, Yaoming Zhu, Weinan Zhang, Jun Wang, and Yong Yu. 2018. Neural text generation: Past, present and beyond. *arXiv preprint arXiv:1803.07133* (2018).

[49] Thang Luong, Hieu Pham, and Christopher D. Manning. 2015. Effective approaches to attention-based neural machine translation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*. 1412–1421.
[50] Xuezhe Ma and Eduard Hovy. 2016. End-to-end sequence labeling via bi-directional LSTM-CNNs-CRF. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. 1064–1074.
[51] A. D. Marwick. 2001. Knowledge management technology. *IBM Systems Journal* 40, 4 (2001), 814–830. https://doi.org/10.1147/sj.404.0814
[52] Tubagus Mohammad Akhriza, Yinghua Ma, and Jianhua Li. 2017. Revealing the gap between skills of students and the evolving skills required by the industry of information and communication technology. *International Journal of Software Engineering and Knowledge Engineering* 27, 05 (2017), 675–698.
[53] Iftekhar Naim, M. Iftekhar Tanveer, Daniel Gildea, and Mohammed Ehsan Hoque. 2015. Automated prediction and analysis of job interview performance: The role of what you say and how you say it. In *2015 11th IEEE International Conference and Workshops on Automatic Face and Gesture Recognition (FG)*, Vol. 1. IEEE, 1–6.
[54] OPM.gov. 2020. Policy, Data, Oversight: Assessment and Selection. https://www.opm.gov/policy-data-oversight/assessment-and-selection/other-assessment-methods/job-knowledge-tests/.
[55] Mingdong Ou, Peng Cui, Jian Pei, Ziwei Zhang, and Wenwu Zhu. 2016. Asymmetric transitivity preserving graph embedding. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. 1105–1114.
[56] Liangming Pan, Wenqiang Lei, Tat-Seng Chua, and Min-Yen Kan. 2019. Recent advances in neural question generation. *arXiv preprint arXiv:1905.08949* (2019).
[57] Minlong Peng, Ruotian Ma, Qi Zhang, and Xuanjing Huang. 2020. Simplify the usage of lexicon in Chinese NER. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. 5951–5960.
[58] Nanyun Peng and Mark Dredze. 2015. Named entity recognition for Chinese social media with jointly trained embeddings. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*. 548–554.
[59] Bryan Perozzi, Rami Al-Rfou, and Steven Skiena. 2014. DeepWalk: Online learning of social representations. In *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. 701–710.
[60] Yip Po-Ching and Don Rimmington. 2015. *Chinese: A Comprehensive Grammar*. Routledge.
[61] Richard A. Posthuma, Julia Levashina, Filip Lievens, Eveline Schollaert, Wei-Chi Tsai, Maria Fernanda Wagstaff, and Michael A. Campion. 2014. Comparing employment interviews in Latin America with other countries. *Journal of Business Research* 67, 5 (2014), 943–951.
[62] Tricia Prickett, Neha Gada-Jain, and Frank J. Bernieri. 2000. The importance of first impressions in a job interview. In *Annual Meeting of the Midwestern Psychological Association, Chicago, IL*.
[63] Chuan Qin, Kaichun Yao, Hengshu Zhu, Tong Xu, Dazhong Shen, Enhong Chen, and Hui Xiong. 2022. Towards automatic job description generation with capability-aware neural networks. *IEEE Transactions on Knowledge and Data Engineering* (2022).
[64] Chuan Qin, Hengshu Zhu, Tong Xu, Chen Zhu, Liang Jiang, Enhong Chen, and Hui Xiong. 2018. Enhancing person-job fit for talent recruitment: An ability-aware neural network approach. In *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval*. 25–34.
[65] Chuan Qin, Hengshu Zhu, Tong Xu, Chen Zhu, Chao Ma, Enhong Chen, and Hui Xiong. 2020. An enhanced neural network approach to person-job fit in talent recruitment. *ACM Transactions on Information Systems (TOIS)* 38, 2 (2020), 1–33.
[66] Chuan Qin, Hengshu Zhu, Chen Zhu, Tong Xu, Fuzhen Zhuang, Chao Ma, Jingshuai Zhang, and Hui Xiong. 2019. DuerQuiz: A personalized question recommender system for intelligent job interview. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*.
[67] Mingcheng Qu, Guang Qiu, Xiaofei He, Cheng Zhang, Hao Wu, Jiajun Bu, and Chun Chen. 2009. Probabilistic question recommendation for question answering communities. In *Proceedings of the 18th International Conference on World Wide Web*. 1229–1230.
[68] Pengjie Ren, Zhumin Chen, Z. Ren, Furu Wei, L. Nie, J. Ma, and M. Rijke. 2018. Sentence relations for extractive summarization with deep neural networks. *ACM Transactions on Information Systems (TOIS)* 36 (2018), 1–32.
[69] S. Robertson and H. Zaragoza. 2009. The probabilistic relevance framework: BM25 and beyond. *Found. Trends Inf. Retr.* 3 (2009), 333–389.
[70] Steven R. Ross. 1991. How to conduct a job interview. *American Journal of Hospital Pharmacy* 48, 4 (1991), 670–677.
[71] Michael Schlichtkrull, Thomas N. Kipf, Peter Bloem, Rianne Van Den Berg, Ivan Titov, and Max Welling. 2018. Modeling relational data with graph convolutional networks. In *European Semantic Web Conference*. Springer, 593–607.
[72] Abigail See, Peter J. Liu, and Christopher D. Manning. 2017. Get to the point: Summarization with pointer-generator networks. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. 1073–1083.

