Large-Scale Talent Flow Forecast with Dynamic Latent Factor Model*

Le Zhang[1,2], Chen Zhu[2], Hengshu Zhu[2], Tong Xu[1], Enhong Chen[1], Chuan Qin[1], Hui Xiong[1,2,3,4]
[1]Anhui Province Key Lab of Big Data Analysis and Application, University of Science and Technology of China
[2]Baidu Talent Intelligence Center, Baidu Inc, [3]Business Intelligence Lab, Baidu Research
[4]National Engineering Laboratory of Deep Learning Technology an Application, China
laughing@mail.ustc.edu.cn, {zc3930155, chuanqin0426, xionghui}@gmail.com
zhuhengshu@baidu.com, {tongxu, cheneh}@ustc.edu.cn

ABSTRACT

The understanding of talent flow is critical for sharpening company talent strategy to keep competitiveness in the current fast-evolving environment. Existing studies on talent flow analysis generally rely on subjective surveys. However, without large-scale quantitative studies, there are limits to deliver fine-grained predictive business insights for better talent management. To this end, in this paper, we aim to introduce a big data-driven approach for predictive talent flow analysis. Specifically, we first construct a time-aware job transition tensor by mining the large-scale job transition records of digital resumes from online professional networks (OPNs), where each entry refers to a fine-grained talent flow rate of a specific job position between two companies. Then, we design a dynamic latent factor based Evolving Tensor Factorization (ETF) model for predicting the future talent flows. In particular, a novel evolving feature by jointly considering the influence of previous talent flows and global market is introduced for modeling the evolving nature of each company. Furthermore, to improve the predictive performance, we also integrate several representative attributes of companies as side information for regulating the model inference. Finally, we conduct extensive experiments on large-scale real-world data for evaluating the model performances. The experimental results clearly validate the effectiveness of our approach compared with state-of-the-art baselines in terms of talent flow forecast. Meanwhile, the results also reveal some interesting findings on the regularity of talent flows, e.g. *Facebook becomes more and more attractive for the engineers from Google in 2016.*

CCS CONCEPTS

• **Information systems → Data mining**.

KEYWORDS

Talent Flow Forecast, Latent Factor Model, Tensor Factorization

ACM Reference Format:
Le Zhang[1,2], Chen Zhu[2], Hengshu Zhu[2], Tong Xu[1], Enhong Chen[1], Chuan Qin[1], Hui Xiong[1,2,3,4]. 2019. Large-Scale Talent Flow Forecast with Dynamic Latent Factor Model. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3308558.3313525

*Hui Xiong, Hengshu Zhu and Tong Xu are corresponding authors.

WWW '19, May 13–17, 2019, San Francisco, CA, USA

ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313525

1 INTRODUCTION

In the fast-evolving business environments, to ensure the competitive advantages, it becomes critical for companies to timely and consistently review their talent strategies. Therefore, talent flow analysis, the process of analyzing the phenomenon of job hopping among different regions or companies, always acts as a major component in market intelligence of modern companies. Indeed, with the help of talent flow analysis, many business insights and managerial initiatives can be achieved. For example, at the macro level, based on the talent flow among different countries, governments can monitor the brain drain, and carry out macroeconomic regulations for gaining and sustaining the global competitive advantage. At the micro level, companies can timely evaluate their employer brand, uncover the major competitors in the talent market, and conduct proactive talent management.

However, most of existing studies on talent flow analysis generally rely on subjective surveys, with a focus on qualitative causal inference from psychological, economic and cultural perspectives [2, 4]. Due to the lack of large-scale market data for quantitative analysis, there are limits for delivering fine-grained and predictive business insights through talent flow analysis [10, 20, 31]. Recently, the prevalence of online professional networks (OPNs) enables the accumulation of a large number of digital resumes, which contains rich information about the career path of talents. For example, as of the end of 2017, there are over 400 million resumes worldwide available at Linkedin. Indeed, these large-scale talent data provide unprecedented opportunities for conducting talent flow analysis.

To this end, in this paper, we aim to introduce a data-driven approach for fine-grained predictive talent flow analysis. Specifically, we first construct a time-aware job transition tensor by mining the large-scale job transition records of digital resumes, where each entry refers to a fine-grained talent flow rate of a specific job position between two companies, i.e., a tuple <`Origin Company`, `Job Position`, `Destination Company`, `Time Slice`>. Then, we design a dynamic latent factor based Evolving Tensor Factorization (ETF) model for predicting the future talent flows, where the origin company and the destination company are represented as the time-dependent vectors, the job position is represented as a time-independent low-dimension vector, and the time is used as the evolving signal. In particular, to model the evolving process of talent flows, we develop a job transition network based on the talent flows, where companies represent nodes, and the aggregated

