An Enhanced Neural Network Approach to Person-Job Fit in Talent Recruitment

CHUAN QIN, School of Computer Science, University of Science and Technology of China
HENGSHU ZHU, Baidu Talent Intelligence Center, Baidu Inc.
TONG XU, School of Computer Science, University of Science and Technology of China
CHEN ZHU and CHAO MA, Baidu Talent Intelligence Center, Baidu Inc.
ENHONG CHEN and HUI XIONG, School of Computer Science, University of Science and Technology of China

The widespread use of online recruitment services has led to an information explosion in the job market. As a result, recruiters have to seek intelligent ways for Person-Job Fit, which is the bridge for adapting the right candidates to the right positions. Existing studies on Person-Job Fit usually focus on measuring the matching degree between talent qualification and job requirements mainly based on the manual inspection of human resource experts, which could be easily misguided by the subjective, incomplete, and inefficient nature of human judgment. To that end, in this article, we propose a novel end-to-end **T**opic-based **A**bility-aware **P**erson-**J**ob **F**it **N**eural **N**etwork (TAPJFNN) framework, which has a goal of reducing the dependence on manual labor and can provide better interpretability about the fitting results. The key idea is to exploit the rich information available in abundant historical job application data. Specifically, we propose a word-level semantic representation for both job requirements and job seekers' experiences based on Recurrent Neural Network (RNN). Along this line, two hierarchical topic-based ability-aware attention strategies are designed to measure the different importance of job requirements for semantic representation, as well as measure the different contribution of each job experience to a specific ability requirement. In addition, we design a refinement strategy for Person-Job Fit prediction based on historical recruitment records. Furthermore, we introduce how to exploit our TAPJFNN framework for enabling two specific applications in talent recruitment: talent sourcing and job recommendation. Particularly, in the application of job recommendation, a novel training mechanism is designed for addressing the challenge of biased negative labels. Finally, extensive experiments on a large-scale real-world dataset clearly validate the effectiveness and interpretability of the TAPJFNN and its variants compared with several baselines.

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

Additional Key Words and Phrases: Recruitment analysis, person-job fit, neural network

15

This is a substantially extended and revised version of Qin et al. [2018], which appears in the 41st International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'18).

This work was partially supported by grants from the National Natural Science Foundation of China (Nos. 91746301, 61703386, 61836013, and U1605251).

Authors' addresses: C. Qin, T. Xu, E. Chen, and H. Xiong (corresponding author), School of Computer Science, University of Science and Technology of China; emails: chuanqin0426@gmail.com, {tongxu, cheneh}@ustc.edu.cn, xionghui@gmail.com; H. Zhu (corresponding author), C. Zhu, and C. Ma, Baidu Talent Intelligence Center, Baidu Inc.; emails: zhuhengshu@gmail.com, {zhuchen02, machao13}@baidu.com.

1046-8188/2020/02-ART15 $15.00

https://doi.org/10.1145/3376927

ACM Reference format:
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 Trans. Inf. Syst.* 38, 2, Article 15 (February 2020), 33 pages.
https://doi.org/10.1145/3376927

1 INTRODUCTION

The rapid development of online recruitment platforms, such as LinkedIn and Lagou, has enabled the new paradigm for talent recruitment. For instance, in 2017, there were *467M* users and *3M* active job listings in LinkedIn from about *200* countries and territories all over the world [Chaudhary 2017]. While popular online recruitment services provide more convenient channels for both employers and job seekers, it also raises the challenge of **Person-Job Fit** due to information explosion. According to Reference SHRM [2016], the recruiters now need about 42 days and 4K dollars on average for locking a suitable employee. Clearly, more effective techniques are urgently required for the Person-Job Fit task, which targets at measuring the matching degree between the talent qualification and the job requirements.

Indeed, as a crucial task in human resource management, traditional Person-Job Fit has been studied from different perspectives, such as big five personality factor model [Chuang and Sackett 2005], job characteristic beliefs [Ehrhart 2006; Ehrhart and Makransky 2007], and job crafting [Kooij et al. 2017]. Recently, some researchers in the field of machine learning and information retrieval have also become interested in Person-Job Fit task. Those studies can be traced back to Malinowski et al. [2006], where the authors focus on the candidate matching by using profile information from both candidates and jobs. Subsequently, some researchers followed the idea of recommender system to recommend suitable jobs for candidates [Diaby et al. 2013; Lee and Brusilovsky 2007; Zhang et al. 2014]. However, these efforts largely depend on the manual inspection of features or key phrases from domain experts and thus lead to high cost and the inefficient, inaccurate, and subjective judgments. Besides, though above studies focus on the applications of Person-Job Fit task, mathematical definition of this task has not been formally provided.

To that end, in this article, we first introduce a formal definition of the Person-Job Fit task and then propose an end-to-end Topic-based Ability-aware Person-Job Fit Neural Network (TAPJFNN) model, which has a goal of reducing the dependence on human labeling data and can provide better interpretation about the fitting results. The key idea of our approach is motivated by the example shown in Figure 1. There are four requirements, including three technical skill requirements (*programming, machine learning,* and *big data processing*) and 1 comprehensive quality requirement (*communication and team work*). Since multiple abilities may fit the same requirement and different candidates may have different abilities, all the abilities should be weighed for a comprehensive score to compare the matching degree among different candidates. During this process, traditional methods, which simply rely on keywords/feature matching [Elsafty et al. 2018; Lee and Brusilovsky 2007], may either ignore some abilities of candidates or mislead recruiters by subjective and incomplete weighing of abilities/experiences from domain experts. Therefore, for developing more effective and comprehensive Person-Job Fit solutions, abilities should be not only represented via the semantic understanding of rich textual information from the large amount of job application data, but also automatically weighed based on the historical recruitment results.

Along this line, all the job postings and resumes should be comprehensively analyzed without relying on human judgment. To be specific, for representing both the job-oriented **abilities** and experiences of candidates, we first propose a word-level semantic representation based on Recurrent Neural Network (RNN) to learn the latent features of each word in a joint semantic space.

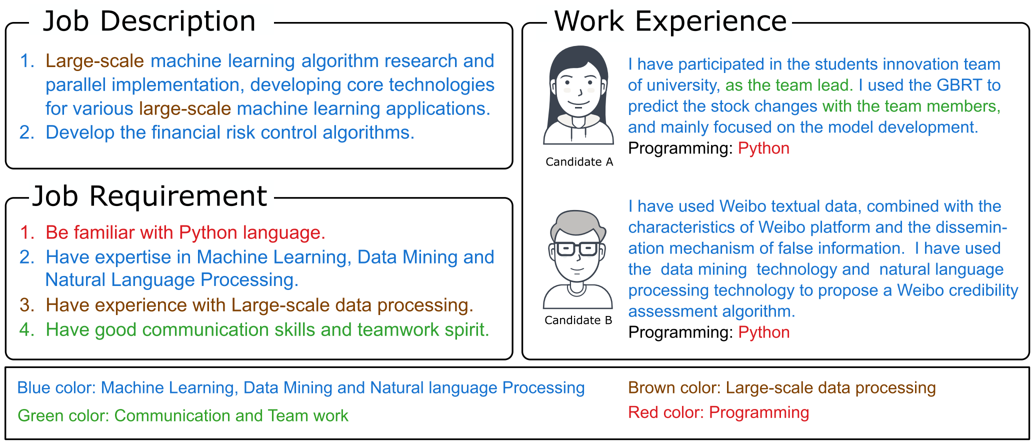


Fig. 1. A motivating example of Person-Job Fit. For job posting, it usually contains the job description and job requirement. For candidate's resume, the description of their work experience can help us a lot to measure their capability. We can measure the matching degree between the job posting and talent from the perspective of ability. Here, the text in different colors represents different kinds of abilities.

Then, two hierarchical **topic-based ability-aware** structures are designed to guide the learning of semantic representation and incorporate the global meaning for job requirements as well as the corresponding experiences of candidates. In addition, for measuring the importance of different abilities, as well as the relevance between requirements and experiences, we also design two hierarchical topic-based ability-aware attention strategies to highlight those crucial abilities or experiences. This scheme will not only improve the performance, but also enhance the interpretability of matching results. Moreover, we introduce a refinement strategy for Person-Job Fit prediction based on the historical recruitment records to enhance the performance of predicting the matching degree between the talent and job. Furthermore, we introduce how to exploit our TAPJFNN framework for enhancing two specific applications in talent recruitment: targeted talent sourcing and job recommendation. Particularly, in the application of job recommendation, a novel training mechanism is designed for addressing the challenge of biased negative labels. Finally, extensive experiments on a large-scale real-world dataset clearly validate the effectiveness of our TAPJFNN framework, which outperforms several competitive baselines. Compared with our preliminary model [Qin et al. 2018], the contribution of this article can be summarized as follows:

- We propose a novel variant framework called TAPJFNN, which involves two different hierarchical topic-based ability-aware attention strategies and a refinement strategy for Person-Job Fit prediction.
- We exploit our TAPJFNN framework for enhancing two specific applications in talent recruitment: targeted talent sourcing and job recommendation. Particularly, in the application of job recommendation, a novel training mechanism is designed for addressing the challenge of biased negative labels.
- Extensive experiments on a large-scale real-world dataset clearly validate the effectiveness of our TAPJFNN framework compared with several baselines.
- We further introduce the expansion version of our TAPJFNN framework to model the non-textual features, and then discuss the impact of involving non-textual features on algorithm fairness through extensive experimentation.

Overview. The rest of this article is organized as follows: In Section 2, we briefly introduce some related works of our study. In Section 3, we introduce the preliminaries and formally define the problem of Person-Job Fit. Technical details of our Topic-based Ability-aware Person-Job Fit Neural Network will be introduced in Section 4. Then, we give two important Person-Job Fit applications, i.e., Talent sourcing and Job Recommendation, in Section 5. We comprehensively evaluate the model performance in Section 6, with some further discussions on the interpretability of results. In Section 7, we conclude the article.

2 RELATED WORK

In this section, we will briefly provide a comprehensive review of the relevant approaches. Specifically, we group the related works into three lines of literature: *Recruitment Analysis*, *Deep Learning for Text Classification and Matching*, and *Recommendation with Textual Information*.

2.1 Recruitment Analysis

Recruitment is always a core function of human resource management to support the success of organizations. Recently, the newly available recruitment big data enables researchers to conduct recruitment analysis through more quantitative ways [Harris 2017; Javed et al. 2017; Lin et al. 2017; Meng et al. 2018, 2019; Sun et al. 2019; Teng et al. 2019; Wu et al. 2019; Xu et al. 2015, 2016; Ye et al. 2019; Zhang et al. 2019a, 2019c; Zhu et al. 2016]. In particular, the study of measuring the matching degree between the talent qualification and job requirements, namely Person-Job Fit [Sekiguchi 2004], has become one of the most striking topics.

The early research efforts of Person-Job Fit can be dated back to Malinowski et al. [2006], where the authors built a bilateral person-job recommendation system using the profile information from both candidates and jobs to find a good match between talents and jobs. Then, Lee and Brusilovsky [2007] followed the idea of recommender systems and proposed a comprehensive job recommender system, which is based on a broad range of job preferences and interests. Also, Diaby et al. [2013] introduced a content-based recommender system that proposes jobs to Facebook and LinkedIn users. Hong et al. [2013] improved the recommendation performance by updating the user profiles in a timely manner. Zhang et al. [2014] compared a number of user-based collaborative filtering and item-based collaborative filtering algorithms on recommending suitable jobs for job seekers. Recently, Zhu et al. [2018] proposed a Convolutional Neural Network (CNN) based Person-Job Fit Neural Network for matching a talent qualification to the requirements of a job, which is close to our research goal. Compared with previous studies, our framework TAPFJNN consists of two hierarchical topic-based ability-aware attention strategies, and thus can achieve better person-job fit performance for applications of talent recruitment, as well as better interpretability.

Moreover, the emergence of various online recruitment services provides a novel perspective for recruitment analysis. For example, Zhang et al. [2016] proposed a generalized linear mixed model (GLMix)—a more fine-grained model at the user or item level—in the LinkedIn job recommender system and generated 20% to 40% more job applications for job seekers. Yin et al. [2019] introduced a "two-way selection" algorithm to help event organizers effectively select attenders. Song et al. [2015, 2016] leveraged multiple social network data including LinkedIn, Twitter, and Facebook for helping recruit suitable volunteers from the huge crowd. After that, Jia et al. [2016] further improved the performance of volunteer recruitment by fusing social networks using deep learning with source confidence and consistency regularization. Cheng et al. [2013] collected the job-related information from various social media sources and constructed an inter-company job-hopping network to demonstrate the flow of talents. Also, Wang et al. [2013] predicted the job transition of employees by exploiting their career path data. Further, Li et al. [2017] designed a survival analysis model to handle two critical issues in talent management: turnover and career progression. In Xu et al. [2018], the authors introduced a data-driven approach to model the popularity of job skills. Xu et al. [2016] proposed a talent circle detection model based on a job transition network that can help the organizations to find the right talents and deliver career suggestions for job seekers to locate suitable jobs. Shen et al. [2018] developed a latent variable model to exploit job interview records and job and user profiles for intelligent job interview assessment. Ye et al. showed the potential of the structured graph data to identify high potential talent [Wang et al. 2017a]. Different from the above studies for, in this article, we focus on the crucial task, i.e., Person-Job Fit in talent recruitment, by proposing an end-to-end neural network approach and enable two specific applications in recruitment system: talent sourcing and job recommendation.

2.2 Deep Learning for Text Classification and Matching

Generally, the study of Person-Job Fit based on textual information can be grouped into the tasks of text mining, which is highly related to text classification [Huang et al. 2017; Kim 2014; Yang and Pedersen 1997] and text matching [Gomaa and Fahmy 2013; Severyn and Moschitti 2015] in Natural Language Processing (NLP) technologies.

Traditional methods on text classification and matching largely depend on the effective human-designed representations and input features (e.g., word n-gram [Wang and Manning 2012], parse trees [Ji and Eisenstein 2013], and lexical features [Melville et al. 2009]). Recently, deep learning has demonstrated its advanced performance and flexibility in many research fields, such as computer vision [Krizhevsky et al. 2012; Wu et al. 2018a, 2018b], information retrieval [Guo et al. 2016; Wang et al. 2017b], recommender system [Chin et al. 2018; He et al. 2018; Xu et al. 2019], and especially text mining [Kim 2014; Tang et al. 2015]. Many researchers have developed effective deep learning models on text classification and text matching tasks without labor-intensive feature engineering.

