Improving Next-Application Prediction with Deep Personalized-Attention Neural Network

Jun Zhu, Gautier Viaud, Céline Hudelot

Improving Next-Application Prediction with Deep Personalized-Attention Neural Network

Jun Zhu, Gautier Viaud, Céline Hudelot
Mathématiques et Informatique pour la Complexité et les Systèmes
CentraleSupélec, Université Paris-Saclay
Gif-sur-Yvette, France
{jun.zhu, gautier.viaud, celine.hudelot}@centralesupelec.fr

***Abstract*—Recently, due to the ubiquity and supremacy of E-recruitment platforms, job recommender systems have been largely studied. In this paper, we tackle the next job application problem, which has many practical applications. In particular, we propose to leverage next-item recommendation approaches to consider better the job seeker's career preference to discover the next relevant job postings (referred to jobs for short) they might apply for. Our proposed model, named Personalized-Attention Next-Application Prediction (PANAP), is composed of three modules. The first module learns job representations from textual content and metadata attributes in an unsupervised way. The second module learns job seeker representations. It includes a personalized-attention mechanism that can adapt the importance of each job in the learned career preference representation to the specific job seeker's profile. The attention mechanism also brings some interpretability to learned representations. Then, the third module models the *Next-Application Prediction* task as a top-K search process based on the similarity of representations. In addition, the geographic location is an essential factor that affects the preferences of job seekers in the recruitment domain. Therefore, we explore the influence of geographic location on the model performance from the perspective of negative sampling strategies. Experiments on the public CareerBuilder12 dataset show the interest in our approach.**

***Index Terms*—Next-Application Prediction, Personalized-Attention, Neural Network, Job Recommendation**

I. Introduction

Recently, the rapid development of E-recruitment has considerably influenced the human resource management field. In particular, due to the explosion of recruitment data (i.e., over 3 million jobs are posted on LinkedIn in the U.S. every month[1]), many works have been proposed to build effective job recommender systems [1]. Like many other domains, Deep Learning (DL) based recommendation models have been extensively studied in the recruitment domain. Nevertheless, while mostly studied, there are still important issues with current approaches, which we summarize below:

- **Issue 1**: The first issue is related to intrinsically dynamic and constantly evolving Job-Person interactions, which can be built from job application records or working histories. Identically, we assume that temporal relations between application records tend to point towards sequential/session-based recommendations[2]. To our knowledge, although various session-based methods have been proposed in others domains, session-based job recommendation is less studied. Moreover, another important aspect of the Job-Person interaction matrix is its sparsity, since most job seekers only apply for a few job positions or work as a specific occupation.
- **Issue 2**: The second issue is related to the personalization of recommendations. Indeed, two job seekers can apply for the same job with different motivations. As a consequence, the important information in a job is specific to the job seeker. Thus, it is essential to weigh the jobs differently in the modeling tasks of job seeker careers. General-purpose approaches are often unable to capture such specific information. In addition, the personal context information of the job seeker is of prime importance in the recruitment domain. For example, the geographic location of a job seeker often has a substantial impact on the application decision and the recommendation result. It raises two opposite goals: (i) it is essential to consider this personal context information when modeling personal preference, (ii) it also requires the recommender system to distinguish this information from more core content, i.e., job content, when making recommendations.
- **Issue 3**: The last issue is due to the evolving nature of the recruitment domain. In fact, in addition, to constantly updated E-recruitment websites (i.e., a large number of job postings are added or removed daily, and hundreds of candidate profiles are created or updated), the domain is still evolving (i.e., new occupations, formations, and skills). All of this intensifies the cold-start problem. Therefore, it is impractical to build a static method that cannot be adjusted quickly as the data constantly changes. We thus need to avoid retraining a deep model for each change. Moreover, from the perspective of content analysis methods, due to the sensitivity and privacy of the data in the recruitment domain, it is not easy to create an unbiased and evolving labeled dataset [2], [3]. Thus it prevents the use of supervised machine learning methods.

[1] https://economicgraph.linkedin.com/resources/linkedin-workforce-report-february-2021

[2] In this paper, we describe the recommendation task to explore the sequential data by a broader term "sequential". Thus "session" and "sequential" are interchangeable.

In this paper, we address the problem of job recommendation under the task of the next job application prediction. We propose a hybrid Personalized-Attention Next-Application Prediction model (PANAP) to answer the previous issues partially. Specifically, for **Issue 1**, we model the next job application problem as a sequential recommendation problem, and then compare our proposed model with different session-based recommendation methods to analyze the dynamics in the recruitment domain. Our model is composed of three independent modules. The first module learns job representations from textual job content and available metadata in an unsupervised way to answer **Issue 3**. The second module contains a personalized-attention mechanism to learn the job seeker representation that can solve the problem of personalized recommendations mentioned in **Issue 2**. The third module is used to predict the next-application that a job seeker will apply for. In order to answer Issue 3, this module is inspired by the Deep Structured Semantic Model (DSSM) [4] with a training loss based on representation similarities. Moreover, based on the nature of the recruitment domain, we propose a specific negative sampling strategy considering the geographic location factor. To summarize our contributions as follows:

- A new session-based model, named PANAP, is used for *Next-Application Prediction* task. It integrates the personalized-attention mechanism to improve the recommendation accuracy.
- An extensive experimental study has been conducted in the CareerBuilder12 dataset, allowing us to investigate the effect of leveraging different metadata types and textual job contents, and the impact of different sampling strategies on the quality of recommendations.