[73] Dazhong Shen, Chuan Qin, Chao Wang, Zheng Dong, Hengshu Zhu, and Hui Xiong. 2021. Topic modeling revisited: A document graph-based neural network perspective. *Advances in Neural Information Processing Systems* 34 (2021), 14681–14693.

[74] Dazhong Shen, Hengshu Zhu, Chen Zhu, Tong Xu, Chao Ma, and Hui Xiong. 2018. A joint learning approach to intelligent job interview assessment. In *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI-18*. International Joint Conferences on Artificial Intelligence Organization, 3542–3548.

[75] Baoxu Shi, Shan Li, Jaewon Yang, Mustafa Emre Kazdagli, and Qi He. 2020. Learning to ask screening questions for job postings. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*. 549–558.

[76] Baoxu Shi, Jaewon Yang, Feng Guo, and Qi He. 2020. Salience and market-aware skill extraction for job targeting. In *Proceedings of the SIGKDD'2020*.

[77] Yan Song, Shuming Shi, Jing Li, and Haisong Zhang. 2018. Directional skip-gram: Explicitly distinguishing left and right context for word embeddings. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*. 175–180.

[78] Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. 2014. Sequence to sequence learning with neural networks. In *Proceedings of the 27th International Conference on Neural Information Processing Systems-Volume 2*. 3104–3112.

[79] Oscar Täckström, Dipanjan Das, Slav Petrov, Ryan McDonald, and Joakim Nivre. 2013. Token and type constraints for cross-lingual part-of-speech tagging. *Transactions of the Association for Computational Linguistics* 1 (2013), 1–12.

[80] Hongkui Tu, Jiahui Wen, Aixin Sun, and Xiaodong Wang. 2018. Joint implicit and explicit neural networks for question recommendation in CQA services. *IEEE Access* 6 (2018), 73081–73092.

[81] Zhaopeng Tu, Zhengdong Lu, Yang Liu, Xiaohua Liu, and Hang Li. 2016. Modeling coverage for neural machine translation. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. 76–85.

[82] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. 2018. Graph attention networks. *Proceedings of the 6th International Conference on Learning Representations*.

[83] Chao Wang, Hengshu Zhu, Peng Wang, Chen Zhu, Xi Zhang, Enhong Chen, and Hui Xiong. 2021. Personalized and explainable employee training course recommendations: A Bayesian variational approach. *ACM Transactions on Information Systems (TOIS)* 40, 4 (2021), 1–32.

[84] Chao Wang, Hengshu Zhu, Chen Zhu, Chuan Qin, and Hui Xiong. 2020. SetRank: A setwise Bayesian approach for collaborative ranking from implicit feedback. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 34. 6127–6136.

[85] Yanan Wang, Qi Liu, Chuan Qin, Tong Xu, Yijun Wang, Enhong Chen, and Hui Xiong. 2018. Exploiting topic-based adversarial neural network for cross-domain keyphrase extraction. In *2018 IEEE International Conference on Data Mining (ICDM)*. IEEE, 597–606.

[86] Joseph Weizenbaum. 1966. ELIZA–a computer program for the study of natural language communication between man and machine. *Commun. ACM* (1966).

[87] Wendy Werner. 1990. *The Impact of Interviewer Training on Questioning Strategies for Use in the Employment Interview*. Ph.D. Dissertation. Hofstra University.

[88] Ronald J. Williams and David Zipser. 1989. A learning algorithm for continually running fully recurrent neural networks. *Neural Computation* 1, 2 (1989), 270–280.

[89] Fangzhao Wu, Junxin Liu, Chuhan Wu, Yongfeng Huang, and Xing Xie. 2019. Neural Chinese named entity recognition via CNN-LSTM-CRF and joint training with word segmentation. In *The World Wide Web Conference*. 3342–3348.

[90] Yu Wu, Yunli Wang, and Shujie Liu. 2020. A dataset for low-resource stylized sequence-to-sequence generation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 34. 9290–9297.