Specifically, the researchers first adopted the Recursive Neural Network to model the parse tree structure of sentences and successfully applied to text classification task [Hermann and Blunsom 2013; Socher et al. 2011]. Then, Convolutional Neural Network (CNN) proved its effectiveness to extract local semantics and hierarchical relationships in textual data. For instance, Kalchbrenner et al. [2014] proposed a Dynamic Convolutional Neural Network (DCNN) for modeling sentences, which obtained remarkable performance in several text classification tasks. After that, Zhang et al. [2015] used character-level convolutional networks for dealing with similar vocabulary redundancy problem. Meanwhile, Recurrent Neural Network (RNN) also benefits the text classification task, as it can capture the serialization information and learn the long-span relations or global semantic representation. Tai et al. [2015] introduced a special RNN—namely, tree-structured Long Short-Term Memory (LSTM) network—for sentiment classification. The Hierarchical Attention Network used two RNN layers with the global attention mechanism for modeling the sentence and document level representations, respectively, to further implement the document-level text classification task [Yang et al. 2016]. Moreover, Lai et al. [2015] and Xiao and Cho [2016] constructed the models by combining both CNN and RNN and showed the improvements in the topic and sentiment classification tasks. Besides, Peng et al. [2018] presented a deep graph CNN model to perform large-scale hierarchical text classification. Finally, Huang et al. [2019] applied the documentation representing layer to text-category attention based hierarchical RNN to handle the hierarchical multi-label text classification problem.

For text matching task, the deep learning models can be roughly divided into two major categories: representation-focused model and interaction-focused model [Zhang et al. 2018]. Usually, representation-focused models first generate sentence representation with neural networks and then measure the matching degree through different types of score functions. For instance, DSSM [Huang et al. 2013] is one of the first deep models proposed for semantic similarity by using DNN to represent each input sentence. Shen et al. [2014] improved the performance through learning representation vectors for input text by CNN. In contrast, interaction-focused models force on extracting the local interactive features based on words or phrases' basic representations, followed by the deep neural networks to learn the complex interaction patterns for matching. Hu et al. [2014] proposed the Architecture-II (ARC-II) to implement text matching, which uses convolution operation to compute the interaction feature vector between the sentence pairs. MatchPyramid [Pang et al. 2016] uses dot product between the word representations as their interaction features. Furthermore, Zhang et al. [2018] leveraged a graph convolutional network to learn the representations and modeled the local interactions based on an attention mechanism to handle the short-long text matching problem.

2.3 Recommendation with Textual Information

Since the Person-Job Fit model can be applied to the recommendation scenarios, we will finally review the relevant works of recommendation system, especially those that leverage some auxiliary textual information to improve the recommendation performance.

Recommender system targets on providing the accurate online items or information to the users [Errico et al. 2015; Sun et al. 2018; Wang et al. 2012]. Usually, recommendation models can be grouped into three categories: collaborative filtering [Wang et al. 2020], content-based filtering, and hybrid recommendation model [Zhang et al. 2019b]. Collaborative filtering models make prediction about the interests of the users based on the user-item historical interactions' either explicit/implicit feedback. Correspondingly, content-based filtering models are based on the description of the item and a profile of the user's preferences information. At the same time, hybrid recommender methods are the approaches that integrate two or more types of recommendation strategies. Among them, traditional collaborative filtering models have two significant drawbacks, including data sparsity and cold-start problem. To alleviate these problems, some researchers have tried to construct the recommender systems with some auxiliary textual information closely related to users and items. For instance, in Cheng et al. [2018a], McAuley and Leskovec [2013], and Tan et al. [2016], the authors have successfully applied topic models to extract the auxiliary features from the reviews for rating prediction. Also, Cheng et al. [2018b] introduced an aspect-aware topic model to model the user preferences and item features integrated into an aspect-aware latent factor model. In Chin et al. [2018] and Zheng et al. [2017], the authors used the deep learning–based model to extract the information from the reviews. In addition, extracting features from textual data (e.g., bag-of-words model) often results in a rich feature set. Therefore, some researchers have attempted to address the problem of limited recommendation performance due to the curse of high-dimensional and sparse features [Chen et al. 2017b, 2019]. Furthermore, different types of textual information are also applied to other recommendation applications. For example, textual features extracted from lyric data are widely used in music recommendations [Van den Oord et al. 2013]. Similarly, question and user answer data are helpful for making the question recommendation in the Community Question Answering (CQA) services [Kabutoya et al. 2010].

In this article, we follow some outstanding ideas in the above works according to the properties of Person-Job Fit task. Along this line, we propose an interpretable end-to-end neural model TAPJFNN based on RNN with four different ability-aware attention mechanisms. Therefore, TAPJFNN can not only improve the performance of Person-Job Fit task, but also enhance the model interpretability in practical scenarios.

3 PROBLEM FORMULATION

In this article, we target at dealing with the problem of Person-Job Fit, which focuses on measuring the matching degree between job description (job duties) and job requirements in a *job posting*

Table 1. Mathematical Notations

Symbol	Description
j_l	The lth job requirement in job posting j
r_l	The lth work/project experience in candidate's resume r
$w^J_{l,t}$	The work embedding of tth word in job requirement j_l
$w^R_{l,t}$	The work embedding of tth word in candidate's experience r_l
$h^J_{l,t}$	The word-level representation of tth word in job requirement j_l
$h^R_{l,t}$	The word-level representation of tth word in candidate's experience r_l
s^J_l	The single topic-based ability-aware representation of job requirement j_l
s^R_l	The single topic-based ability-aware representation of candidate's experience r_l
g^J	The multiple topic-based ability-aware representation of job posting j
g^R	The multiple topic-based ability-aware representation of candidate's resume r
$g^{J,R}_{+,1:k}$	The multiple topic-based ability-aware representations of k candidates' resumes who successfully applied for the job J
$g^{J,R}_{-,1:k}$	The multiple topic-based ability-aware representations of k candidates' resumes who unsuccessfully applied for the job J
p	The number of job requirements in job posting j
q	The number of work/project experiences in candidate's resume r
m_l	The number of words in job requirement j_l
n_l	The number of words in candidate's experience r_l

and the candidates' experiences in a *resume*.[1] For facilitating illustration, we list some important mathematical notations used throughout this article in Table 1.

Specifically, to formulate the problem of Person-Job Fit, we use J to denote a **job posting**, which totally contains p pieces of job requirements and job duties, denoted as $J = \{j_1, j_2, \ldots, j_p\}$. For simplicity, we call them **ability requirements**. For instance, there exist 4 job requirements and 2 duties in Figure 1, thus $p = 6$ in this case. Generally, we consider two types of ability requirements, i.e., the **professional skill** requirements (e.g., *Data Mining* and *Natural Language Processing* skills) and **comprehensive quality** requirements (e.g., *Team Work*, *Communication Skill,* and *Sincerity*). All the requirements will be analyzed comprehensively without special distinction by different types. Moreover, each j_l is assumed to contain m_l words, i.e., $j_l = \{j_{l,1}, j_{l,2}, \ldots, j_{l,m_l}\}$.

Similarly, we use R to represent a **resume** of a candidate, which includes q pieces of **experiences**, i.e., $\{r_1, r_2, \ldots, r_q\}$. In particular, due to the limitation of our real-world data, in this article, we mainly focus on the working experiences of candidates, as well as description of some other achievements, e.g., *project experiences, competition awards,* or *research paper publications.* Besides, each experience r_l is described by n_l words, i.e., $r_l = \{r_{l,1}, r_{l,2}, \ldots, r_{l,n_l}\}$.

Finally, we use S to indicate a *job application*, i.e., a Person-Job pair. Correspondingly, we have a *recruitment result label* $y \in \{0, 1\}$ to indicate whether the candidate has passed the interview process, i.e., $y = 1$ means a successful application, while $y = 0$ means a failed one.[2] What should be noted is that one candidate can apply for several jobs simultaneously, and one job can be applied for by multiple candidates. Thus, the same J may exist in different S, so does R. Along this line, we can formally define the problem of Person-Job Fit as follows:

[1]We ignore other non-textual features here and put this part of the discussion in Section 5.
[2]In our study, as long as the offer notification is issued, it is considered a successful application.

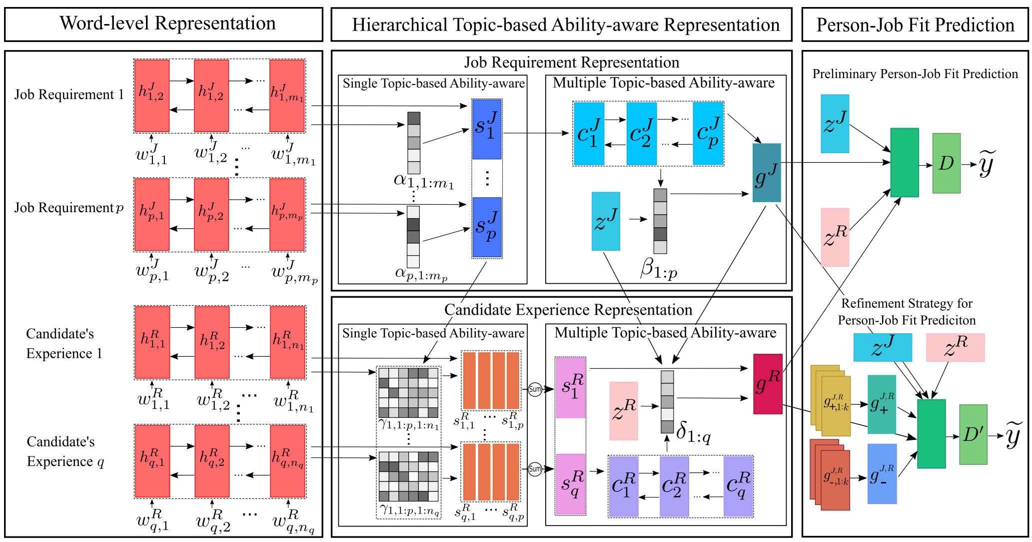


Fig. 2. An illustration of the proposed Topic-based Ability-aware Person-Job Fit Neural Network (TAPJFNN), which can be separated into three components: Word-level Representation, Hierarchical Topic-based Ability-aware Representation, and Person-Job Fit Prediction. Two different hierarchical structures are used to learn the topic-based ability-aware representation of job requirement and candidate experience, respectively. Meanwhile, the Person-Job Fit Prediction contains two parts: Preliminary Person-Job Fit Prediction and Refinement Strategy for Person-Job Fit Prediction.

Definition 3.1 (Problem Definition). Given a set of job applications $\mathcal{S}$, where each application $S \in \mathcal{S}$ contains a job posting J and a resume R, as well as the corresponding recruitment result label y, the target of Person-Job Fit is to learn a predictive model M for measuring the matching degree between J and R, and then the corresponding result label y can be predicted.

In the following section, we will introduce the technical details of our TAPJFNN model for addressing the above problem.

4 TOPIC-BASED ABILITY-AWARE PERSON-JOB FIT NEURAL NETWORK

As shown in Figure 2, TAPJFNN mainly consists of three components: *Word-level Representation*, *Hierarchical Topic-based Ability-aware Representation*, and *Person-Job Fit Prediction*.

Specifically, in Word-level Representation, we first leverage an RNN to project words of job postings and resumes onto latent representations, respectively, along with sequential dependence between words. Then, we feed the word-level representations into Hierarchical Topic-based Ability-aware Representation and then extract the ability-aware representations for job postings and resumes simultaneously by hierarchical representation structures. To capture the semantic relationships between job postings and resumes and further enhance the interpretability of the model, we design four attention mechanisms from the perspective of ability to polish their representations at different levels in this component. Finally, the jointly learned representations of job postings and resumes are fed into Person-Job Fit Prediction to evaluate the matching degree between them.

4.1 Word-level Representation

To embed the sequential dependence between words into corresponding representations, we leverage a special RNN—namely, Bi-directional Long Short Term Memory network (BiLSTM)—on a shared word embedding to generate the word-level representations for job postings and resumes.

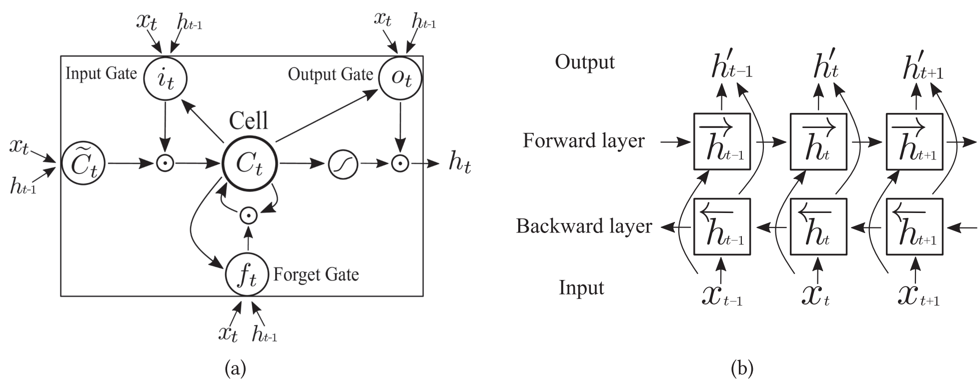


Fig. 3. (a) The architecture of Long Short-Term Memory block with one cell. (b) The architecture of bidirectional recurrent neural network.

Compared with the vanilla RNN, LSTM [Hochreiter and Schmidhuber 1997] can not only store and access a longer range of contextual information in the sequential input, but also handle the vanishing gradient problem in the meanwhile. As a variant of LSTM, BiLSTM is composed of a forward LSTM and backward LSTM [Graves and Schmidhuber 2005]. Because it is able at any point in the sequence to preserve information from both past and future, BiLSTM can better understand context than LSTM and show its advantages in many different text mining tasks, such as sentiment analysis [Chen et al. 2017a], sentence similarity [Neculoiu et al. 2016], and name entity recognition [Lample et al. 2016].

Figure 3(a) illustrates a single cell in LSTM, which has a cell state and three gates, i.e., input gate i, forget gate f, and output gate o. Formally, the LSTM can be formulated as follows:

$$\begin{aligned} i_t &= \sigma(W_i[x_t, h_{t-1}] + b_i),\ f_t = \sigma(W_f[x_t, h_{t-1}] + b_f), \\ \widetilde{C}_t &= tanh(W_C[x_t, h_{t-1}] + b_C),\ C_t = f_t \odot C_{t-1} + i_t \odot \widetilde{C}_t, \\ o_t &= \sigma(W_o[x_t, h_{t-1}] + b_o),\ h_t = o_t \odot tanh(C_t), \end{aligned} \tag{1}$$

where $X = \{x_1, x_2, \ldots, x_m\}$ and m denote the input vector and the length of X, respectively. Also, W_f, W_i, W_C, W_o, b_f, b_i, b_C, b_o are the parameters as weight matrices and biases, $\odot$ represents element-wise multiplication, σ is the sigmoid function, and $\{h_1, h_2, \ldots, h_m\}$ represents a sequence of semantic features. Furthermore, the above formulas can be represented in short as:

$$h_t = LSTM(x_t, h_{t-1}). \tag{2}$$

As shown in Figure 3(b), BiLSTM uses the input sequential data and their reverse to train the semantic vectors $\{h'_1, h'_2, \ldots, h'_m\}$. The hidden vector h'_t is the concatenation of the forward hidden vector $\overrightarrow{h_t}$ and backward hidden vector $\overleftarrow{h_t}$ at t-step. Specifically, we have

$$\begin{aligned} \overrightarrow{h_t} &= LSTM(x_t, \overrightarrow{h_{t-1}}), \\ \overleftarrow{h_t} &= LSTM(x_t, \overleftarrow{h_{t+1}}), \\ h'_t &= \left[\overrightarrow{h_t}; \overleftarrow{h_t}\right]. \end{aligned} \tag{3}$$

We can represent the above formulas in short as:

$$h'_t = BiLSTM(x_{1:m}, t),\ \ \forall t \in [1, \ldots, m], \tag{4}$$

where $x_{1:m}$ denotes the input sequence $\{x_1, \ldots, x_m\}$.