II. Related Work

Job recommender systems based on traditional approaches (i.e., collaborative-, content-based and hybrid methods) have been well-studied [5]. However, these methods have some important limitations in the recruitment domain. For example, the sparsity of Job-Person interaction matrices limits the performance of collaborative-based methods. When the information contained in job contents is insufficient, or the feature-engineering on contents is difficult, content-based approaches may become inaccurate. Furthermore, these methods generally lack the ability to model personal information. Recently, DL has dramatically revolutionized recommendation systems. As in other domains, DL-based methods have been applied in the recruitment domain. These works can be categorized into three classes according to the data they used and the learning paradigm. (i) **Text-based matching models**: When labeled Person-Job matching records are available, the recommendation problem can be seen as a supervised text matching task to match jobs with candidates automatically. Many approaches [2], [3], [6] with different architectures have been proposed in the literature. However, due to the protection of trade secrets and personal privacy, collecting such labeled records is difficult. (ii) **Unsupervised representation learning models**: When matching records are not available, the recommendation task can be seen as the learning of representations of job and person in the same embedding space. The recommendation is then modeled as a top-K search based on these learned embeddings using similarity-based algorithms. For example, [7] uses graphs to learn job title and skills representations. [8] proposes a collective multi-view method to learn job title representations. In these approaches, the different level/view of information is of prime importance. (iii) **Approaches based on career paths**: The methods mentioned above primarily focus on characterizing the person or the job information. Transitions in career paths are less considered. To address this problem, some works are proposed, e.g., NEMO [9] explores action dependencies in career paths through Long Short-Term Memory (LSTM) [10] to predict the next career move. In [11], they use a hierarchical LSTM to predict the next potential employer of a person and how long he/she will stay in the new position. However, predicting the next job application based on application records received relatively little consideration except [12], which recommends the next job posting in a K-nearest neighbor manner. It is one of our baselines.

The next-application prediction problem can be considered as a sequential recommendation problem, which has been extensively studied recently in other domains, such as news or product recommendation [13]–[16]. Recurrent Neural Networks (RNNs) have been widely studied for the sequential recommendation since they have demonstrated their effectiveness in processing sequential data, one representative work is GRU4Rec [17]–[19]. The attention mechanism [20] has shown promising potential in improvements of accuracy and interpretability, like the vanilla attention-based [14], [21] and self-attention based one [22], [23].

Different from the previous state-of-the-art in the recruitment domain, in this paper, we tackle the problem of next-application prediction by leveraging session-based recommendation models. We further study the effectiveness of session-based models in the recruitment domain. Moreover, we integrate the personalized-attention mechanism to model the different informativeness of job postings for different job seekers, and we take into account the "location" factor when making recommendations.

III. Problem Formulation

Let $\mathcal{U}$ be a job seeker set and $\mathcal{J}$ be a job posting (referred to jobs for short) set. A job $j \in \mathcal{J}$ can be represented as $j = (\text{ID}_j, \{w_1^j, \ldots, w_m^j\}, \text{meta}_j)$, where ID_j is the unique identifier of job j. $\{w_1^j, \ldots, w_m^j\}$ is a sequence of m words, representing the textual content of job j. It can be any text information about the job. meta_j are associated metadata attributes (i.e., job city). Similarly, each job seeker $u \in \mathcal{U}$ can be summarized as a tuple $u = (\text{ID}_u, \mathcal{H}_u, \text{meta}_u)$ with an identifier ID_u, a professional profile $\mathcal{H}_u$, and metadata attributes meta_u, like the education background or the geographical information. Specifically, the professional profile $\mathcal{H}_u$ can be represented either as his/her working experience or his/her job application record. In this work, we target at the job application record,

and we denote $\mathcal{H}_u$ as a job sequence (session) ordered by time $\mathcal{H}_u = \{j_1, \ldots, j_n\}$, where each j_i is a specific job in $\mathcal{J}$. With the above notations, the *Next-Application Prediction* problem can be formally defined as follows:

Given a set of jobs $\mathcal{J}$ and a set of job seeker $\mathcal{U}$, for each specific job seeker $u \in \mathcal{U}$ who has a job application sequence $\mathcal{H}_u = \{j_1, \ldots, j_t\}$ at one specific time step t.

Predict the next most likely job j_{t+1} that the job seeker u might apply for, which means maximizing the likelihood $P(j_{t+1} = j^+ | \mathcal{H}_u, \mathcal{J})$ of the actual next-applied job j^+ given the application sequence $\mathcal{H}_u$.

IV. The Proposed Model

An overview of our PANAP model is given in Figure 1. We detail our model by describing these three different modules.

A. Job Content Representation

JCR (Job Content Representation), is responsible for learning a representation of each job $j \in \mathcal{J}$, where $j = (\text{ID}_j, \{w_1^j, \ldots, w_m^j\}, \text{meta}_j)$. Specifically, for job j, a textual representation $\boldsymbol{h}_j \in \mathbb{R}^d$ is learned from its textual content $\{w_1^j, \ldots, w_{m_j}^j\}$ with a text encoder:

$$\boldsymbol{h}_j = \text{text\_encoder}(\{w_1^j, \ldots, w_m^j\}), \tag{1}$$

where the text encoder can be some unsupervised textual embedding approaches (e.g., Word2Vec [24] and Doc2Vec [25]) or pre-trained models (e.g., BERT [26]). We believe that textual representations generated by different text encoders will affect the accuracy of the recommendations. It will a research line in our future work. The metadata attributes meta_j (e.g., city and state) are respectively embedded into vectors (e.g., $\boldsymbol{v}_j^{\text{city}}$ and $\boldsymbol{v}_j^{\text{state}}$) through different trainable embedding matrices (e.g., $\boldsymbol{M}^{\text{city}}$ and $\boldsymbol{M}^{\text{state}}$), which will be jointly learned during the training process. The vector $\boldsymbol{v}_j^{\text{meta}}$ is then obtained by concatenation of all vectors, $\boldsymbol{v}_j^{\text{meta}} = [\boldsymbol{v}_j^{\text{city}} \oplus \boldsymbol{v}_j^{\text{country}} \oplus \ldots]$, where the symbol $\oplus$ represents the concatenation operator. Then the textual job representation $\boldsymbol{h}_j$ and job metadata vector $\boldsymbol{v}_j^{\text{meta}}$ are combined by using a sequence of Fully Connected (FC) layers to produce the *Job Content Vector* $\boldsymbol{v}_j \in \mathbb{R}^{d^{\mathcal{J}}}$:

$$\boldsymbol{v}_j = \text{FCs}\big([\boldsymbol{h}_j \oplus \boldsymbol{v}_j^{\text{meta}}]\big). \tag{2}$$

It is worth mentioning that the textual job representation $\boldsymbol{h}_j$ is trained separately in an unsupervised way or generated by a pre-trained language model, so that a new job can be added easily, thereby alleviating the job cold-start problem.