[91] Linli Xu, Liang Jiang, Chuan Qin, Zhe Wang, and Dongfang Du. 2018. How images inspire poems: Generating classical Chinese poetry from images with memory networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 32.

[92] Xiao Yan, Jaewon Yang, Mikhail Obukhov, Lin Zhu, Joey Bai, Shiqi Wu, and Qi He. 2019. Social skill validation at Linkedin. In *Proceedings of the SIGKDD'2019*.

[93] Diyi Yang, David Adamson, and Carolyn Penstein Rosé. 2014. Question recommendation with constraints for massive open online courses. In *Proceedings of the 8th ACM Conference on Recommender Systems*. 49–56.

[94] Yaosheng Yang, Wenliang Chen, Zhenghua Li, Zhengqiu He, and Min Zhang. 2018. Distantly supervised NER with partial annotation learning and reinforcement learning. In *Proceedings of the COLING'2018*.

[95] Yang Yang, Jia-Qi Yang, Ran Bao, De-Chuan Zhan, Hengshu Zhu, Xiao-Ru Gao, Hui Xiong, and Jian Yang. 2021. Corporate relative valuation using heterogeneous multi-modal graph neural network. *IEEE Transactions on Knowledge and Data Engineering* (2021).

[96] Zhilin Yang, Ruslan Salakhutdinov, and William W. Cohen. 2017. Transfer learning for sequence tagging with hierarchical recurrent networks. In *International Conference on Learning Representations*.

[97] Kaichun Yao, Chuan Qin, Hengshu Zhu, Chao Ma, Jingshuai Zhang, Yi Du, and Hui Xiong. 2021. An interactive neural network approach to keyphrase extraction in talent recruitment. In *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*. 2383–2393.

[98] Kaichun Yao, Jingshuai Zhang, Chuan Qin, Peng Wang, Hengshu Zhu, and Hui Xiong. 2022. Knowledge enhanced person-job fit for talent recruitment. In *2022 IEEE 38th International Conference on Data Engineering (ICDE)*. IEEE, 3467–3480.

[99] Daojian Zeng, Kang Liu, Yubo Chen, and Jun Zhao. 2015. Distant supervision for relation extraction via piecewise convolutional neural networks. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*.

[100] Le Zhang, Tong Xu, Hengshu Zhu, Chuan Qin, Qingxin Meng, Hui Xiong, and Enhong Chen. 2020. Large-scale talent flow embedding for company competitive analysis. In *Proceedings of The Web Conference 2020*. 2354–2364.

[101] Le Zhang, Ding Zhou, Hengshu Zhu, Tong Xu, Rui Zha, Enhong Chen, and Hui Xiong. 2021. Attentive heterogeneous graph embedding for job mobility prediction. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*. 2192–2201.

[102] Yue Zhang and Jie Yang. 2018. Chinese NER using lattice LSTM. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. 1554–1564.

[103] Zheng Zhang, Minlie Huang, Z. Zhao, Feng Ji, Haiqing Chen, and Xiaoyan Zhu. 2019. Memory-augmented dialogue management for task-oriented dialogue systems. *ACM Transactions on Information Systems (TOIS)* 37 (2019), 1–30.

[104] Zhe Zhang, Hui Xiong, Tong Xu, Chuan Qin, Le Zhang, and Enhong Chen. 2022. Complex attributed network embedding for medical complication prediction. *Knowledge and Information Systems* 64, 9 (2022), 2435–2456.

[105] Wayne Xin Zhao, Sui Li, Yulan He, Edward Y. Chang, Ji-Rong Wen, and Xiaoming Li. 2015. Connecting social media to e-commerce: Cold-start product recommendation using microblogging information. *IEEE Transactions on Knowledge and Data Engineering* 28, 5 (2015), 1147–1159.

[106] GuoDong Zhou and Jian Su. 2002. Named entity recognition using an HMM-based chunk tagger. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*. 473–480.

[107] Liang Zhou and Eduard Hovy. 2004. Template-filtered headline summarization. In *Text Summarization Branches Out*. 56–60.

[108] Hengshu Zhu, Enhong Chen, Hui Xiong, Kuifei Yu, Huanhuan Cao, and Jilei Tian. 2014. Mining mobile user preferences for personalized context-aware recommendation. *ACM Transactions on Intelligent Systems and Technology (TIST)* 5, 4 (2014), 1–27.

[109] Fuzhen Zhuang, Zhiyuan Qi, Keyu Duan, Dongbo Xi, Yongchun Zhu, Hengshu Zhu, Hui Xiong, and Qing He. 2020. A comprehensive survey on transfer learning. *Proc. IEEE* 109, 1 (2020), 43–76.

Received 4 July 2021; revised 13 April 2023; accepted 27 May 2023