Now, we can use BiLSTM to model word-level representation in job posting J and resume R. For the lth job requirement $j_l = \{j_{l,1}, \ldots, j_{l,m_l}\}$, we first embed the words in j_l to vectors by

$$w^J_{l,t} = W_e j_{l,t}, \quad w^J_{l,t} \in \mathbb{R}^{d_0}, \tag{5}$$

where $w^J_{l,t}$ denotes d_0-dimensional word embedding of tth word in j_l. As for R, word embedding $w^R_{l',t'}$ of t'th word in candidate experience $r_{l'}$ is generated by a similar way. It should be noted that the job postings and resumes share the same matrix W_e, which is initialized by a pre-trained word vector matrix and re-trained during training processing.

Then, for each word in the lth job requirement j_l and l'th candidate experience $r_{l'}$, we can calculate the word-level representation $\{h^J_{l,1}, h^J_{l,2}, \ldots, h^J_{l,m_l}\}$ and $\{h^R_{l',1}, h^R_{l',2}, \ldots, h^R_{l',n_{l'}}\}$ by:

$$\begin{aligned} h^J_{l,t} &= BiLSTM\left(w^J_{l,1:m_l}, t\right), \quad \forall t \in [1, \ldots, m_l], \\ h^R_{l',t'} &= BiLSTM\left(w^R_{l',1:n_{l'}}, t'\right), \quad \forall t' \in [1, \ldots, n_{l'}], \end{aligned} \tag{6}$$

where $w^J_{l,1:m_l}$ and $w^R_{l',1:n_{l'}}$ denote the word vectors' input sequences of j_l and $r_{l'}$, respectively. Also, $h^J_{l,t}$ presents the d_0-dimension semantic representation of the tth word in the lth job requirement j_l, and $h^R_{l',t'}$ denotes the representation of t'th word in the l'th candidate experience $r_{l'}$.

4.2 Hierarchical Topic-based Ability-aware Representation

After getting the word-level representations of job postings and resumes, we further extract more high-level representations for them. As for job postings, we consider that each ability requirement refers to a specific need of a job, and the entire needs of a job can further be summarized from all of its requirements. Following this intuition, we design a hierarchical neural network structure to model such hierarchical representation. At the same time, for resumes, similar hierarchical relationships also exist between candidate experiences and her qualification, thus a similar hierarchical neural network structure is also applied for resumes.

Besides, as we know, both job postings and resumes are documents with relatively well-defined formats. For example, most of candidates tend to separate their past experiences by work contents and order them by time for facilitating understanding. Indeed, such kinds of formats can help us to better extract representations. Thus, to improve the performance and interpretability, we follow the above intuitions and design four attention mechanisms to polish representations extracted by our model at different levels.

Specifically, this component can be further divided into four parts: (1) *Single Ability-aware Part for Job Requirement* for getting the semantic representation of each requirement in a job posting, (2) *Multiple Topic-based Ability-aware Part for Job Requirement* for further extracting entire representation of a job posting, (3) *Single Ability-aware Part for Candidate Experience* for highlighting some experiences in resumes by ability requirements, (4) *Multiple Topic-based Ability-aware Part for Candidate Experience* for finally profiling candidates with all previous experiences. In the following, we will introduce the technical details of each component.

• **Single Ability-aware Part for Job Requirement.** It is obvious that the meaning of a sentence is dominated by several keywords or phrases. Thus, to better capture the key information for each ability requirement, we use an attention mechanism to estimate the importance of each word in it.

This attention layer is the weighted sum of the semantic vector of each word in each ability requirement. Specifically, for lth ability requirement j_l in job posting J, we first use the word representation $\{h^J_{l,t}, \ldots, h^J_{l,m_l}\}$ as input of a fully connected layer and calculate the similarity with word-level context vector. Then, we use a softmax function to calculate the attention score α, i.e.,

$$\begin{aligned}\alpha_{l,t} &= \frac{exp(e^J_{l,t})}{\sum_{i=1}^{m_l} exp(e^J_{l,i})},\\ e^J_{l,t} &= {v_\alpha}^{\mathrm{T}} tanh\Big(W_\alpha h^J_{l,t} + b_\alpha\Big),\end{aligned} \tag{7}$$

where v_α, W_α, and b_α are the parameters to be learned during the training processing. Specifically, v_α denotes the context vector of the j_l, which is randomly initialized. The attention score α can be seen as the importance of each word in j_l. Finally, we calculate the single ability-aware requirement representation s^J_l for j_l by:

$$s^J_l = \sum_{t=1}^{m_l} \alpha_{l,t} h^J_{l,t}. \tag{8}$$

• **Multiple Topic-based Ability-aware Part for Job Requirement.** In this part, we leverage the representations extracted by *Single Ability-aware Part for Job Requirement* to summarize the general needs of jobs. In most jobs, although different ability requirements refer to different specific needs, their importance varies a lot. For example, for recruiting a software engineer, education background is much less important than professional skills.

Moreover, the order of ability requirements in job description will also reflect their importance. In fact, with considering the job posting as a document, each job requirement is a paragraph of this document. Here, we first use a BiLSTM to model the sequential information of ability requirements. Then, we add an attention layer to learn the importance of each ability requirement. Formally, sequential ability representation $\{s^J_1, \ldots, s^J_p\}$, learned in Single Ability-aware in Job Requirement, are used as input of a BiLSTM to generate a sequence of hidden state vectors $\{c^J_1, \ldots, c^J_p\}$, i.e.,

$$c^J_t = BiLSTM\Big(s^J_{1:p}, t\Big), \ \ \forall t \in [1, \ldots, p]. \tag{9}$$

It should be noted that the single ability-aware job requirement representation above only considers the local context between the focus ability. Meanwhile, each job covers multiple abilities. Thus, to better capture the key information for the Job J and learn importance of each ability requirement, we design a topic-based attention mechanism. Specifically, we first use a pre-trained Latent Dirichlet Allocation (LDA) model to extract the topic distribution z^J of the job J. Then, we calculate the importance β_t of each ability requirement j_t based on the similarity between its hidden state c^J_t, topic vector z^J of the integral job posting, and the context vector v_β of all the ability requirements, i.e.,

$$\begin{aligned}\beta_t &= \frac{exp(f^J_t)}{\sum_{i=1}^{p} exp(f^J_i)},\\ f^J_t &= {v_\beta}^{\mathrm{T}} tanh\Big(W_\beta c^J_t + U_\beta z^J + b_\beta\Big),\end{aligned} \tag{10}$$

where the parameters W_β, U_β, b_β, and context vector v_β are learned during training stage. Also, $z^J \in \mathbb{R}^{d_1}$, where d_1 is the hyper-parameter indicating the number of topics in the job posting J. Then, a latent multiple ability-aware job requirement vector will be calculated by weighted sum of the hidden state vectors of abilities, i.e.,

$$g^J = \sum_{t=1}^{p} \beta_t c^J_t. \tag{11}$$

Particularly, the attention scores β can greatly improve the interpretability of the model. It is helpful for visualizing the importance of each ability requirement in practical recruitment applications.

• **Single Ability-aware Part for Experience.** Now, we turn to introduce the learning of resume representations. Specifically, when a recruiter examines whether a candidate matches a job, he/she tends to focus on those specific skills related to this job, which can be reflected by the candidate experiences. As shown in Figure 1, for candidate A, considering the fourth job requirement, we will pay more attention to the highlighted "*green*" sentences. Meanwhile, we may focus on the "*blue*" sentences when matching the second requirement. And, we should consider the global information of the candidate experiences R, which will help us extract key information from these highlights. Thus, we design a novel topic-based ability-aware attention mechanism to qualify the ability-aware contributions of each word in candidate experience to a specific ability requirement.

Formally, for the lth candidate experience r_l, its word-level semantic representation is calculated by a BiLSTM. And, we use an attention-based relation score $\gamma_{l,k,t}$ to qualify the ability-aware contribution of each semantic representation $h^R_{l,t}$ to the kth ability requirement j_k. It can be calculated by

$$\begin{aligned} \gamma_{l,k,t} &= \frac{exp(e^R_{l,k,t})}{\sum_{i=1}^{n_l}\sum_{j=1}^{p} exp(e^R_{j,k,i})}, \\ e^R_{l,k,t} &= v_\gamma^{\mathrm{T}} tanh\Big(W_\gamma s^J_k + U_\gamma h^R_{l,t} + b_\gamma\Big), \end{aligned} \tag{12}$$

where the $W_\gamma, U_\gamma, v_\gamma, b_\gamma$ are parameters. s^J_k is the semantic vector of ability requirement j_k, which is calculated by Equation (8).

Finally, the single ability-aware candidate experience representation is calculated by the weighted sum of the word-level semantic representation of r_l, i.e.,

$$s^R_{l,k} = \sum_{t=1}^{n_l} \gamma_{l,k,t} h^R_{l,t}. \tag{13}$$

Here, the attention score γ further enhances the interpretability of TAPJFNN. It enables us to understand whether and why a candidate is qualified for an ability requirement. We will give a deep analysis in the experiments.

• **Multiple topic-based Ability-aware Part for Experience.** For a candidate, her ordered experiences can reveal her growth process well and such temporal information can also benefit the evaluation on her abilities. To capture such temporal relationships between experiences, we leverage another BiLSTM. Specifically, we first accumulate the single ability-aware candidate experience representation to generate the latent semantic vector s^R_l for lth candidate experience r_l, i.e.,

$$s^R_l = \sum_{t=1}^{p} s^R_{l,t}. \tag{14}$$

Now, we get a set of semantic vectors for candidate experiences, i.e., $\{s^R_1, \ldots, s^R_q\}$. Considering there exist temporal relationships among $\{s^R_1, \ldots, s^R_q\}$, we use a BiLSTM to chain them, i.e.,

$$c^R_t = BiLSTM\Big(s^R_{1:q}, t\Big), \quad \forall t \in [1, \ldots, q]. \tag{15}$$

Finally, considering the topic distribution z^R of resume R and topic distribution z^J of job posting J, we use the weighted sum of the hidden states $\{c^R_1, \ldots, c^R_q\}$ to generate the multiple topic-based ability-aware candidate experience representation g^R, i.e.,

$$\begin{aligned} \delta_t &= \frac{exp(f_t^R)}{\sum_{i=1}^{q} exp(f_i^R)}, \\ f_t^R &= v_\delta^{\mathrm{T}} tanh\Big(W_\delta g^J + U_\delta c_t^R + M_\delta z^J + V_\delta z^R + b_\delta\Big), \\ g^R &= \sum_{t=1}^{q} \delta_t c_t^R, \end{aligned} \tag{16}$$

where the W_δ, U_δ, M_δ, V_δ, and b_δ are parameters to be learned during the training processing, and $z^R \in \mathbb{R}^{d_2}$.

4.3 Person-Job Fit Prediction

With the process of Hierarchical Ability-aware Representation, we can jointly learn the representations for both job postings and resumes. To measure the matching degree between them, we first propose the *Preliminary Person-Job Fit Prediction*. Then, we design a *Refinement Strategy for Person-Job Fit Prediction* to further improve the performance.

• **Preliminary Person-Job Fit Prediction.** Here, we treat the outputs of the hierarchical ability-aware representation as input and apply a comparison mechanism based on a fully connected network to learn the overall Person-Job Fit representation D for predicting the label $\widetilde{y}$ by a logistic function. In addition, for using the concatenation to combine the information of job posting and resumes, we use two matching heuristics including element-wise difference and element-wise product of two representation vectors. These heuristics are certain measures of "similarity" that consider both distance and angle of representation vectors. Similar to Mou et al. [2015] and Tai et al. [2015], we use the combination of both "similarity" measures for better experimental results. The mathematical definition is as follows:

$$\begin{aligned} D &= tanh(W_d[g^J; g^R; g^J - g^R; g^J \odot g^R; z^J; z^R; (W_J z^J + b_J) \odot (W_R z^R + b_R)] + b_d), \\ \widetilde{y} &= Sigmoid(W_y D + b_y), \end{aligned} \tag{17}$$

where W_d, b_d, W_y, b_y, W_J, b_J, W_R, b_R are the parameters to tune the network and $\widetilde{y} \in [0, 1]$. We also use $\odot$ to denote element-wise product. Meanwhile, we minimize the binary cross entropy to train our model.

• **Refinement Strategy for Person-Job Fit Prediction.** Although the above model can already predict the matching degree between job and candidate, it does not make full use of the historical recruitment records. Specifically, during each iteration, the model only considers the job posting and resume in the current application instead of using the historical applications for the current job in each training step. Meanwhile, the resumes of previous job seekers can intuitively help to enrich the information of job requirement. Therefore, we design a Refinement Strategy for Person-Job Fit Prediction with modeling the difference between the representations of those resumes and the resume in the current application, to enhance the performance of Person-Job Fit Prediction. Specifically, we first train the Preliminary Person-Job Fit Prediction. After the model converges, we get the representation g^J for each job posting J, as well as the representation g^R of the resume R that applied for the job J. Now, given a job posting J and a resume R, we first randomly select the K resumes (excluding R to prevent information leakage) of successful and unsuccessful candidates applying for job J, respectively. We denote these two representation matrices as $g_{+,1:K}^{J,R} = [g^{R_1}; g^{R_2}; \ldots; g^{R_k}]$ and $g_{-,1:K}^{J,R} = [g^{R_1'}; g^{R_2'}; \ldots; g^{R_K'}]$, where $R_i, i \in [1, K]$ are the resumes of the successful applications for the job J, and $R_i', i \in [1, K]$ are the unsuccessful candidates' resumes.

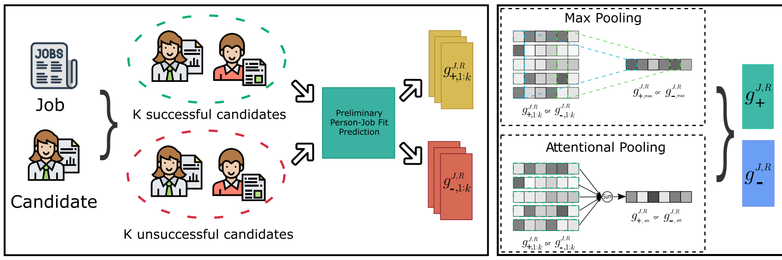


Fig. 4. An illustration of the calculations of $g_{+}^{J,R}$ and $g_{-}^{J,R}$.