B. Job Seeker Representation

JSR (Job Seeker Representation), is responsible for learning representations of job seekers. It requires three inputs: the identifier ID_u of job seeker u, his/her historical applied job sequence $\mathcal{H}_u = \{j_1, \ldots, j_n\}$[3] and associated metadata attributes meta_u. In addition, $\mathcal{H}_u$ can be transformed as a sequence of *Job Content Vectors* $\{\boldsymbol{v}_{j_1}, \ldots, \boldsymbol{v}_{j_n}\}$, where each $\boldsymbol{v}_{j_i} \in \mathbb{R}^{d^{\mathcal{J}}}$ is obtained from JCR module. Since the same job may have different informativeness for different job seekers, it contributes differently to characterize career profiles of job seekers. Then, we learn the job seeker representation with a personalized-attention mechanism [27]. We first embed the ID_u of job seeker u into a vector $\boldsymbol{e}_u$ using an identifier embedding matrix $\mathbf{M}^{\mathcal{U}} \in \mathbb{R}^{|\mathcal{U}| \times d^s}$, where d^s denotes the dimension of identifier embedding. Then $\boldsymbol{e}_u$ is passed to a dense layer parameterized with $\boldsymbol{W}^q \in \mathbb{R}^{d^{\mathcal{U}} \times d^q}$ and $\boldsymbol{b}^q \in \mathbb{R}^{d^q}$ to form the preference query vector $\boldsymbol{q}_u = \text{ReLU}(\boldsymbol{W}^q \times \boldsymbol{e}_u + \boldsymbol{b}^q)$, where d^q is the query dimension. The importance score of i-th job for job seeker u is calculated as follows:

$$\alpha_i = \exp\left(\boldsymbol{v}_{j_i}^T \boldsymbol{p}_u\right) / \textstyle\sum_{i'=1}^{n} \exp\left(\boldsymbol{v}_{j_{i'}}^T \boldsymbol{p}_u\right), \tag{3}$$

where $\boldsymbol{p}_u = \tanh\big(\boldsymbol{W}^a \times \boldsymbol{q}_u + \boldsymbol{b}^a\big)$, with projection parameters $\boldsymbol{W}^a \in \mathbb{R}^{d^q \times d^{\mathcal{J}}}$ and $\boldsymbol{b}^a \in \mathbb{R}^{d^{\mathcal{J}}}$. Note that the attention scores might differ for the same job, as the query vector depends on the ID of the job seeker. This allows forming a weighted career preference representation $\boldsymbol{h}_u$ for job seeker u:

$$\boldsymbol{h}_u = \textstyle\sum_{i=1}^{n} \alpha_i \boldsymbol{v}_{j_i}. \tag{4}$$

Similar to JCR module, the metadata vector of job seeker u is represented as $\boldsymbol{v}_u^{\text{meta}} = [\boldsymbol{v}_u^{\text{city}} \oplus \boldsymbol{v}_u^{\text{education}} \oplus \ldots]$ by concatenation. $\boldsymbol{v}_u^{\text{meta}}$ and $\boldsymbol{h}_u$ are combined by using a sequence of FC layers to produce the final job seeker representation:

$$\boldsymbol{v}_u = \text{FCs}\big([\boldsymbol{h}_u \oplus \boldsymbol{v}_u^{\text{meta}}]\big), \tag{5}$$

where $\boldsymbol{v}_u$ has a dimension of $d^{\mathcal{U}}$, which is equal to the dimension of *Job Content Vector* $\boldsymbol{v}_j$ (e.g., $d^{\mathcal{U}} = d^{\mathcal{J}}$).

C. Next-Application Predictor

In classical deep recommendation models, the output is usually a probability vector whose dimension is the number of available items. In our scenario of job recommendation, due to the evolving nature of the recruitment domain (Issue 3), such models are not tractable in practice. Thus this dynamic scenario needs an approach that does not need to retrain the whole network at each change. As a consequence, inspired by the highly dynamic news recommendation scenario, we use the ranking loss [13] to train the predictor. The principle of this loss is to train the predictor to maximize the similarity between user preferences and positive samples, while minimizing similarities with negative samples. The idea of the ranking loss comes from DSSM [4], which is an effective document ranking model and has been leveraged for the recommendation. In our case, it can immediately recommend a newly published job as soon as its representation is learned. More precisely, given the application history $\mathcal{H}_u = \{j_1, \ldots, j_n\}$ of job seeker u, we formulate the above representation generating processes as $\boldsymbol{v}_u = JSR(\text{ID}_u, \mathcal{H}_u, \text{meta}_u; \Theta_1)$, where each $j_i \in \mathcal{H}_u$ can be embedded into a *Job Content Vector* $\boldsymbol{v}_{j_i}$ through $JCR(j_i; \Theta_2)$, Θ_1 and Θ_2 are model parameters. $\boldsymbol{v}_u$ and $\boldsymbol{v}_{j_i}$ are vectors of the same dimension, and we thus can define the relevance score between job seeker u and job j by the cosine similarity of their representations:

$$sim(u, j_i) = cos(\boldsymbol{v}_u, \boldsymbol{v}_{j_i}) = \frac{\boldsymbol{v}_u \cdot \boldsymbol{v}_{j_i}}{\|\boldsymbol{v}_u\| \|\boldsymbol{v}_{j_i}\|} \tag{6}$$

The posterior probability of j being the next-job for u is formulated as follows:

$$P(j|u) = \frac{\exp\big(sim(u, j)\big)}{\sum_{j' \in j^+ \cup \mathcal{J}^-} \exp\big(sim(u, j')\big)}, \tag{7}$$

[3] We omit the superscript of u without loss of clarity.

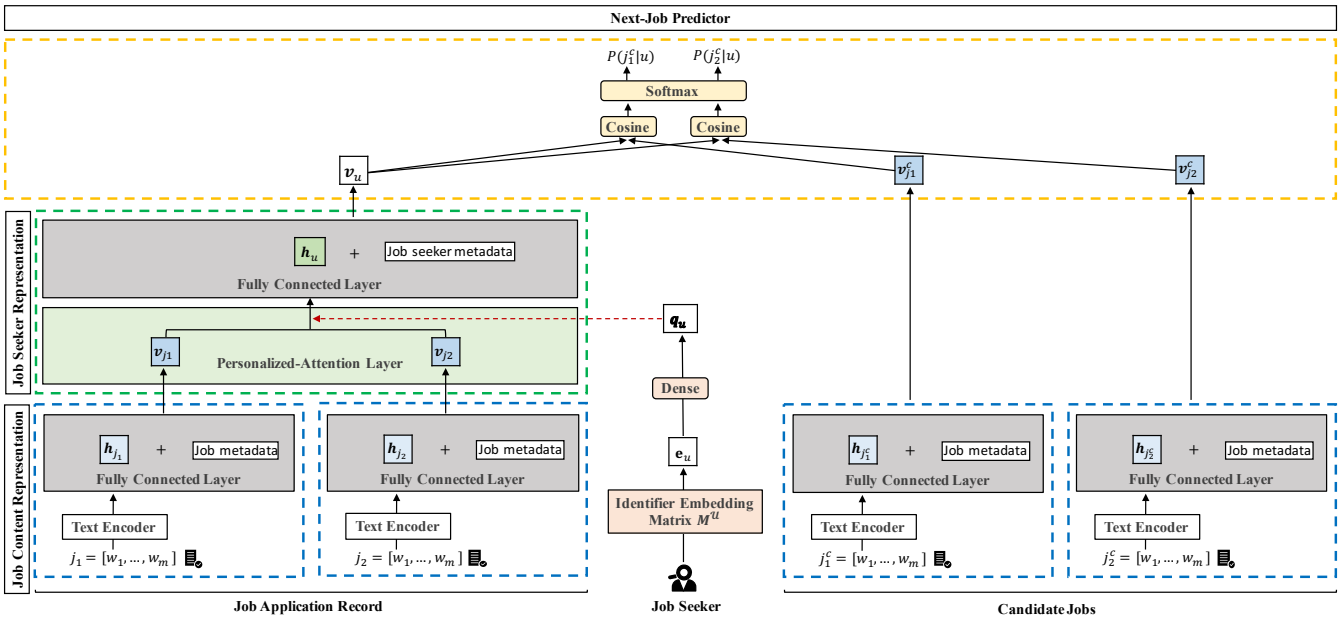


Fig. 1: The proposed PANAP framework consists of three parts: (i) *Job Content Representation* (blue dashed box) is used for job content representation learning. (ii) *Job Seeker Representation* (green dashed box) uses the personalized-attention mechanism to characterize the career preference of job seekers based on the Job Application Record. (iii) *Next-Application Predictor* (yellow dashed box) utilizes a ranking loss based on the representation similarity of job and job seeker to train the model.

where j^+ is the actual next-application, and $\mathcal{J}^-$ represents the negative sample set of jobs which are not applied by the job seeker during his active session. The recommender should learn to maximize the similarity between the content vector $\boldsymbol{v}_{j^+}$ of j^+ and the job seeker preference vector $\boldsymbol{v}_u$, while minimizing similarities with job vector $\boldsymbol{v}_{j^-}$ in the set $\mathcal{J}^-$.

$$\mathcal{L}_{sim} = -\log \prod_{(u,j^+)} P(j^+|\text{ID}_u, \mathcal{H}_u, \text{meta}_u; \Theta_1, \Theta_2). \quad (8)$$

D. Negative Sampling Strategies

Since our method is trained and evaluated with negative samples as described in Section IV-C, the negative sample set $\mathcal{J}^-$ significantly influences the model performance. A well-used sampling strategy is the mini-batch based sampling proposed in [17], which treats the items from the other training/evaluation sessions in the same mini-batch as negative samples. [19] extends the mini-batch based strategy by adding additional samples (based on unity or based on popularity). For some applications, such as news, music, and video, popularity is also an essential factor that influences user choice besides personal preference. Therefore, sampling based on popularity is a good sampling strategy. However, in the recruitment field, unlike these applications, the choices of job seekers are more influenced by their personal contexts, such as the geographic location factor. When job seekers make a choice, they usually first need to consider the job location. They are more likely to apply for jobs in their cities or other cities not far from their current locations (i.e., cities in the same state). We will prove this observation in Section V-A. To solve this problem, our method cooperates with the "location" metadata attributes (e.g., city, state and country) of job and job seeker to model their representations, respectively. As a consequence, these representations contain the "location" information. In addition, considering the "location" when generating negative samples can enable our model to learn useful information for more meaningful recommendations. More Experimental details are provided in Section V-B.