Then, we use the pooling layers to convert $g_{+,1:K}^{J,R}$ and $g_{-,1:K}^{J,R}$ to two vectors $g_{+}^{J,R}$ and $g_{-}^{J,R}$, respectively. Here, we introduce two kinds of pooling layer, including the max pooling and attentional pooling. When using the max pooling, we have

$$\begin{aligned} g_{+,max}^{J,R} &= \left[max\left(g_{+,1:K,1}^{J,R}\right); max\left(g_{+,1:K,2}^{J,R}\right); \ldots ; max\left(g_{+,1:K,d^R}^{J,R}\right)\right], \\ g_{-,max}^{J,R} &= \left[max\left(g_{-,1:K,1}^{J,R}\right); max\left(g_{-,1:K,2}^{J,R}\right); \ldots ; max\left(g_{-,1:K,d^R}^{J,R}\right)\right], \end{aligned} \tag{18}$$

where d^R is the dimension of each representation vector $g_{+,i}^{J,R}, g_{-,i}^{J,R}, i \in [1, K]$.

Since those historical representations of resumes may have different effects on predicting the matching degree, we also design an attentional pooling to calculate $g_{+,att}^{J,R}$ as follows:

$$\begin{aligned} g_{+,att}^{J,R} &= \sum_{t=1}^{K} \kappa_t g^{R_t}, \\ \kappa_t &= \frac{exp(e_t^+)}{\sum_{i=1}^{K} exp(e_t^+)}, \\ e_t^+ &= v_\kappa{}^{\mathrm{T}} tanh(W_\kappa g^{R_t} + b_\kappa), \end{aligned} \tag{19}$$

where v_κ, W_κ, and b_κ are the parameters. Similarly, we can get $g_{-,att}^{J,R}$ with attentional pooling layer. Then, we concatenate the max pooling and attentional pooling result to get $g_{+}^{J,R}$ and $g_{-}^{J,R}$.

$$\begin{aligned} g_{+}^{J,R} &= W_{c^+}\left[g_{+,max}^{J,R}, g_{+,att}^{J,R}\right] + b_{c^+}, \\ g_{-}^{J,R} &= W_{c^-}\left[g_{-,max}^{J,R}, g_{-,att}^{J,R}\right] + b_{c^-}, \end{aligned} \tag{20}$$

where W_{c^+}, W_{c^-}, b_{c^+}, b_{c^-} are the parameters, and $g_{+}^{J,R}, g_{-}^{J,R} \in \mathbb{R}^{d^R}$.

Finally, we leverage a fully connected network similar to Equation (17) to fine-tune Person-Job Fit. Specifically, we involve the historical application information of job J through $g_{+}^{J,R}$ and $g_{-}^{J,R}$.

$$\begin{aligned} &D' = tanh(W_d'[g^J; g^R; g^J - g^R; g^J \odot g^R; g_{+}^{J,R} \odot g^R; g_{-}^{J,R} \odot g^R; z^J; z^R; (W_J' z^J + b_J') \odot (W_R' z^R + b_R')] + b_d'), \\ &\widetilde{y} = Sigmoid(W_y' D' + b_y'). \end{aligned} \tag{21}$$

Note that we only update the parameters contained in Equation (21), i.e., W_d', b_d', W_y', b_y', W_J', b_J', W_R', b_R', to fine-tune Person-Job Fit. In fact, we can use Equation (21) to train the entire network, including the representations in the preliminary model. However, it will cause the time complexity of the entire TAPFJNN network training to become $2K + 1$ times of the current training strategy, which is intolerable.

5 APPLICATION

In this section, we will introduce two Person-Job Fit applications enabled by TAPJFNN, i.e., **Talent sourcing** and **Job Recommendation**. In addition, we will present a novel learning algorithm in Job Recommendation to figure out the problem of learning from natural scarcity of negative instances.

5.1 Talent Sourcing

Talent sourcing is the process of finding suitable candidates for a specific job posting. Intuitively, we can train our TAPFJNN on the historical job application data and get the model $\mathcal{M}$. Then, for a specific job posting J, given a candidates set $\{R_1, R_2, \ldots, R_n\}$, we can estimate the probability of success for each candidate $R_i,\ i \in [1, n]$ by $M(J, R_i)$. After that, we can find suitable talents by sorting $M(J, R_i)$.

5.2 Job Recommendation

Job Recommendation is another important application in Person-Job Fit. It is the process of recommending a list of appropriate job postings for the job seeker. Since the job recommendation mainly helps the job seekers to find the suitable jobs that match their abilities, a failed application can only indicate that the job seeker has a gap compared to other candidates in the same period. In other words, it does not necessarily mean that this candidate should not be recommended for this position. Therefore, different with talent sourcing, we cannot treat the failed applications as the negative samples to train our TAPFJNN model.

Actually, this problem also called the on-class problem, because of the lack of negative feedback while only positive feedback (i.e., apply) is available. To address this special semi-supervised [Wang et al. 2016] issue, several approaches use sample negative feedback from the missing data with different sampling methods. For instance, uniformly sampling is one of the most common methods in a recommender system [He et al. 2017]. The non-uniform sampling strategy takes the confidence whether the unobserved samples are indeed negative ones into consideration, such as user-oriented ones [Pan et al. 2008]. Here, we design a new sampling method for our job recommendation scenario.

Formally, let M and N denote the number of candidates and job postings from the implicit feedback of successful job applications $\mathcal{S}^+$, respectively. We define the candidate-job interaction matrix $\mathbf{Y} \in \mathbb{R}^{M \times N}$ from candidates' implicit feedback as,

$$y_{i,j} = \begin{cases} 1, & \text{if application (candidate } i,\ \text{job } j\text{) is observed in } \mathcal{S}^+; \\ 0, & \text{otherwise,} \end{cases} \tag{22}$$

where $y_{i,j} = 1$ indicates that candidate i and job posting j is a successful job application in $\mathcal{S}^+$; however, $y_{i,j} = 0$ does not necessarily mean job posting j is not relevant to candidate i. Intuitively, we can treat all the $y_{i,j} = 0$ as the negative instances to train our TAPFJNN. And, due to the imbalance of label y, we can use the under-sampling method to randomly select negative instances before each iteration of training.

However, it provides noisy signals about candidates' preference, which leads to the negative impact on model training. To overcome the challenge of learning from the scarcity of negative instances, we design a new learning algorithm to train our model for job recommendation.

• **Learning TAPJFNN with Noisy Labels.** Generally, for candidate i, each position j has a probability that is suitable for recommendation, which is denoted by $p^*_{i,j}$. Clearly, we have $p^*_{i,j^+} = 1$ if candidate i and job posting j^+ is a successful job application in $\mathcal{S}^+$. Here, we estimate $p^*_{i,j}$ based on a simple but reliable assumption, i.e., a higher degree of similarity between a job j and the successful position j^+ of the job seeker i leads to a larger value $p^*_{i,j}$. Therefore, we can get the corresponding $p^*_{i,j}$ by estimating the confusion matrix of semantic similarity between job postings,

$$p^*_{i,j} = Q_{j^+,j}, \quad \text{if interaction (candidate } i, \text{ job } j^+) \text{ is observed in } \mathcal{S}^+, \tag{23}$$

where $Q \in \mathbb{R}^{N \times N}$ is the confusion matrix between job postings.

To estimate the confusion matrix Q, we design a cluster-based method. We first use k-Means clustering [MacQueen et al. 1967] to group all the job postings into K_c clusters. Then, we calculate Q as follows:

$$Q_{ij} = \begin{cases} 1, & i = j; \\ K_\alpha, & i \neq j, i \text{ and } j \text{ in the same clusters;} \\ \frac{K_\beta - K_\gamma}{K_{min} - K_{max}} d(c_i, c_j) + \frac{K_\gamma K_{min} - K_\beta K_{max}}{K_{min} - K_{max}}, & i \text{ and } j \text{ not in the same clusters,} \end{cases} \tag{24}$$

where c_i, c_j are the cluster centers of candidate i, j, respectively; $d(\cdot)$ is used to calculate Euclidean distance; K_{max} and K_{min} are the maximum and minimum Euclidean distance of all two different cluster centers, respectively; $K_\alpha, K_\beta, K_\gamma \in [0, 1]$ are hyper-parameters.

After we get the confusion matrix Q, the loss can be defined as

$$\begin{aligned} \mathcal{L}(\Theta) &= \frac{1}{N'} \sum_{n=1}^{N'} -\left(p^*_{i_n,j_n} log(p(\widetilde{y}_{i_n,j_n} = 1|\Theta, i_n, j_n)) + (1 - p^*_{i_n,j_n}) log(p(\widetilde{y}_{i_n,j_n} = 0|\Theta, i_n, j_n))\right) \\ &= \frac{1}{N'} \sum_{n=1}^{N'} -\left(Q_{j^+_n,j_n} log(p(\widetilde{y}_{i_n,j_n} = 1|\Theta, i_n, j_n)) + (1 - Q_{j^+_n,j_n}) log(p(\widetilde{y}_{i_n,j_n} = 0|\Theta, i_n, j_n))\right), \end{aligned} \tag{25}$$

where the N' represents the instances size; j^+_n is position that candidate i_n has successfully applied for; $\widetilde{y}_{i_n,j_n}$ is the predicted output of our TAPFJNN model, which is parameterized by Θ (network weights and biases). The overall learning process of our model in job recommendation is described in Algorithm 1.

6 EXPERIMENTS

In this section, we will introduce the extensive experiments conducted on a real-world recruitment dataset. Specifically, the following questions will be answered by the experimental results.

- **Question 1.** How does the proposed TAPFJNN approach perform compared with other state-of-the-art algorithms?
- **Question 2.** Is the proposed cluster-based algorithm able to handle the problem of learning from natural scarcity of negative instances in job recommendation?
- **Question 3.** How does the proposed TAPFJNN model involve the non-textual features? Do there exist some non-textual features that may impair the fairness of algorithm?
- **Question 4.** How does the proposed TAPFJNN approach achieve the interpretable Person-Job Fit result?

6.1 Data Description

In this article, we conducted our validation on a real-world dataset, which was provided by Baidu Inc., a leading high-tech company in China. To protect the privacy of candidates, all the job application records were anonymized by deleting sensitive personal information.

ALGORITHM 1: Framework of learning TAPFJNN for job recommendation.

Input: The set of candidates $\mathcal{R} = \{R1, R_2, \ldots, R_M\}$, the set of job postings $\mathcal{J} = \{J_1, J_2, \ldots, J_N\}$, the candidate-job interaction matrix Y, Hyper-parameters K_c, $K_\alpha, K_\beta, K_\gamma$, batch size N' and negative under-sampling ratio $\mathcal{N}$

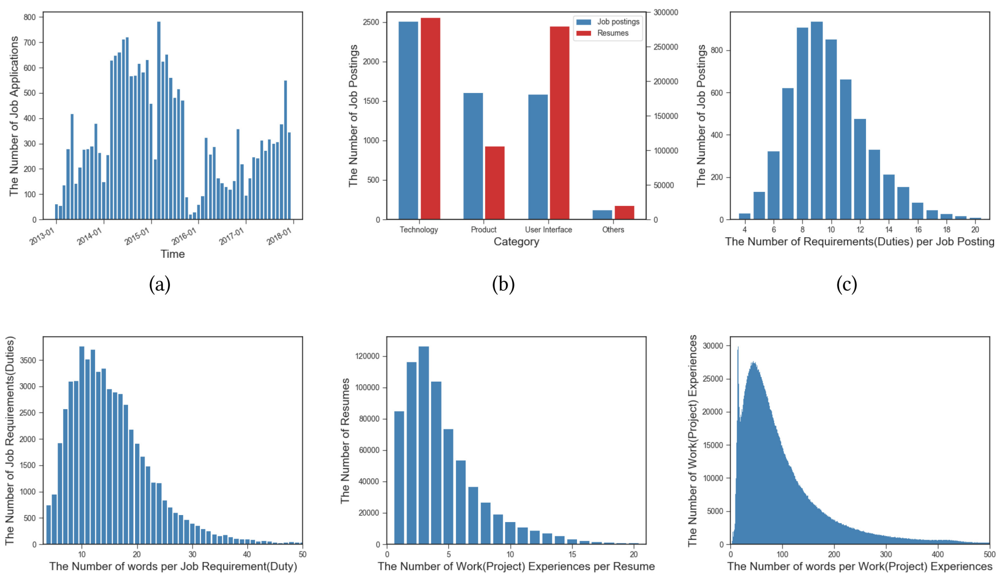
Output: Model parameters Θ

1: Let $\mathcal{A}$ be an SGD-like stochastic optimization algorithm, such as Adam [Kingma and Ba 2014];
2: Group $\{J_1, J_2, \ldots, J_N\}$ into K_c clusters by using K-Means, calculate the cluster centers $\{c_1, c_2, \ldots, c_K\}$;
3: Calculate confusion matrix Q by using Equation (24);
4: **while** Not meet the stopping criterion **do**
5: Randomly select $\mathcal{N}M$ candidate-job pair with satisfying $\{y_{i,j} = 0, \forall i \in \mathcal{R}, j \in \mathcal{J}\}$ without replacement, and combined with the successful candidate-job pair to get the training instances set $\mathcal{X}$.
6: Shuffle $\mathcal{X}$ into $\left\lceil \frac{(\mathcal{N}+1)M}{N'} \right\rceil$ mini-batches and denote by $\mathcal{X}^i$ the ith mini-batch;
7: **for** $i = 1$ to $\left\lceil \frac{(\mathcal{N}+1)M}{N'} \right\rceil$ **do**
8: Calculate the Equation (25) and set gradient $\triangledown_\Theta \mathcal{L}(\Theta)$;
9: Update Θ by $\mathcal{A}$ with its current step size η;
10: **end for**
11: **end while**

In total, we collected *2,301,125* job applications with a range of several years.[3] The time distribution of successful job applications is shown in Figure 5(a). After removing those incomplete job postings and resumes (e.g., resumes without any experience records, job postings without any job requirements), we finally got *1,495,166* job applications data, which consist of *20,184* successful applications and *1,474,982* failed ones. Noted that we treated all the candidates who receive a job offer as successful applicants and do not need to accept it. Correspondingly, we found *5,804* job postings and *696,251* resumes totally. We find that only about 1.3% of applications were accepted, which leads to a typical imbalanced situation and highlights the difficulty of talent recruitment. To a certain degree, this phenomenon may also validate the practical value of our work, as the results of Person-Job Fit may help both recruiters and job seekers to enhance the success rate. Some basic statistics of the pruned dataset are summarized in Table 2. What should be noted is that it is reasonable to have more applications than the number of resumes, since one candidate can apply for several positions at the same time, which is mentioned above.

Moreover, there are four categories of job postings in our collected dataset: *Technology*, *Product*, *User Interface (UI)*, and *Others*. Figure 5(b) summarizes the distribution of job postings and corresponding applied resumes according to different categories. Clearly, most of the applications are technology-oriented. In addition, Figure 6 demonstrates the different recruitment demands of these four kinds of job postings.[4] We can observe that technical position is more concerned with specific professional skills requirements, such as "programming," "algorithm," and "machine learning." Meanwhile, product-oriented positions like product manager are responsible for the development of products for an organization and work to define the business strategy behind a product, requiring some non-professional abilities such as "logical thinking" and "innovation." At the same time, we pay more attention to the visual design capabilities of UI designers. Furthermore, because sales-related positions are included in other positions, relevant skills are highlighted.