V. Experiments

A. Datasets

We employ *CareerBuilder12*[4] to evaluate our proposed method. Its statistics are given in Table I. In this dataset, job seeker has five metadata: *City*, *State*, *Country*, *Degree*, and *Major*. Job metadata are *City*, *State* and *Country*. The textual job content includes a *Job Title*, a *Job Description* and some *Job Requirements*. From this initial dataset, we created two datasets: (i) *CB12_s* like in [12], in which sessions are created via a time-based split of 30 minutes inactivity threshold, and we discarded sessions with less than two applications for next-job prediction purpose. (ii) *CB12_l* uses all application records during 13 weeks to model the career profile of each job seeker. Thus, *CB12_l* has longer sequences that enable us to evaluate the effectiveness of our proposed method. We further split the last 14 days for testing and the remaining sessions for training. We filter job applications in the test set that do not belong to the training set as this enables a better comparison with the approaches, which can only recommend items that have been used to train the model.

[4] https://www.kaggle.com/c/job-recommendation

TABLE I: Statistics of datasets, $\#S$ represents the session number, and $\#A$ is the applications number. *Avg_SLen* is the average application number in sessions. $\#$*Metadata* contains the cardinality of each metadata attribute.

Dataset	$\lvert\mathcal{U}\rvert$	$\lvert\mathcal{J}\rvert$	#S	#A	Avg_SLen	#**Metadata** City/State/Country/Degree/Major
CB12_s	111,785	207,972	165,027	638,469	3.87	8,226/122/33/7/21,224
CB12_l	137,642	239,581	137,642	772,305	5.61	8,856/130/37/7/25,201

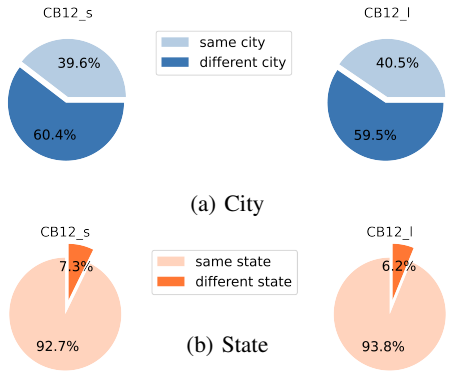


Fig. 2: The relationship between locations of the job seeker and the applied job in *CareerBuilder12* datasets.

As we described in Section IV-D, the "location" is an essential factor that needs to be considered in the job recommendation. In order to further prove this observation, in Figure 2, we illustrate the relationship between locations of job seekers and applied jobs in *CareerBuilder12* datasets. As shown in Figure 2b, most job seekers (92.7% in CB12_s and 93.8% in CB12_l) have applied for jobs in their states. Among these people, 81.3% and 79.7% of job seekers only consider jobs in their own states. Figure 2a shows that only 39.6% and 40.5% of job seekers apply for jobs in their own cities, because job opportunities in their cities are usually limited. People are also applying for jobs in other cities in the same state, as explained above. Therefore, the "location" (the job site or the current location of the job seeker) is an essential factor to be considered in the job recommendation.

B. Experimental Settings

In this experiment, job representations $\boldsymbol{h}_j$ were obtained by Doc2Vec with dimension $d = 300$ via the distributed memory. The dimension d^s of identifier embedding $\boldsymbol{e}_u$ and the query dimension d^q were set to 100. The dimensions $d^{\mathcal{U}}$ and $d^{\mathcal{J}}$ were also set to 300. We applied the dropout technique with a rate of 0.2 to each layer and L2 regularization with rate 1e-4 to parameter weights. The PANAP was trained and evaluated with a 256 mini-batch size, and we used 15 negative samples for training and 50 for evaluation. The Adam [28] optimizer with a learning rate of 5e-4 was used. Metadata attributes with low cardinality ($<= 10$) were one-hot encoded, and high cardinality attributes were represented as trainable embeddings. We used two FC layers, with Leaky ReLU [29] and tanh activation functions to combine metadata vectors.

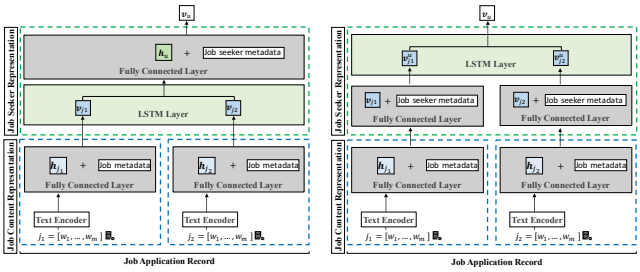
We first explore a sampling strategy inspired by [30], and then propose an improved sampling strategy for the job recommendation scenario. This improved strategy considers the current geographic location of the job seeker and the job site when sampling negative jobs. The two strategies are:

- **Strategy 1 (S1)**-*mini-batch + additional samples*: It is proposed in [30], which adds additional samples with a uniform sampling strategy from a global buffer of the N most recently applied jobs, when there are not enough negative samples within the mini-batch. Jobs are uniformly sampled from the "candidate set" (i.e., jobs within the mini-batch and additional jobs);
- **Strategy 2 (S2)**-*mini-batch + additional samples + location-biased*: Different from Strategy 1, in Strategy 2, we first select the jobs in the same state as the job seeker from the "candidate set" as negative samples. When there are not enough negative samples in the current state, we sample jobs in other states from the "candidate set".

The baselines used are listed as follows:

- *POP*: recommends the most applied job;
- *Association Rule (AR)*: is a simplified version of association rule [31] with a maximum rule size of two;
- *Content Similarity (CS)*: recommends similar jobs based on the cosine similarity between representations of each applied job and the N most recently applied jobs in the global buffer;
- *Item-kNN (IkNN)*: recommends jobs that are similar to the last applied job during the current session as in [17];
- *Session-kNN (SkNN)* [32]: compares the entire current session with the past sessions in the training dataset, rather than considering only the last job;
- *Vector Multiplication SkNN (V-SkNN)* [33]: is a variant of *SkNN* that emphasizes jobs more recently interacted within the current session, when computing the similarities with past sessions (a linear decay function is used);
- *VAE_Comb*: is the best model proposed in [12];
- *GRU4Rec*: is the most recent version of *GRU4Rec* [19];
- *Bert4Rec* [22]: introduces the bi-directional self-attention model to model user behavior sequences;
- Two variants of PANAP: (i) *LSTM* replaces the personalized-attention layer with a LSTM layer, Figure 3a, and (ii) *PLSTM* adds job seeker metadata to each job representation to generate *Personalized job embedding* [13] before the LSTM layer, Figure 3b.