[3] Compared to our previous work, we collected a dataset with a larger time span and collected textual data on job duties.
[4] All the words are originally in Chinese. We automatically translated them by using a commercial translation tool: http://api.fanyi.baidu.com/api/trans/product/index.



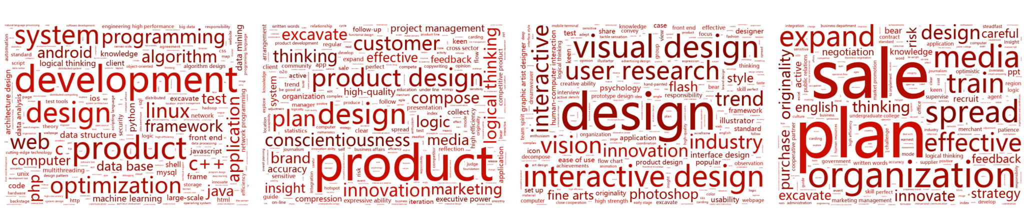
(a) (b) (c)

(d) (e) (f)

Fig. 5. (a): The time distribution of successful job applications. (b): The distribution of different categories w.r.t. job posting and resume, respectively. (c): The distribution of job requirements (duties). (d): The words distribution of job requirement (duty). (e): The distribution of candidate experiences. (f): The words distribution of candidate experience.

Table 2. The Statistics of the Dataset

Statistics	Values
# of job postings	5,804
# of resumes	696,251
# of successful applications	20,184
# of failed applications	1,474,982
Average job requirements(duties) per posting	9.784
Average project/work experiences per resume	4.556
Average words per job requirement	15.450
Average words per project/work experience	106.678



(a) Technology. (b) Product. (c) User Interface. (d) Others.

Fig. 6. The word cloud representation related to four kinds of job postings in our dataset, where the size of each keyword is proportional to its frequency.

6.2 Experimental Setup

Here, we introduce the detailed settings of our experiments, including the technique of word embedding, parameters for our TAPJFNN, as well as the details of training stage.

• **Word Embedding.** First, we explain the embedding layer, which is used to transfer the original *"bag of words"* input to a dense vector representation. In detail, we first used the Skip-gram model to pre-train the word embedding from all the job postings and candidates' resumes of our collected dataset. Then, we utilized the pre-trained word embedding results to initialize the embedding layer weight W_e, which was further fine-tuned during the training processing of TAPJFNN. Specifically, the dimension of word vectors was set to 256.

• **TAPJFNN Setting.** In TAPJFNN model, according to the observation in Figures 5(c), 5(d), 5(e), and 5(f), we set the maximum number of job requirements in each job posting as 18, the same with the constraint of candidate experiences in each resume. Then, the maximum number of words in each requirement/experience was set as 30 and 300, respectively. Along this line, the excessive parts were removed. Also, the dimension of hidden state in BiLSTM was set as 200 to learn the word-level joint representation and requirement/experience representation. Besides, we used two pre-trained LDA models to extract the topic distribution for job requirement and experience. Here, we used all the job postings and resumes as the training data, respectively. We set topic numbers as $d_1 = 50$ and $d_2 = 150$ in our experiments. The dimensions of parameters to calculate the attention score α, β, γ, and δ were set as 200. Finally, the hyper-parameter K in refinement strategy was set as 7.

• **Training Setting.** To achieve better convergence result of our model, following the idea in Glorot and Bengio [2010], we initialized all the matrix and vector parameters in our TAPJFNN model with uniform distribution in $[-\sqrt{6/(n_{in} + n_{out})}, \sqrt{6/(n_{in} + n_{out})}]$, where n_{in}, n_{out} denote the number of the input and output units, respectively. Also, models were optimized by using the Adam [Kingma and Ba 2014] algorithm. Moreover, we set batch size as 32 for training and further used the dropout layer with the probability 0.8 to prevent overfitting.

6.3 Baseline Methods

To validate the performance of our TAPJFNN model, several state-of-the-art supervised models were selected as baseline methods, including the classic supervised learning methods such as *Logistical Regression* (**LR**), *Decision Tree* (**DT**), *Adaboost* (**AB**), *Random Forests* (**RF**), and *Gradient Boosting Decision Tree* (**GBDT**). For these baselines, we used two kinds of input features to construct the experiment, separately.

- **Bag-of-words vectors.** We first created the bag-of-words vectors of ability requirements and candidate experiences, respectively, where the ith dimension of each vector is the frequency of the ith word in dictionary. Then, two vectors were spliced together as input.
- **Mean vector of word embedding.** We respectively averaged the pre-trained word vector of the requirements and experiences and then spliced them as model input.

Besides, we also propose an RNN-based model called **B**asic **P**erson-**J**ob **F**it **N**eural **N**etwork (**BPJFNN-RNN**) as baseline, which can be treated as a simplified version of our TAPJFNN model. The structure of the BPJFNN-RNN model is shown in Figure 7. To be specific, in this model, two BiLSTM models are used to get the semantic representation of each word in requirements and experiences. What should be noted is that, here, we treat all the ability requirements in one job posting as a unity, i.e., a "long sentence," instead of separate requirements, so do the experiences in candidate resumes. Then, we add a mean-pooling layer above them to get two semantic vectors s^J,

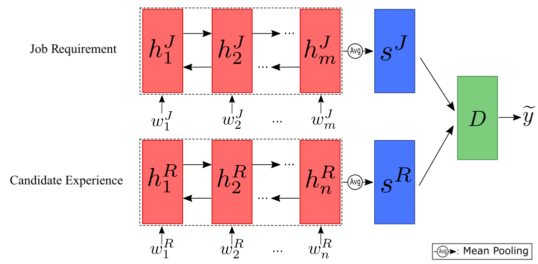


Fig. 7. The illustrations of the proposed RNN-based Basic Person-Job Fit Neural Network.

s^R, respectively. Finally, we can use the following equations to estimate the Person-Job Fit result label $\widetilde{y}$.

$$D = tanh\left(W_d \left[s^J; s^R; s^J - s^R\right] + b_d\right),$$
$$\widetilde{y} = softmax(W_y D + b_y), \tag{26}$$

where the W_d and b_d are the parameters to learn.

Also, we chose two state-of-the-art Person-Job Fit models as the baseline models. One is our preliminary model **APJFNN** [Qin et al. 2018], the other is **PJFNN** [Zhu et al. 2018]. Specifically, PJFNN is a CNN-based Person-Job Fit Neural model with minimizing cosine distance between the semantic representations of job posting and resume in successful job applications and maximizing the fail ones. In addition, *TAPJFNN_preliminary* is a variant of our model that only uses the preliminary Person-Job Fit Prediction.

6.4 Evaluation Metrics

Considering the first application, talent sourcing, since in the real-world process of talent recruitment, we usually have a potential *"threshold"* to pick up those adequate candidates, which results in a certain *"ratio of acceptance."* However, we could hardly determine the acceptance rate properly, as it could be a personalized value that is affected by complicated factors. Thus, to comprehensively validate the performance, we selected the **AUC** index to measure the performance under different situations. Besides, we adopted the **Accuracy**, **Precision**, **Recall**, and **F1-measure** as the evaluation metrics.

To measure the performance in job recommendation, we adopted the commonly used error metric in learning to rank, **Hit Ratio** (HR), which intuitively measures whether the candidate's successfully applied job posting is present on the top-N list of recommendations. And it is defined as:

$$HR@N = \frac{n^+_{test}}{n_{test}}, \tag{27}$$

where the n^+_{test} denotes the number of instances that the successful applied job posting is present on the top-N list in the testing set and n_{test} is the number of instances in the testing set. Without special mention, we truncated $N = 10, 20$ in our experiments.

6.5 Experiment Results in Talent Sourcing

• **Overall Performance Comparisons (Q1).** We conducted the task of Person-Job Fit based on the real-word dataset, i.e., we used the successful job applications as *positive samples* and then used the failed applications as the *negative instance* to train the models. To reduce the impact of imbalances in data, we used the under-sampling method to randomly select negative instances that are equal to the number of positive instances for each job posting to evaluate our model. Along

Table 3. The Performance of TAPJFNN and Baselines

Methods	Accuracy	Precision	Recall	F1	AUC
LR	0.6856	0.6918	0.6695	0.6804	0.7346
AB	0.7351	0.7420	0.7210	0.7313	0.8021
DT	0.7111	0.7741	0.5961	0.6736	0.7561
RF	0.7299	0.7370	0.7151	0.7259	0.7997
GBDT	0.7782	0.7902	0.7577	0.7736	0.8551
LR (with word2vec)	0.6777	0.6870	0.6526	0.6694	0.7426
AB (with word2vec)	0.6635	0.6675	0.6516	0.6595	0.7237
DT (with word2vec)	0.6224	0.6280	0.6006	0.6140	0.6706
RF (with word2vec)	0.6536	0.6672	0.6130	0.6389	0.7163
GBDT (with word2vec)	0.6737	0.6741	0.6725	0.6733	0.7419
BPJFNN-RNN	0.7800	0.7897	0.7631	0.7762	0.8535
PJFNN [Zhu et al. 2018]	0.8045	0.8179	0.7834	0.8003	0.8729
APJFNN [Qin et al. 2018]	0.8273	0.8704	0.7691	0.8166	0.8959
TAPJFNN_preliminary	0.8387	0.8676	0.7993	0.8321	0.9091
TAPJFNN	**0.8508**	**0.8774**	**0.8156**	**0.8454**	**0.9300**

this line, we randomly selected 80% of the dataset as training data, another 10% for tuning the parameters, and the last 10% as test data to validate the performance.

The performance is shown in Table 3.[5] According to the results, clearly, we realize that our TAPJFNN outperforms all the baselines with a significant margin, which verifies that our framework could well distinguish those adequate candidates with given job postings. Especially, both our TAPJFNN and preliminary model APJFNN perform better than BPJFNN. It seems that our attention strategies can not only distinguish the critical ability/experience for better explanation, but also improve the performance with better estimation of matching results. Since our TAPJFNN model can catch more important information from the job posting and resume textual data by using the hierarchical topic-based attention strategies and leverage the historical recruitment records data by using the Refinement Strategy for Person-Job Fit Prediction, it boosts by 2.35%, 2.88%, and 3.14% for metric Accuracy, F1, and AUC than APJFNN, respectively.

At the same time, we find that almost all the baselines using the Bag-of-Words as input feature outperform those using the pre-trained word vector as input features (i.e., those with "word2vec" in Table 3). This phenomenon may indicate that the pre-trained word vectors are not enough to characterize the semantic features of the recruitment textual data; this is the reason of why we use the BiLSTM above the embedding layer to extract the word-level semantic word representation.

• **The Effectiveness of Different Components.** To demonstrate the role of each component, we gradually removed each component and compared it to our TAPJFNN. Specifically, we constructed variants of our model as follows:

- *TAPJFNN_max* is the variant that only uses max pooling layer in the refinement strategy for Person-Job Fit prediction.
- *TAPJFNN_att* is the variant that only uses attentional pooling layer in the refinement strategy for Person-Job Fit prediction.

[5]Compared with the experiments of Qin et al. [2018], we increased the size of the dataset and additively used the job duties data, which improved the overall performance.

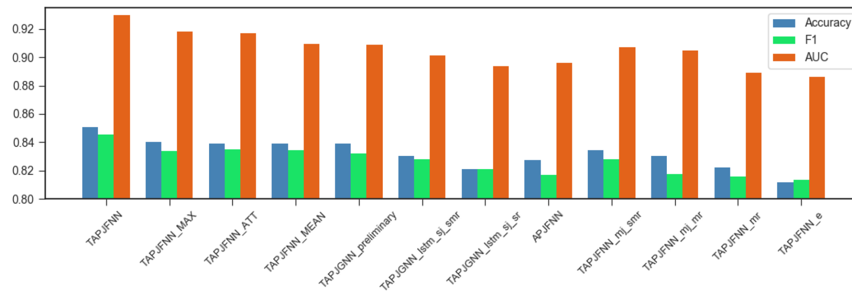
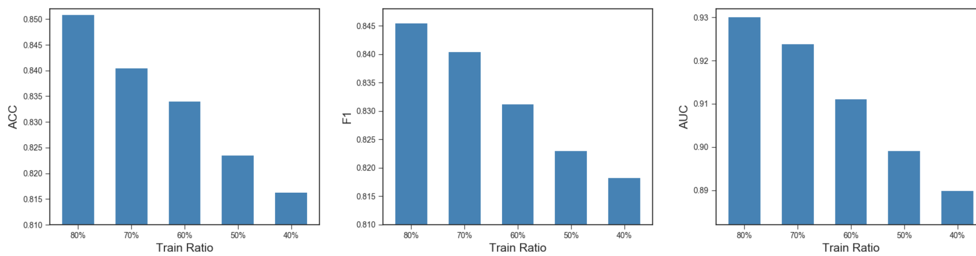


Fig. 8. The performance of TAPJFNN and its variants.

- *TAPJFNN_mean* is the variant that only uses mean pooling layer in the refinement strategy for Person-Job Fit prediction.
- *TAPJFNN_preliminary* is the variant that only uses preliminary Person-Job Fit prediction.
- *TAPJGNN_lstm_sj_smr* is the variant of *TAPJFNN_preliminary*, which removes LSTM in multiple ability-aware job representation.
- *TAPJGNN_lstm_sj_sr* is the variant of *TAPJFNN_preliminary*, which removes LSTMs in multiple ability-aware job and resume representations.
- APJFNN is our previously proposed model. It can be seen as a variant of *TAPJFNN_preliminary*, which removes the effects of topic information.
- *TAPJFNN_mj_smr* is the variant of *TAPJFNN_preliminary*, which further removes the single ability-aware attention mechanism in job requirement.
- *TAPJFNN_smr* is the variant of *TAPJFNN_mj_smr*, which further removes the single ability-aware attention mechanisms in candidate experience.
- *TAPJFNN_mr* is the variant of *TAPJFNN_smr*, which further removes the multiple topic-based ability-aware in job requirement.
- *TAPJFNN_e* is the variant of *TAPJFNN_mr*, which further removes the multiple topic-based ability-aware in candidate experience.

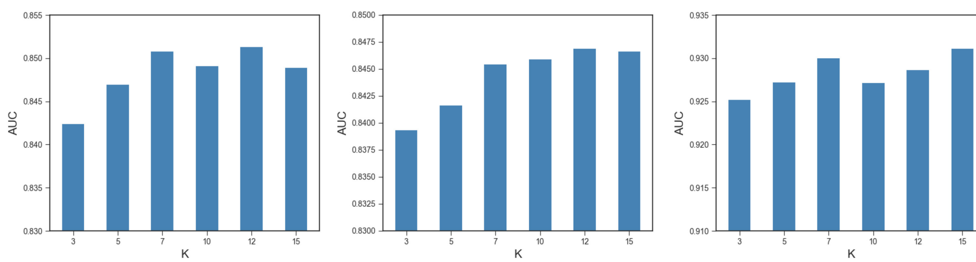
The results are shown in Figure 8. First, we can observe that the TAPJFNN gets a boost of 2.9% for the metric AUC than the *TAPJFNN_preliminary*, which clearly verifies the effectiveness of our refinement strategy for Person-Job Fit prediction. We also find that only using max pooling layer or attentional pooling layer reduces the AUC about 1.2%~1.3%. In addition, because using the mean pooling layer does not have a significant improvement, we did not involve it in our refinement strategy. Second, we find that gradually removing LSTM from the multiple ability-aware representations of job and resume significantly reduces the performance. Besides, the *TAPJFNN_preliminary* outperforms our original model APJFNN by a boost of 1.14%, 1.55%, and 1.32% for metric Accuracy, F1, and AUC, respectively. It shows the effectiveness of involving the topic information by using the LDA model. Last, as the single and multiple ability-aware attention mechanisms are removed progressively, the performance is getting worse, which clearly demonstrates each component in two hierarchical topic-based ability attention strategies.