The evaluation metrics used in this work are Hit Rate (HR@5) [33], Mean Reciprocal Rank (MRR@5) [33] and Normalized Discounted Cumulative Gain (NDCG@5) [34].



(a) LSTM (b) PLSTM

Fig. 3: Two variants of PANAP.

VI. Results

In this section, we present our experimental results (HR@5/MRR@5/NDCG@5). For all tables shown in this section, the score in bold is the best in each metric, and the score in the underline is the second best.

A. Next-Application Prediction

TABLE II: *Next-Application Prediction* performance.

Method	CB12_s	CB12_l
POP	0.089/0.042/0.054	0.093/0.046/0.058
AR	0.273/0.200/0.218	0.202/0.146/0.160
CS	0.226/0.128/0.152	0.214/0.118/0.142
IkNN	0.274/0.202/0.220	0.202/0.147/0.161
SkNN	0.349/0.247/0.272	0.276/0.196/0.216
V-SkNN	0.349/0.248/0.273	0.276/0.196/0.217
VAE_Comb	0.345/0.239/0.256	0.282/0.203/0.224
GRU4Rec	0.367/0.208/0.230	0.403/0.309/0.352
BERT4Rec	0.373/0.232/0.245	0.438/0.311/0.369
PANAP (S2)	**0.691/0.492/0.541**	**0.742/0.543/0.593**
LSTM	0.540/0.339/0.389	0.569/0.368/0.418
PLSTM	0.480/0.292/0.338	0.516/0.323/0.371

According to Table II, we have the following observations: (i) Among all models, *POP* and *CS* give the lowerest scores, as they neither model the personalized preference nor consider the sequential information. Although *POP* is often a strong baseline in certain domains, i.e., news and movies, career preferences are less affected by popularity factors in the recruitment domain. (ii) Overall, the NN-based methods consistently outperform traditional methods, demonstrating that NNs are good at modeling sequential information, and the self-attention mechanism can improve accuracy. (iii) Our proposed method *PANAP (S2)* with sampling strategy S2 and its variants *LSTM* and *PLSTM* perform best among all baselines, which indicates the job content and metadata can effectively improve the recommendation performance, similar observations can be found in Section VI-C. (iv) *PANAP (S2)* outperforms *LSTM* and *PLSTM*, which use LSTM to model the sequential information. One possible reason is that in the recruitment domain, career preferences are less dynamic than other domains, and application sequences are relative short (3.87 and 5.61 on average for both sets), thus the advantage of RNN can not be well demonstrated. This result also demonstrates the advantage of personalized-attention as different jobs might have different importance for career preference modeling, and selecting the more critical jobs is useful for achieving better recommendation performance. Moreover, *LSTM* is better than *PLSTM*, one possible reason is that *PLSTM* merges the job seeker metadata into each job in the session, which will weaken the information carried by the job itself.

B. Effectiveness of Personalized Attention

In this part, we analyze the effectiveness of the personalized-attention mechanism using CB12_s dataset. As shown in Table III, the models with attention mechanism consistently outperform the model without attention, and our model with the personalized-attention outperforms its variant with vanilla attention (similar to the global attention used in [11]). Such a result is probably since the vanilla attention uses a fixed query vector and cannot adjust to different personal preferences.

TABLE III: Different attention mechanisms.

Attention_mechanism	HR@5/MRR@5/NDCG@5
Personalized-attention	**0.691/0.492/0.541**
Vanilla attention	0.685/0.480/0.531
No attention (avg)	0.565/0.367/0.416
LSTM	0.540/0.339/0.389

C. Effectiveness of Different Features

As shown in Table IV, (i) It is obvious that methods with the additional information (i.e., content representation or metadata) generally give better results than what is achievable from the job identifier ID_j only (*Only_JobID*) on CB12_s dataset. (ii) The metadata has a fewer influence on *PANAP (S2)* than *LSTM* (29.1% average reduction between *Meta+Content+JobID* and *No_Meta* on three metrics compared to 41.0%). One possible reason is that *PANAP (S2)* utilizes a personalized-attention mechanism to model career preferences, which already contain personal information. (iii) Since the "location" factor affects the choice of job seekers, and our sampling strategy is location-biased. The job metadata, e.g., *City*, *State* and *Country* could give more relevant information about "location", so the scores of *No_JobMeta* are lower than that of *No_SeekerMeta*. (iv) *PANAP (S2)* consistently outperforms *LSTM*, even only the job identifier ID_j is used, which indicates that *PANAP (S2)* does capture personal preference. This observation also proves the advantage of the personalized-attention mechanism in cases where no additional information is available.

D. Negative Sampling Analysis

We examine the performance of two sampling strategies described in Section V-B on CB12_s dataset. According to Table V, *PANAP (S1)* outperforms *PANAP (S2)*.

To explain these results, we visualize job seeker representations (session representations) generated from *PANAP (S1)* and *PANAP (S2)* in Figure 4. We use t-SNE [35] to reduce representation dimensions. For illustration purposes, we categorize each job seeker according to his/her *Major*.

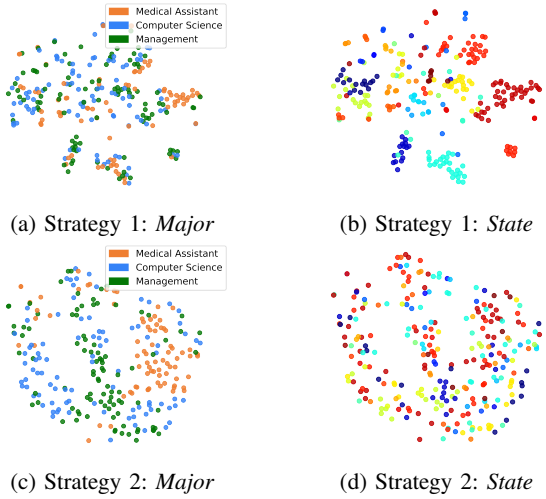
TABLE IV: Different feature combinations. *No_Meta* means that neither the job seeker metadata nor the job metadata is considered.