• **The Robustness on Different Data Split.** To observe how our model performs at different train/test split, we randomly selected 80%, 70%, 60%, 50%, 40% of the dataset as training set, another 10% for testing set, and the rest for tuning the parameters. The results are shown in Figures 9(a), 9(b), and 9(c). We can observe that the overall performance of our model is relatively stable, while it gets better as the training data increases. Indeed, the improvements of the best performance compared with the worst one are only 3.47%, 2.72%, and 4.02% for three metrics, respectively.



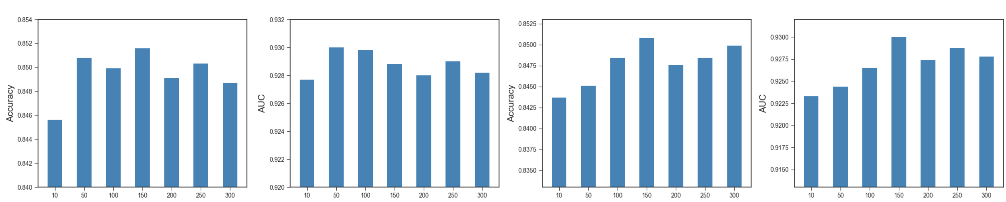
(a) The Accuracy performance. (b) The F1 performance. (c) The AUC performance.

Fig. 9. The performance of TAPJFNN at different train/test split.



(a) The Accuracy performance. (b) The F1 performance. (c) The AUC performance.

Fig. 10. The performance of TAPJFNN with different K in fine tuning Person-Job Fit.



(a) (b) (c) (d)

Fig. 11. (a) and (b) show the Accuracy/AUC performance of TAPJFNN with different topic number of job postings d_1. (c) and (d) show the Accuracy/AUC performance of TAPJFNN with different topic number of candidates' experiences d_2.

Furthermore, we find that our model with 60% of data for training has already outperformed all the baselines methods, which used 80% of the data for training. The results clearly validate the robustness of our model in terms of training scalability.

• **Impact of Hyper-parameter K.** The hyper-parameter K controls the number of successful/unsuccessful applications when using the refinement strategy for Person-Job Fit prediction. We evaluated the impact of performance with different K ranging from 3 to 15, which is shown in Figure 10. We can observe that when $K = 7$, the performance is good enough, which will be difficult to further improve when enlarging the value of K.

• **Impact of Topic Number d_1, d_2.** The d_1, d_2 values control the topic number when using the topic-based attention mechanisms. Here, we separately evaluated the effect of d_1, d_2 by fixing $d_1 = 50$ or $d_2 = 150$ and setting another value range from 10 to 300. The result is shown in Figure 11. We can observe that when $d_1 \geq 50$, the performance of our model does not change significantly

Table 4. The Performance of TAPJFNN and Baselines

Methods	HR@10	HR@20
ItemPop	0.2418	0.3198
BPJFNN-RNN	0.2695	0.3476
PJFNN [Zhu et al. 2018]	0.2734	0.3722
APJFNN [Qin et al. 2018]	0.3149	0.4156
TAPJFNN_preliminary	0.3291	0.4430
TAPJFNN	0.3493	0.4557
TAPJFNN-K	**0.4076**	**0.5443**

with increasing the value of d_1. Meanwhile, we find that the performance is not good when d_2 is smaller, i.e., $d_2 = 10$. Thus, we adopted a large value for d_2 for better result, and $d_2 = 150$ seems to be good enough.

6.6 Experiment Results in Job Recommendation

• **Overall Performance Comparisons (Q1,Q2).** The original job application data contain four different categories of job posting. Without loss of generality, we use the technical data to construct the experiment. And, we filtered the dataset to make each job posting have at least 10 corresponding candidates who have successfully applied it. Finally, we collected 3,957 candidates' resumes and 189 job postings to construct the successful job applications dataset. Along this line, we randomly selected 80% of the candidates' in the dataset as the training set, another 10% to tune the parameters, and the last 10% as the test data. Meanwhile, with the help of domain experts, we labeled the ground-truth of the test data to verify the performance. Also, we involved a baseline called **ItemPop**, which ranks the job posting by their popularity judged by the number of interactions of the job postings and the candidates.

The overall performance is shown in Table 4. Clearly, we find our TAPJFNN performs the best compared with other baselines. We also observed that TAPJFNN-K performs better than the basic TAPFJNN, which set the negative under-sampling ratio $\mathcal{N} = 3$, $K_c = 40$, $K_\alpha = 0.95$, $K_\beta = 0.1$, and $K_\gamma = 0$.

6.7 How to Involve the Non-textual Features?

• **Modeling the non-textual features.** Our model is friendly to involve the non-textual features. Here, we denote non-textual features of the job posting and resumes as $o^J \in \mathbb{R}^{d_4}$ and $o^R \in \mathbb{R}^{d_5}$, respectively. Similar to the refinement strategy in person-job fit prediction, we collected the non-textual features of K resumes of the successful/unsuccessful candidates applying for job J, respectively, which denote as $o^{J,R}_{+,1:K}$ and $o^{J,R}_{-,1:K}$. Then, we can calculate the vectors $o^{J,R}_{+,max}$, $o^{J,R}_{+,att}$, $o^{J,R}_{-,max}$, and $o^{J,R}_{-,att}$ from $o^{J,R}_{+,1:K}$ and $o^{J,R}_{-,1:K}$ with max pooling and attentional pooling layers like Equation (18), (19), and concatenate them as $o^{J,R}_{+}$ and $o^{J,R}_{-}$ by using a full connection layer as Equation (20). Then, we adjust the Equation (21) to:

$$\begin{aligned} D' = tanh(W_d'[o^J; o^R; o^R \odot o^{J,R}_+; o^R \odot o^{J,R}_-; g^J; g^R; g^J - g^R; g^J \odot g^R; \\ g^{J,R}_+ \odot g^R; g^{J,R}_- \odot g^R; z^J; z^R; (W_J' z^J + b_J') \odot (W_R' z^R + b_R')] + b_d'), \\ \widetilde{y} = Sigmoid(W_y' D' + b_y'). \end{aligned} \tag{28}$$

Table 5. The Description of Features

Features	Feature Type	Description	Dimension
Job Postings Categories	Discrete	The category of the job, such as Technical, Product	4
Candidates' Gender	Discrete	Male or Female	2
Candidates' age	Continuous	Age	1
Candidates' Education	Discrete	The highest education of the candidates, such as graduate student, doctoral student	7
Candidates' Graduated School	Discrete	Is candidate graduated from one of the list of Chinese universities in Project 211, and so on.	14

Table 6. The Performance of TAPJFNN and Baselines Using Non-textual Features

Methods	Accuracy	Precision	Recall	F1	AUC
LR	0.6925	0.6930	0.6913	0.6921	0.7429
AB	0.7418	0.7470	0.7314	0.7391	0.8074
DT	0.7111	0.7777	0.5912	0.6717	0.7499
RF	0.7340	0.7367	0.7279	0.7323	0.8045
GBDT	0.7750	0.7832	0.7607	0.7717	0.8577
LR (with word2vec)	0.6757	0.6815	0.6595	0.6704	0.7478
AB (with word2vec)	0.6615	0.6633	0.6561	0.6597	0.7266
DT (with word2vec)	0.6204	0.6172	0.6343	0.6256	0.6652
RF (with word2vec)	0.6583	0.6670	0.6323	0.6492	0.7233
GBDT (with word2vec)	0.6715	0.6703	0.6749	0.6726	0.7475
BPJFNN-RNN	0.7804	0.7894	0.7651	0.7771	0.8546
PJFNN [Zhu et al. 2018]	-	-	-	-	-
APJFNN [Qin et al. 2018]	0.8362	0.8738	0.7859	0.8276	0.9019
TAPJFNN	**0.8556**	**0.8823**	**0.8206**	**0.8503**	**0.9332**

We constructed four kinds of non-textual features, which are job postings categories (T, P, U, O), candidates' gender, age, and their education (highest education, graduated school). Also, we transferred all the discrete features into the one-hot vectors and normalized all the continuous features into a range of [0, 1]. Table 5 shows the details of the features.

Along this line, we added the non-textual features into the baselines by invovling o^J and o^R[6]. Table 6 shows the performance of our model and the baselines with non-textual features in talent sourcing. We can observe that most of the methods including our model have some improvement when adding the non-textual features.

• **How to avoid unfairness of the algorithm when considering the non-textual features.** Admittedly, while the accuracy of the algorithm is essential, another paramount issue that needs to be paid attention to is empowering the correct values of the intelligent recruitment system and ensuring the fairness of the algorithm. In recent years, it has received extensive attention from academics and the media [Dastin 2018]. For any machine learning–based algorithm to become fair, it must first avoid the bias of training data, such as the significant difference in employment ratio of women and men. Unfortunately, for many existing recruitment practices in real life, the prejudices

[6]Because PJFNN [Zhu et al. 2018] uses the cosine distance as the loss function, we cannot add the non-textual features.

seem hard to completely avoid. For example, according to a recent report [Test-doctoring 2018], doctoring has long been a male bastion at the Tokyo Medical University, where they confessed to marking down the test scores of female applications to keep the ratio of women in each class below 30%.

So, in the construction of the intelligent recruitment system, one of the questions that must be answered is that, if we already have a dataset with potential value discrepancy, how can we avoid further misleading the algorithm? Intuitively, if the data with gender bias are used for training machine learning models of intelligent recruitment, *Gender* would be regarded as a dominant feature based on the common feature engineering, since the Chi-squared test result, information gain, or correlation coefficient score indicate whether it has a significant correlation with the recruitment result. Therefore, *Gender* feature is seen as a potential factor affecting the values of the machine learning algorithm itself. In this case, we should not add *Gender* feature to train the model.

To confirm our conjecture, here, we evaluated on semi-synthetic data based on a real-world recruitment system. First, we constructed a "standard dataset" without value discrepancy. Specifically, we randomly selected 5,678 successful job applications (positive instances) from the recruitment records of historical job postings, where half of them are female candidates. Then, for each of the job postings, we also randomly selected the same number of failed job applications (negative instances). In particular, both successful and failed applications satisfy that the numbers of male and female candidates are equal, which means there is no gender bias in the original dataset. Next, in the model validation step, we randomly selected 80% of the dataset as training data, another 10% for tuning the parameters, and the last 10% as test data to validate the performance and robustness. At the same time, to simulate the possible unfairness scenario in the recruitment system, we randomly labeled 50% of successful female applications as negative and labeled 50% of male failed applications as positive ones in the **training set** and **validation set**. After the manual construction, in both training and validation sets, the success rates of male and female candidates become 75% and 25%, respectively. Note that we did not change the labels in **test set**, which has the same cutoff ratio as "standard dataset" for both women and men to ensure it has the correct values.

Table 7 shows the performance on the validation set and testing set of the semi-synthetic data. Clearly, we observe that with adding the Gender feature, each model in validation set has the better performance, since validation set has a similar distribution with training set. However, in other words, those models have unfortunately learned the value bias that existed therein. In contrast, we realize that all the models perform better without using gender information on the testing set, which demonstrates that the models can avoid value deviation from the training data to a great extent without leveraging the Gender information. Therefore, we can conclude that when historical recruitment dataset contains the bias of data distribution, such as gender discrimination, we should not use the corresponding features to train the model, thus avoiding the algorithm to produce value deviations like humans.

6.8 Case Study

With the proposed attention strategies, we target at not only improving the matching performance, but also enhancing the interpretability of matching results. To that end, in this subsection, we will illustrate the matching results in three different levels by visualizing the attention results.

◇ **Word-level: Capturing the key phrases from the sentences of job requirement.**

First, we would like to evaluate whether our TAPJFNN model can reveal the word-level key phrase from long sentences in job requirements. The corresponding case study is shown in Figure 12, in which some words (in Chinese) are highlighted as *key phrases*, and their darkness correlated to the value of attention α.

Table 7. The Performance of TAPJFNN and Baselines on Semi-synthetic Data

Features		Without gender feature					With gender feature				
Methods	Datasets	Accuracy	Precision	Recall	F1	AUC	Accuracy	Precision	Recall	F1	AUC
LR	Validation set	0.5122	0.5126	0.4957	0.5040	0.5348	0.6783	0.6758	0.6852	0.6805	0.7063
	Testing set	0.5855	0.5913	0.5913	0.5913	0.6093	0.5203	0.5281	0.5061	0.5169	0.5693
AB	Validation set	0.5713	0.5724	0.5635	0.5679	0.5847	0.7217	0.7040	0.7652	0.7333	0.7882
	Testing set	0.6402	0.6567	0.6087	0.6318	0.6770	0.5459	0.5549	0.5270	0.5406	0.6274
DT	Validation set	0.5870	0.6179	0.4557	0.5245	0.5951	0.7261	0.7167	0.7478	0.7319	0.7744
	Testing set	0.6711	0.7349	0.5496	0.6289	0.6807	0.5079	0.5159	0.4800	0.4973	0.5701
RF	Validation set	0.5991	0.6096	0.5513	0.5790	0.6118	0.7148	0.6939	0.7687	0.7294	0.7531
	Testing set	0.6279	0.6527	0.5687	0.6078	0.6857	0.5141	0.5211	0.5165	0.5188	0.5807
GBDT	Validation set	0.5913	0.5953	0.5704	0.5826	0.6290	0.7200	0.7030	0.7617	0.7312	0.7945
	Testing set	0.6896	0.7069	0.6626	0.6840	0.7436	0.5194	0.5271	0.5078	0.5173	0.6208
LR (with word2vec)	Validation set	0.5652	0.5693	0.5357	0.5520	0.5985	0.7113	0.6989	0.7426	0.7201	0.7625
	Testing set	0.5873	0.6011	0.5530	0.5761	0.6140	0.5079	0.5150	0.5078	0.5114	0.5642
AB (with word2vec)	Validation set	0.5626	0.5655	0.5409	0.5529	0.5780	0.7217	0.7121	0.7443	0.7280	0.7685
	Testing set	0.5540	0.5647	0.5235	0.5433	0.5860	0.5256	0.5322	0.5322	0.5322	0.5565
DT (with word2vec)	Validation set	0.5313	0.5304	0.5461	0.5381	0.5577	0.7243	0.7067	0.7670	0.7356	0.7435
	Testing set	0.5502	0.5534	0.5861	0.5693	0.5853	0.4929	0.5000	0.5009	0.5004	0.5340
RF (with word2vec)	Validation set	0.5565	0.5610	0.5200	0.5397	0.5756	0.6991	0.6856	0.7357	0.7097	0.7332
	Testing set	0.5847	0.6057	0.5183	0.5586	0.6301	0.5212	0.5282	0.5217	0.5249	0.5387
GBDT (with word2vec)	Validation set	0.5809	0.5841	0.5617	0.5727	0.5983	0.7157	0.7033	0.7461	0.7241	0.7687
	Testing set	0.5970	0.6105	0.5670	0.5979	0.6317	0.5088	0.5157	0.5130	0.5144	0.5587
PJFNN-RNN	Validation set	0.5974	0.6261	0.4835	0.5456	0.6443	0.7183	0.6865	0.8035	0.7404	0.7858
	Testing set	0.6296	0.6824	0.5043	0.5800	0.7034	0.5397	0.5425	0.5878	0.5643	0.6178
APJFNN	Validation set	0.6191	0.6179	0.6243	0.6211	0.6681	0.7157	0.6649	0.8696	0.7536	0.8091
	Testing set	0.7425	0.7386	0.7617	0.7500	0.8036	0.5917	0.5780	0.7217	0.6419	0.6444
TAPJFNN	Validation set	0.6508	0.6346	0.7339	0.6806	0.6926	0.7804	0.7612	0.8261	0.7923	0.8363
	Testing set	0.7707	0.7556	0.8226	0.7877	0.8181	0.6305	0.6114	0.7443	0.6714	0.6738