Feature	PANAP (S2)	LSTM
Meta+Content+JobID	**0.691/0.492/0.541**	0.540/0.339/0.389
No_Meta	0.516/0.334/0.379	0.341/0.189/0.226
No_Content	0.530/0.341/0.386	0.449/0.286/0.326
Only_JobID	0.501/0.312/0.355	0.331/0.182/0.218
No_JobMeta	0.511/0.338/0.381	0.377/0.212/0.253
No_SeekerMeta	0.601/0.399/0.449	0.505/0.297/0.349

TABLE V: Different negative sampling strategies.

Sampling_strategy	HR@5/MRR@5/NDCG@5
PANAP (S1)	**0.756/0.620/0.680**
PANAP (S2)	0.691/0.492/0.541

Note that categorizing job seekers through *Major* is not the most reasonable way because some job seekers are currently engaged in occupations that do not match their majors. Thus, we select three non-similar majors. People with these professional backgrounds are more likely to engage in related jobs, including *Management*, *Computer Science* and *Medical Assistant*. We also plot job seeker representations labeled with *State* in Figure 4b and Figure 4d to show the influences of different sampling strategies. Each color corresponds to one *State*. We observe from Figure 4b that representations learned by our model with S1 (i.e., *PANAP (S1)*) are well-clustered into groups, each corresponds to a *State*. This can be used to explain why the accuracy scores of *PANAP (S1)* are better than that of *PANAP (S2)*. A reasonable explanation is that S1 does not consider the "location" factor when generating negative samples. More specifically, as mentioned in Section IV-D, job seekers are more likely to apply for jobs located in their cities or other cities in the same state. The two main reasons why a job seeker does not apply for a job are: the job content is inappropriate, or the work location is not suitable. If most of the negative samples are jobs in other states, regardless of the job content, these negative samples tend to force the model to capture subtle information between different states rather than different job contents. For instance, a job seeker in *Seattle-Washington* has a background of *computer science*. *sales representative* and *java developer* are two negative samples from *Miami-Florida*. Due to the "location" factor, this job seeker may not apply for *java developer*, while *sales representative* is not suitable from both "location" and job content perspectives. With these negative samples, the model learns to distinguish between locations (e.g., *Seattle-Washington* and *Miami-Florida*) rather than the contents of jobs (e.g., *computer science* and *sales representative*). Therefore, the learned job seeker representations are not categorized according to their *Major* categories, as shown in Figure 4a. Instead, they are grouped together by the "location" as in Figure 4b. To handle this problem, we propose a location-biased sampling strategy, S2, as described in Section V-B, which prioritizes jobs in the same state as the job seeker as negative samples. We visualize the representations learned through S2 in Figures 4c and 4d to show the promising results of our proposed strategy. We observe that the job seekers are grouped by *Major* in Figure 4c, which demonstrates the effectiveness of S2.



(a) Strategy 1: *Major* (b) Strategy 1: *State*

(c) Strategy 2: *Major* (d) Strategy 2: *State*

Fig. 4: Visualization of learned representations.

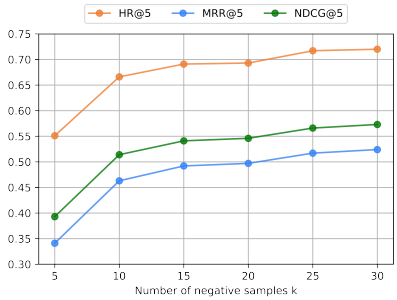


Fig. 5: Different numbers of training negative samples k.

E. Number of Negative Samples

This section discusses the influence of the number of negative samples used for training on prediction accuracy. The experimental results on CB12_s dataset are shown in Figure 5, where k negative samples are used for training and 50 samples for evaluation. When k increases (e.g., from 5 to 10), the performance of our model first has a noticeable improvement. This may be because when k is too small, the information provided by negative samples is relatively limited, and the model cannot learn useful information. Then, when k continues to increase (e.g., from 10 to 20), the performance becomes stable. However, if k is too large, there may not be enough jobs in the same state. Negative samples in other states become dominant, making it difficult for the model to identify the valuable information correctly. As explained in Section VI-D, the model will learn to distinguish the difference of positions, just like *PANAP (S1)*. As a result, the performance consistently improves.

VII. Conclusions

In this work, we proposed a personalized-attention model for the *Next-Application Prediction* problem, which improves

the prediction accuracy. The experiments confirm that, by incorporating personalized-attention, our method can better capture the personal career preference than baseline methods. We investigated the importance of incorporating job content information and metadata in the recruitment domain. Moreover, considering the "location" factor when generating negative samples can provide more useful information on job content.

ACKNOWLEDGMENT

This work is supported by Randstad corporate research chair in collaboration with Université Paris-Saclay, CentraleSupélec, MICS. We would like to thank the *Mésocentre* computing center of CentraleSupélec and École Normale Supérieure Paris-Saclay [5] for providing computing resources.

REFERENCES

[1] P. Tripathi, R. Agarwal, and T. Vashishtha, "Review of job recommender system using big data analytics," in *2016 3rd International Conference on Computing for Sustainable Global Development (INDIACom)*. IEEE, 2016, pp. 3773–3777.

[2] C. Qin *et al.*, "Enhancing person-job fit for talent recruitment: An ability-aware neural network approach," in *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval*, 2018, pp. 25–34.