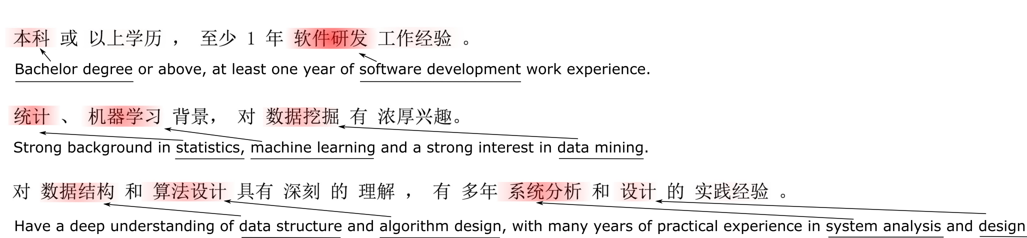


Fig. 12. Two examples for demonstrating the advantage of Attention α in capturing the informed part of the ability requirement sentence.

According to the results, it is unsurprising that the crucial skills are highlighted compared with common words. Furthermore, in the same requirement, different abilities may have different importance. For instance, in the requirement in line 1, the work experience requirement for professional skills could be more important than education requirement, which might be due to *bachelor degree* being usually treated as the basic requirement. Similarly, for the technique-related skills

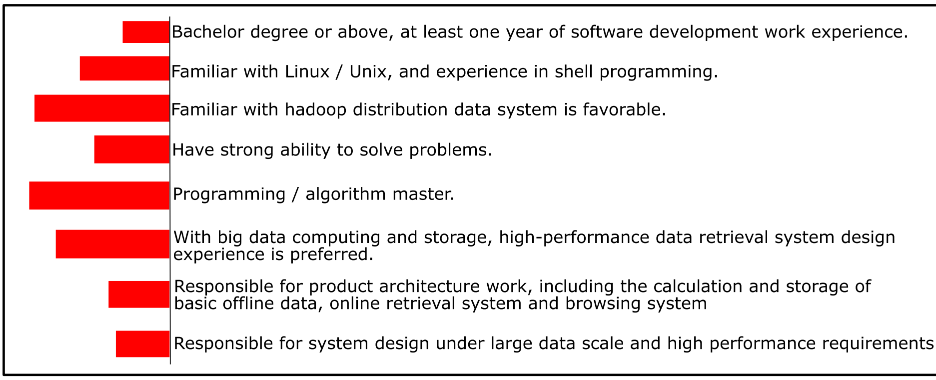


Fig. 13. An example demonstrating the advantage of Attention β in measuring the importance of each ability requirement among all the job needs. The left bar charts denote the distribution of β over all requirements.

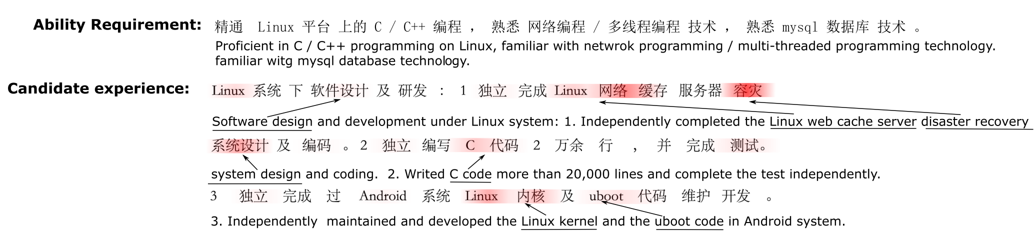


Fig. 14. An example for demonstrating the advantage of Attention γ in capturing the ability-aware informed part from the experience of candidate.

requirements in line 3, *statistics, machine learning* are more important than *data mining*, which might be due to the different degrees ("strong background in" v.s. "strong interest in").

⋄ Ability-level: Measuring the different importance among all abilities.

Second, we would like to evaluate whether TAPJFNN can highlight the most critical abilities. The corresponding case study is shown in Figure 13, in which a histogram indicates the importance of each ability, i.e., the distribution of attention β.

From the figure, a striking contrast can be observed among the eight abilities, in which the *bachelor degree* with the lowest significance is usually treated as the basic requirement. Correspondingly, the ability of *programming/algorithm*, *hadoop*, and experience in big data computing and storage, data retrieval system design could be quite beneficial in practice, which leads to higher importance. Especially *programming/algorithm* achieves the highest significance, which might be due to the strong degree word "master." In other words, the importance of abilities could be measured by the *scarcity*, as most candidates have the bachelor degree, but only a few of them could reach the ability of the programming/algorithm master.

⋄ Matching-level: Understanding the matching between job requirements and candidate experiences.

At last, we would like to evaluate how TAPJFNN model can guide the matching between requirements and experiences. The corresponding case study is shown in Figure 14, in which darkness is also correlated to the importance of experience while considering the different job requirements, i.e., the attention value of γ.

Definitely, we find that those key phrases that could satisfy the requirements are highlighted, e.g., *Linux kernel*, *Linux web cache server disaster recovery system design*, and *C code* for the requirement *Linux*, *C/C++*, and *network programming*, respectively. Also, we realize that the "importance" here indeed indicates the degree of satisfying the requirements. For instance, the phrase *Linux web cache server disaster recovery system design* is darker than *software design and development in Linux system*, since the former one is a concrete project strongly related to the *Linux* and *network programming*, but the latter one is only a rough matching. Thus, this case study also proves that our TAPJFNN method could provide good interpretability for Person-Job Fit task, since key clues for matching the job requirements and candidate experience can be highlighted.

7 CONCLUSION

In this article, we proposed a novel end-to-end Topic-based Ability-aware Person-Job Fit Neural Network (TAPJFNN) model, which has a goal of reducing the dependence on manual labor and can provide better interpretation about the fitting results. The key idea is to exploit the rich information available in abundant historical job application data. Specifically, we first proposed a word-level semantic representation for both job requirements and job seekers' experiences based on Recurrent Neural Network (RNN). Then, two hierarchical topic-based ability-aware attention strategies were designed to measure the different importance of job requirements for semantic representation, as well as measure the different contributions of each job experience to a specific ability requirement. Moreover, we designed a refinement strategy for Person-Job Fit prediction based on historical recruitment records to enhance the performance of predicting the matching degree between the talent and job. We further applied our TAPJFNN framework into two Person-Job Fit applications, including talent sourcing and job recommendation. Specifically, to overcome the challenge of learning from natural scarcity of negative instances, we designed a novel learning algorithm for the model training process of job recommendation task. Finally, extensive experiments conducted on a large-scale real-world dataset clearly validated the effectiveness and interpretability of our TAPJFNN framework compared with several baselines.

REFERENCES

Meenakshi Chaudhary. 2017. LinkedIn by the numbers: 2017 statistics. Retrieved from https://www.linkedin.com/pulse/linkedin-numbers-2017-statistics-meenakshi-chaudhary/.

Tao Chen, Ruifeng Xu, Yulan He, and Xuan Wang. 2017a. Improving sentiment analysis via sentence type classification using BiLSTM-CRF and CNN. *Exp. Syst. Applic.* 72 (2017), 221–230.

Yifan Chen, Pengjie Ren, Yang Wang, and Maarten de Rijke. 2019. Bayesian personalized feature interaction selection for factorization machines. In *Proceedings of the 42nd International ACM SIGIR Conference on Research & Development in Information Retrieval.*

Yifan Chen, Xiang Zhao, and Maarten de Rijke. 2017b. Top-N recommendation with high-dimensional side information via locality preserving projection. In *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval.* ACM, 985–988.

Yu Cheng, Yusheng Xie, Zhengzhang Chen, Ankit Agrawal, Alok Choudhary, and Songtao Guo. 2013. Jobminer: A real-time system for mining job-related patterns from social media. In *Proceedings of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 1450–1453.

Zhiyong Cheng, Ying Ding, Xiangnan He, Lei Zhu, Xuemeng Song, and Mohan S. Kankanhalli. 2018a. Aˆ3NCF: An adaptive aspect attention model for rating prediction. In *Proceedings of the International Joint Conferences on Artificial Intelligence Organization.* 3748–3754.

Zhiyong Cheng, Ying Ding, Lei Zhu, and Mohan Kankanhalli. 2018b. Aspect-aware latent factor model: Rating prediction with ratings and reviews. In *Proceedings of the World Wide Web Conference*. International World Wide Web Conferences Steering Committee, 639–648.

Jin Yao Chin, Kaiqi Zhao, Shafiq Joty, and Gao Cong. 2018. ANR: Aspect-based neural recommender. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management.* ACM, 147–156.

Aichia Chuang and Paul R. Sackett. 2005. The perceived importance of person-job fit and person-organization fit between and within interview stages. *Soc. Behav. Person.: Int. J.* 33, 3 (2005), 209–226.

Jeffrey Dastin. 2018. Amazon scraps secret AI recruiting tool that showed bias against women. Retrieved from https://www.reuters.com/article/us-amazon-com-jobs-automation-insight/amazon-scraps-secret-ai-recruiting-tool-that-showed-bias-against-women-idUSKCN1MK08G.

Mamadou Diaby, Emmanuel Viennet, and Tristan Launay. 2013. Toward the next generation of recruitment tools: An online social network-based job recommender system. In *Proceedings of the IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM'13)*. IEEE, 821–828.

Karen Holcombe Ehrhart. 2006. Job characteristic beliefs and personality as antecedents of subjective person–job fit. *J. Bus. Psych.* 21, 2 (2006), 193–226.

Karen Holcombe Ehrhart and Guido Makransky. 2007. Testing vocational interests and personality as predictors of person-vocation and person-job fit. *J. Career Assess.* 15, 2 (2007), 206–226.

Ahmed Elsafty, Martin Riedl, and Chris Biemann. 2018. Document-based recommender system for job postings using dense representations. In *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 3 (Industry Papers)*. 216–224.

James H. Errico, M. Ibrahim Sezan, George R. Borden, Gary A. Feather, and Mick G. Grover. 2015. Collaborative recommendation system. US Patent 8,949,899.

Xavier Glorot and Yoshua Bengio. 2010. Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the 13th International Conference on Artificial Intelligence and Statistics*. 249–256.

Wael H. Gomaa and Aly A. Fahmy. 2013. A survey of text similarity approaches. *Int. J. Comput. Applic.* 68, 13 (2013).

Alex Graves and Jürgen Schmidhuber. 2005. Framewise phoneme classification with bidirectional LSTM and other neural network architectures. *Neur. Netw.* 18, 5–6 (2005), 602–610.

Jiafeng Guo, Yixing Fan, Qingyao Ai, and W. Bruce Croft. 2016. A deep relevance matching model for ad hoc retrieval. In *Proceedings of the 25th ACM International on Conference on Information and Knowledge Management*. ACM, 55–64.

Christopher G. Harris. 2017. Finding the best job applicants for a job posting: A comparison of human resources search strategies. In *Proceedings of the International Conference on Data Mining Workshops*. IEEE, 189–194.

Xiangnan He, Zhankui He, Xiaoyu Du, and Tat-Seng Chua. 2018. Adversarial personalized ranking for recommendation. In *Proceedings of the 41st International ACM SIGIR Conference on Research & Development in Information Retrieval*. ACM, 355–364.

Xiangnan He, Lizi Liao, Hanwang Zhang, Liqiang Nie, Xia Hu, and Tat-Seng Chua. 2017. Neural collaborative filtering. In *Proceedings of the 26th International Conference on World Wide Web*. International World Wide Web Conferences Steering Committee, 173–182.

Karl Moritz Hermann and Phil Blunsom. 2013. The role of syntax in vector space models of compositional semantics. In *Proceedings of the 51st Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. 894–904.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neur. Comput.* 9, 8 (1997), 1735–1780.

Wenxing Hong, Siting Zheng, and Huan Wang. 2013. Dynamic user profile-based job recommender system. In *Proceedings of the 8th International Conference on Computer Science & Education*. IEEE, 1499–1503.

Baotian Hu, Zhengdong Lu, Hang Li, and Qingcai Chen. 2014. Convolutional neural network architectures for matching natural language sentences. In *Proceedings of the International Conference on Advances in Neural Information Processing Systems*.

Minlie Huang, Qiao Qian, and Xiaoyan Zhu. 2017. Encoding syntactic knowledge in neural networks for sentiment classification. *ACM Trans. Inf. Syst.* 35, 3 (2017), 26.

Po-Sen Huang, Xiaodong He, Jianfeng Gao, Li Deng, Alex Acero, and Larry Heck. 2013. Learning deep structured semantic models for web search using clickthrough data. In *Proceedings of the 22nd ACM International Conference on Information & Knowledge Management*. ACM, 2333–2338.

Wei Huang, Qi Liu, Enhong Chen, Yuying Chen, Zai Huang, Yang Liu, Dan Zhang, and Zhou Zhao. 2019. Hierarchical multi-label text classification: An attention-based recurrent network approach. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*.

Faizan Javed, Phuong Hoang, Thomas Mahoney, and Matt McNair. 2017. Large-scale occupational skills normalization for online recruitment. In *Proceedings of the 31st AAAI Conference on Artificial Intelligence*. 4627–4634.

Yangfeng Ji and Jacob Eisenstein. 2013. Discriminative improvements to distributional sentence similarity. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*. 891–896.

Yongpo Jia, Xuemeng Song, Jingbo Zhou, Li Liu, Liqiang Nie, and David S. Rosenblum. 2016. Fusing social networks with deep learning for volunteerism tendency prediction. In *Proceedings of the 30th AAAI Conference on Artificial Intelligence*.

Yutaka Kabutoya, Tomoharu Iwata, Hisako Shiohara, and Ko Fujimura. 2010. Effective question recommendation based on multiple features for question answering communities. In *Proceedings of the 4th International AAAI Conference on Weblogs and Social Media*.