[3] C. Zhu *et al.*, "Person-job fit: Adapting the right talent for the right job with joint representation learning," *ACM Transactions on Management Information Systems (TMIS)*, vol. 9, no. 3, pp. 1–17, 2018.

[4] P.-S. Huang *et al.*, "Learning deep structured semantic models for web search using clickthrough data," in *Proceedings of the 22nd ACM international conference on Information & Knowledge Management*, 2013, pp. 2333–2338.

[5] S. T. Al-Otaibi and M. Ykhlef, "A survey of job recommender systems," *International Journal of Physical Sciences*, vol. 7, no. 29, pp. 5127–5142, 2012.

[6] S. Bian *et al.*, "Domain adaptation for person-job fit with transferable deep global match network," in *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, 2019, pp. 4812–4822.

[7] V. S. Dave, B. Zhang, M. Al Hasan, K. AlJadda, and M. Korayem, "A combined representation learning approach for better job and skill recommendation," in *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*, 2018, pp. 1997–2005.

[8] D. Zhang *et al.*, "Job2vec: Job title benchmarking with collective multi-view representation learning," in *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*, 2019, pp. 2763–2771.

[9] L. Li *et al.*, "Nemo: Next career move prediction with contextual embedding," in *Proceedings of the 26th International Conference on World Wide Web Companion*, 2017, pp. 505–513.

[10] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[11] Q. Meng, H. Zhu, K. Xiao, L. Zhang, and H. Xiong, "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*, 2019, pp. 14–24.

[12] E. Lacic *et al.*, "Using autoencoders for session-based job recommendations," *User Modeling and User-Adapted Interaction*, vol. 30, no. 4, pp. 617–658, 2020.

[13] G. de Souza Pereira Moreira, F. Ferreira, and A. M. da Cunha, "News session-based recommendations using deep neural networks," in *Proceedings of the 3rd Workshop on Deep Learning for Recommender Systems*, 2018, pp. 15–23.

[14] J. Li *et al.*, "Neural attentive session-based recommendation," in *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*, 2017, pp. 1419–1428.

[15] F. Yu, Q. Liu, S. Wu, L. Wang, and T. Tan, "A dynamic recurrent model for next basket recommendation," in *Proceedings of the 39th International ACM SIGIR conference on Research and Development in Information Retrieval*, 2016, pp. 729–732.

[16] H. Fang, D. Zhang, Y. Shu, and G. Guo, "Deep learning for sequential recommendation: Algorithms, influential factors, and evaluations," *ACM Transactions on Information Systems (TOIS)*, vol. 39, no. 1, pp. 1–42, 2020.

[17] B. Hidasi, A. Karatzoglou, L. Baltrunas, and D. Tikk, "Session-based recommendations with recurrent neural networks," *arXiv preprint arXiv:1511.06939*, 2015.

[18] B. Hidasi, M. Quadrana, A. Karatzoglou, and D. Tikk, "Parallel recurrent neural network architectures for feature-rich session-based recommendations," in *Proceedings of the 10th ACM conference on recommender systems*, 2016, pp. 241–248.

[19] B. Hidasi and A. Karatzoglou, "Recurrent neural networks with top-k gains for session-based recommendations," in *Proceedings of the 27th ACM international conference on information and knowledge management*, 2018, pp. 843–852.

[20] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," *arXiv preprint arXiv:1409.0473*, 2014.

[21] Q. Liu, Y. Zeng, R. Mokhosi, and H. Zhang, "Stamp: short-term attention/memory priority model for session-based recommendation," in *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2018, pp. 1831–1839.

[22] F. Sun *et al.*, "Bert4rec: Sequential recommendation with bidirectional encoder representations from transformer," in *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*, 2019, pp. 1441–1450.

[23] W.-C. Kang and J. McAuley, "Self-attentive sequential recommendation," in *2018 IEEE International Conference on Data Mining (ICDM)*. IEEE, 2018, pp. 197–206.

[24] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," *arXiv preprint arXiv:1301.3781*, 2013.

[25] Q. Le and T. Mikolov, "Distributed representations of sentences and documents," in *International conference on machine learning*. PMLR, 2014, pp. 1188–1196.

[26] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," *arXiv preprint arXiv:1810.04805*, 2018.

[27] C. Wu *et al.*, "Npa: neural news recommendation with personalized attention," in *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2019, pp. 2576–2584.

[28] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[29] A. L. Maas, A. Y. Hannun, and A. Y. Ng, "Rectifier nonlinearities improve neural network acoustic models," in *Proc. icml*, vol. 30, no. 1. Citeseer, 2013, p. 3.

[30] P. M. Gabriel De Souza, D. Jannach, and A. M. Da Cunha, "Contextual hybrid session-based news recommendation with recurrent neural networks," *IEEE Access*, vol. 7, pp. 169 185–169 203, 2019.

[31] R. Agrawal, T. Imieliński, and A. Swami, "Mining association rules between sets of items in large databases," in *Proceedings of the 1993 ACM SIGMOD international conference on Management of data*, 1993, pp. 207–216.

[32] D. Jannach and M. Ludewig, "When recurrent neural networks meet the neighborhood for session-based recommendation," in *Proceedings of the Eleventh ACM Conference on Recommender Systems*, 2017, pp. 306–310.

[33] M. Ludewig and D. Jannach, "Evaluation of session-based recommendation algorithms," *User Modeling and User-Adapted Interaction*, vol. 28, no. 4-5, pp. 331–390, 2018.

[34] H. Fang, G. Guo, D. Zhang, and Y. Shu, "Deep learning-based sequential recommender systems: Concepts, algorithms, and evaluations," in *International Conference on Web Engineering*. Springer, 2019, pp. 574–577.

[35] L. Van der Maaten and G. Hinton, "Visualizing data using t-sne." *Journal of machine learning research*, vol. 9, no. 11, 2008.

[5] http://mesocentre.centralesupelec.fr/