Nal Kalchbrenner, Edward Grefenstette, and Phil Blunsom. 2014. A convolutional neural network for modelling sentences. In *Proceedings of the 52nd Meeting of the Association for Computational Linguistics*.

Yoon Kim. 2014. Convolutional neural networks for sentence classification. *arXiv preprint arXiv:1408.5882* (2014).

Diederik Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* (2014).

Dorien T. A. M. Kooij, Marianne van Woerkom, Julia Wilkenloh, Luc Dorenbosch, and Jaap J. A. Denissen. 2017. Job crafting towards strengths and interests: The effects of a job crafting intervention on person–job fit and the role of age. *J. Appl. Psych.* 102, 6 (2017), 971.

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. 2012. Imagenet classification with deep convolutional neural networks. In *Proceedings of the International Conference on Advances in Neural Information Processing Systems*. 1097–1105.

Siwei Lai, Liheng Xu, Kang Liu, and Jun Zhao. 2015. Recurrent convolutional neural networks for text classification. In *Proceedings of the 29th AAAI Conference on Artificial Intelligence*.

Guillaume Lample, Miguel Ballesteros, Sandeep Subramanian, Kazuya Kawakami, and Chris Dyer. 2016. Neural architectures for named entity recognition. In *Proceedings of the Human Language Technology: Conference of the North American Chapter of the Association of Computational Linguistics (HLT-NAACL'16)*. 260–270.

Danielle H. Lee and Peter Brusilovsky. 2007. Fighting information overflow with personalized comprehensive information access: A proactive job recommender. In *Proceedings of the 3rd International Conference on Autonomic and Autonomous Systems*. IEEE, 21–21.

Huayu Li, Yong Ge, Hengshu Zhu, Hui Xiong, and Hongke Zhao. 2017. Prospecting the career development of talents: A survival analysis perspective. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 917–925.

Hao Lin, Hengshu Zhu, Yuan Zuo, Chen Zhu, Junjie Wu, and Hui Xiong. 2017. Collaborative company profiling: Insights from an employee's perspective. In *Proceedings of the 31st AAAI Conference on Artificial Intelligence*. 1417–1423.

James MacQueen et al. 1967. Some methods for classification and analysis of multivariate observations. In *Proceedings of the 5th Berkeley Symposium on Mathematical Statistics and Probability*, Vol. 1. 281–297.

Jochen Malinowski, Tobias Keim, Oliver Wendt, and Tim Weitzel. 2006. Matching people and jobs: A bilateral recommendation approach. In *Proceedings of the 39th Hawaii International Conference on System Sciences*, Vol. 6. IEEE, 137c–137c.

Julian McAuley and Jure Leskovec. 2013. Hidden factors and hidden topics: Understanding rating dimensions with review text. In *Proceedings of the 7th ACM Conference on Recommender Systems*. ACM, 165–172.

Prem Melville, Wojciech Gryc, and Richard D. Lawrence. 2009. Sentiment analysis of blogs by combining lexical knowledge with text classification. In *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 1275–1284.

Qingxin Meng, Hengshu Zhu, Keli Xiao, and Hui Xiong. 2018. Intelligent salary benchmarking for talent recruitment: A holistic matrix factorization approach. In *Proceedings of the IEEE International Conference on Data Mining (ICDM'18)*. IEEE, 337–346.

Qingxin Meng, Hengshu Zhu, Keli Xiao, Le Zhang, and Hui Xiong. 2019. A hierarchical career-path-aware neural network for job mobility prediction. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. ACM, 14–24.

Lili Mou, Rui Men, Ge Li, Yan Xu, Lu Zhang, Rui Yan, and Zhi Jin. 2015. Natural language inference by tree-based convolution and heuristic matching. *arXiv preprint arXiv:1512.08422* (2015).

Paul Neculoiu, Maarten Versteegh, and Mihai Rotaru. 2016. Learning text similarity with siamese recurrent networks. In *Proceedings of the 1st Workshop on Representation Learning for NLP*. 148–157.

Rong Pan, Yunhong Zhou, Bin Cao, Nathan N. Liu, Rajan Lukose, Martin Scholz, and Qiang Yang. 2008. One-class collaborative filtering. In *Proceedings of the 8th IEEE International Conference on Data Mining*. IEEE, 502–511.

Liang Pang, Yanyan Lan, Jiafeng Guo, Jun Xu, Shengxian Wan, and Xueqi Cheng. 2016. Text matching as image recognition. In *Proceedings of the 30th AAAI Conference on Artificial Intelligence*.

Hao Peng, Jianxin Li, Yu He, Yaopeng Liu, Mengjiao Bao, Lihong Wang, Yangqiu Song, and Qiang Yang. 2018. Large-scale hierarchical text classification with recursively regularized deep graph-CNN. In *Proceedings of the World Wide Web Conference*.

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 *Proceedings of the 41st International ACM SIGIR Conference on Research & Development in Information Retrieval*. ACM, 25–34.

Tomoki Sekiguchi. 2004. Person-organization fit and person-job fit in employee selection: A review of the literature. *Osaka Keidai Ronshu* 54, 6 (2004), 179–196.

Aliaksei Severyn and Alessandro Moschitti. 2015. Learning to rank short text pairs with convolutional deep neural networks. In *Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval*. ACM, 373–382.

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 International Joint Conferences on Artificial Intelligence Organization*. 3542–3548.

Yelong Shen, Xiaodong He, Jianfeng Gao, Li Deng, and Grégoire Mesnil. 2014. Learning semantic representations using convolutional neural networks for web search. In *Proceedings of the 23rd International Conference on World Wide Web*. ACM, 373–374.

Society for Human Resource Management SHRM. 2016. 2016 Human Capital Benchmarking Report. Retrieved from https://www.shrm.org/hr-today/trends-and-forecasting/research-and-surveys/Documents/2016-Human-Capital-Report.pdf.

Richard Socher, Jeffrey Pennington, Eric H. Huang, Andrew Y. Ng, and Christopher D. Manning. 2011. Semi-supervised recursive autoencoders for predicting sentiment distributions. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, 151–161.

Xuemeng Song, Zhao-Yan Ming, Liqiang Nie, Yi-Liang Zhao, and Tat-Seng Chua. 2016. Volunteerism tendency prediction via harvesting multiple social networks. *ACM Trans. Inf. Syst.* 34, 2 (2016), 10.

Xuemeng Song, Liqiang Nie, Luming Zhang, Mohammad Akbari, and Tat-Seng Chua. 2015. Multiple social network learning and its application in volunteerism tendency prediction. In *Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval*. ACM, 213–222.

Peijie Sun, Le Wu, and Meng Wang. 2018. Attentive recurrent social recommendation. In *Proceedings of the 41st International ACM SIGIR Conference on Research & Development in Information Retrieval*. ACM, 185–194.

Ying Sun, Fuzhen Zhuang, Hengshu Zhu, Xin Song, Qing He, and Hui Xiong. 2019. The impact of person-organization fit on talent management: A structure-aware convolutional neural network approach. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. ACM, 1625–1633.

Kai Sheng Tai, Richard Socher, and Christopher D. Manning. 2015. Improved semantic representations from tree-structured long short-term memory networks. In *Proceedings of the 53rd Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing*.

Yunzhi Tan, Min Zhang, Yiqun Liu, and Shaoping Ma. 2016. Rating-boosted latent topics: Understanding users and items with ratings and reviews. In *Proceedings of the International Joint Conferences on Artificial Intelligence Organization*, Vol. 16. 2640–2646.

Duyu Tang, Bing Qin, and Ting Liu. 2015. Document modeling with gated recurrent neural network for sentiment classification. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*. 1422–1432.

Mingfei Teng, Hengshu Zhu, Chuanren Liu, Chen Zhu, and Hui Xiong. 2019. Exploiting the contagious effect for employee turnover prediction. In *Proceedings of the 33rd AAAI Conference on Artificial Intelligence*.

Toxic Test-doctoring. 2018. Test-doctoring to keep Japanese women out of medical school. Retrieved from https://www.economist.com/asia/2018/08/09/test-doctoring-to-keep-japanese-women-out-of-medical-school.

Aaron Van den Oord, Sander Dieleman, and Benjamin Schrauwen. 2013. Deep content-based music recommendation. In *Proceedings of the International Conference on Advances in Neural Information Processing Systems*. 2643–2651.

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 34th AAAI Conference on Artificial Intelligence*. AAAI Press.

Jian Wang, Yi Zhang, Christian Posse, and Anmol Bhasin. 2013. Is it time for a career switch? In *Proceedings of the 22nd International Conference on World Wide Web*. ACM, 1377–1388.

Meng Wang, Weijie Fu, Shijie Hao, Hengchang Liu, and Xindong Wu. 2017a. Learning on big graph: Label inference and regularization with anchor hierarchy. *IEEE Trans. Knowl. Data Eng.* 29 (2017), 1101–1114.

Meng Wang, Weijie Fu, Shijie Hao, Dacheng Tao, and Xindong Wu. 2016. Scalable semi-supervised learning by efficient anchor graph regularization. *IEEE Trans. Knowl. Data Eng.* 28, 7 (2016), 1864–1877.

Meng Wang, Hao Li, Dacheng Tao, Ke Lu, and Xindong Wu. 2012. Multimodal graph-based reranking for web image search. *IEEE Trans. Image Proc.* 21, 11 (2012), 4649–4661.

Sida Wang and Christopher D. Manning. 2012. Baselines and bigrams: Simple, good sentiment and topic classification. In *Proceedings of the 50th Meeting of the Association for Computational Linguistics*, Vol. 2. 90–94.

Yang Wang, Xuemin Lin, Lin Wu, and Wenjie Zhang. 2017b. Effective multi-query expansions: Collaborative deep networks for robust landmark retrieval. *IEEE Trans. Image Proc.* 26, 3 (2017), 1393–1404.

Lin Wu, Yang Wang, Xue Li, and Junbin Gao. 2018a. Deep attention-based spatially recursive networks for fine-grained visual recognition. *IEEE Trans. Cyber.* 49, 5 (2018), 1791–1802.

Lin Wu, Yang Wang, Xue Li, and Junbin Gao. 2018b. What-and-where to match: Deep spatially multiplicative integration networks for person re-identification. *Pattern Recog.* 76 (2018), 727–738.

Xunxian Wu, Tong Xu, Hengshu Zhu, Le Zhang, Enhong Chen, and Hui Xiong. 2019. Trend-aware tensor factorization for job skill demand analysis. In *Proceedings of the 28th International Joint Conference on Artificial Intelligence*. AAAI Press, 3891–3897.

Yijun Xiao and Kyunghyun Cho. 2016. Efficient character-level document classification by combining convolution and recurrent layers. *ArXiv* abs/1602.00367 (2016).

Chengfeng Xu, Pengpeng Zhao, Yanchi Liu, Jiajie Xu, Victor S. Sheng, Zhiming Cui, Xiaofang Zhou, and Hui Xiong. 2019. Recurrent convolutional neural network for sequential recommendation. In *Proceedings of the World Wide Web Conference.* ACM, 3398–3404.

Huang Xu, Zhiwen Yu, Hui Xiong, Bin Guo, and Hengshu Zhu. 2015. Learning career mobility and human activity patterns for job change analysis. In *Proceedings of the IEEE International Conference on Data Mining.* IEEE, 1057–1062.

Huang Xu, Zhiwen Yu, Jingyuan Yang, Hui Xiong, and Hengshu Zhu. 2016. Talent circle detection in job transition networks. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining.* ACM, 655–664.

Tong Xu, Hengshu Zhu, Chen Zhu, Pan Li, and Hui Xiong. 2018. Measuring the popularity of job skills in recruitment market: A multi-criteria approach. In *Proceedings of the 32nd AAAI Conference on Artificial Intelligence.*

Yiming Yang and Jan O. Pedersen. 1997. A comparative study on feature selection in text categorization. In *Proceedings of the 14th International Conference on Machine Learning.* 412–420.

Zichao Yang, Diyi Yang, Chris Dyer, Xiaodong He, Alexander J. Smola, and Eduard H. Hovy. 2016. Hierarchical attention networks for document classification. In *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies.*

Yuyang Ye, Hengshu Zhu, Tong Xu, Fuzhen Zhuang, Runlong Yu, and Hui Xiong. 2019. Identifying high potential talent: A neural network based dynamic social profiling approach. In *Proceedings of the IEEE International Conference on Data Mining.* IEEE.

Zikai Yin, Tong Xu, Hengshu Zhu, Chen Zhu, Enhong Chen, and Hui Xiong. 2019. Matching of social events and users: A two-way selection perspective. *Springer World Wide Web* (2019). DOI: https://doi.org/10.1007/s11280-019-00724-7

Denghui Zhang, Junming Liu, Hengshu Zhu, Yanchi Liu, Lichen Wang, Pengyang Wang, and Hui Xiong. 2019a. Job2Vec: Job title benchmarking with collective multi-view representation learning. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management.* ACM, 2763–2771.

Le Zhang, Hengshu Zhu, Tong Xu, Chen Zhu, Chuan Qin, Hui Xiong, and Enhong Chen. 2019c. Large-scale talent flow forecast with dynamic latent factor model. In *Proceedings of the World Wide Web Conference.* ACM, 2312–2322.

Shuai Zhang, Lina Yao, Aixin Sun, and Yi Tay. 2019b. Deep learning based recommender system: A survey and new perspectives. *ACM Comput. Surv.* 52, 1 (2019), 5.

Ting Zhang, Bang Liu, Di Niu, Kunfeng Lai, and Yu Xu. 2018. Multiresolution graph attention networks for relevance matching. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management.* ACM, 933–942.

Xiang Zhang, Junbo Jake Zhao, and Yann LeCun. 2015. Character-level convolutional networks for text classification. In *Proceedings of the International Conference on Advances in Neural Information Processing Systems.*

XianXing Zhang, Yitong Zhou, Yiming Ma, Bee-Chung Chen, Liang Zhang, and Deepak Agarwal. 2016. Glmix: Generalized linear mixed models for large-scale response prediction. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining.* ACM, 363–372.

Yingya Zhang, Cheng Yang, and Zhixiang Niu. 2014. A research of job recommendation system based on collaborative filtering. In *Proceedings of the 7th International Symposium on Computational Intelligence and Design*, Vol. 1. IEEE, 533–538.

Lei Zheng, Vahid Noroozi, and Philip S. Yu. 2017. Joint deep modeling of users and items using reviews for recommendation. In *Proceedings of the 10th ACM International Conference on Web Search and Data Mining.* ACM, 425–434.

Chen Zhu, Hengshu Zhu, Hui Xiong, Pengliang Ding, and Fang Xie. 2016. Recruitment market trend analysis with sequential latent variable models. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining.* ACM, 383–392.

Chen Zhu, Hengshu Zhu, Hui Xiong, Chao Ma, Fang Xie, Pengliang Ding, and Pan Li. 2018. Person-job fit: Adapting the right talent for the right job with joint representation learning. *ACM Trans. Manag. Inf. Syst.* 9, 3 (2018), 12.

Received April 2019; revised November 2019; accepted December 2019