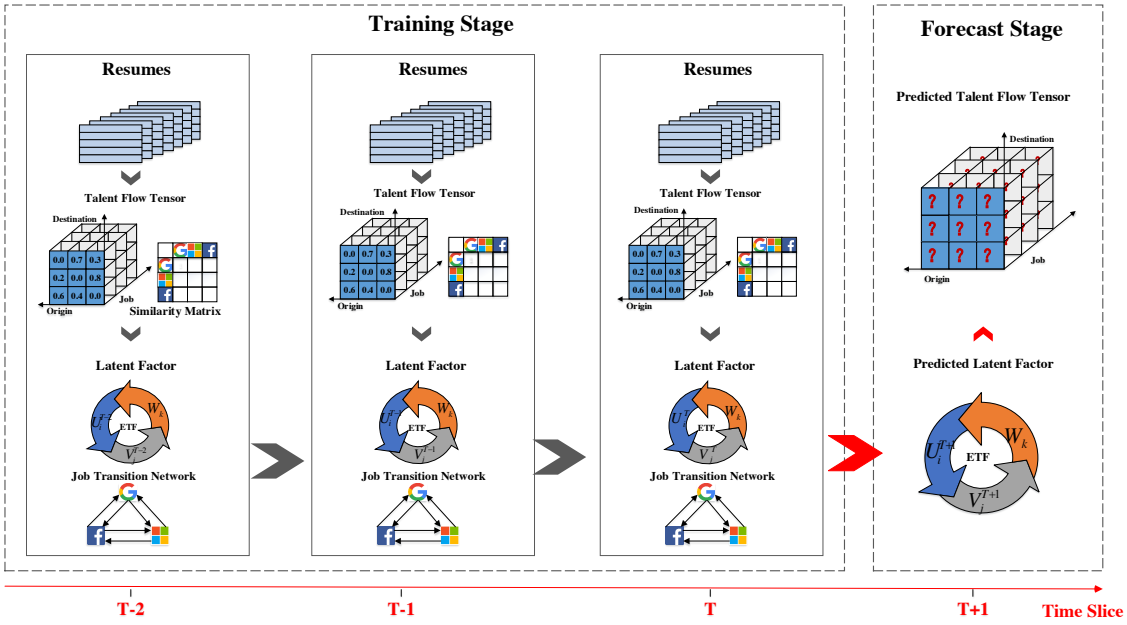


Figure 1: The diagrammatic sketch of our approach for talent flow forecast, where the dynamic latent factors are learned from the historical talent flows, and are used for forecasting the future talent flows.

job transition between two companies represents the weight of edge. With this network, a novel evolving feature by jointly considering the influence of previous talent flows and global market is introduced for modeling the evolving principles of each company. Furthermore, to improve the predictive performance of ETF, we also integrate several representative attributes of companies as side information for regulating the model inference. Figure 1 illustrates the diagrammatic sketch of our approach. Finally, the extensive experiments conducted on a real-world dataset clearly validate the effectiveness of our approach compared with other state-of-the-art baselines in terms of talent flow forecast.

Specifically, the major contributions of this paper can be summarized as follows:

- We introduce a big data-driven approach for predictive talent flow analysis based on the large-scale online resume data. Our approach allows to identify fine-grained and predictive business insights.
- We propose a novel dynamic latent factor based evolving model for talent flow forecast, which consists of a novel evolving feature for jointly modeling the influence of previous talent flows and global market, and a company similarity based regularizer for enhancing model inference.
- We conduct extensive experiments on large-scale real-world data for evaluating the model performance. The experimental results not only validate the performance of our approach for talent flow forecast, but also reveal some interesting findings on the regularity of talent flows, such as *Facebook becomes more and more attractive for the engineers from Google.*

Overview. The rest of the paper is organized as follows. In Section 2, we describe the real-world data and formulate the problem. Section 3 shows the technical details of our dynamic latent factor model for talent flow forecast. Then, we introduce the experimental results in Section 4. After that, we summarize the important related works in Section 5. Finally, Section 6 concludes the work.

2 PRELIMINARIES

In this section, we first introduce the real-world dataset in our study, and then formulate the problem of fine-grained talent flow forecast.

2.1 Data Description

In this paper, we use a representative real-world dataset from one of the largest commercial OPNs, which contains more than 220 million professional resumes, to study the problem of talent flow forecast. Specifically, each resume contains a list of job experience records, where each record consists of the company name, job title with brief job description, and the working duration in a monthly granularity. An example is shown in Figure 2, more details of the dataset could be found in Section 4.

To extract the job transition records from the resumes, here we extend the methods introduced in [24] by allowing job overlaps since it is reasonable for an employee to start a new job a few months before the official resignation of her former job. Formally, a resume R contains a list of job experience records $[R_{c1}, R_{c2}, ..., R_{cn}]$, and each record R_{ci} contains a start date R^s_{ci} and an end date R^e_{ci}. Therefore, as long as the absolute difference between the end time of the former job and the start time of the successor job less than a predefined threshold θ, the transition from the former job to the successor job will be regarded as effective. Note that, in our settings, a former job may has multiple successor jobs. Furthermore, we formulate the effective job transition as a father-son relationship, and

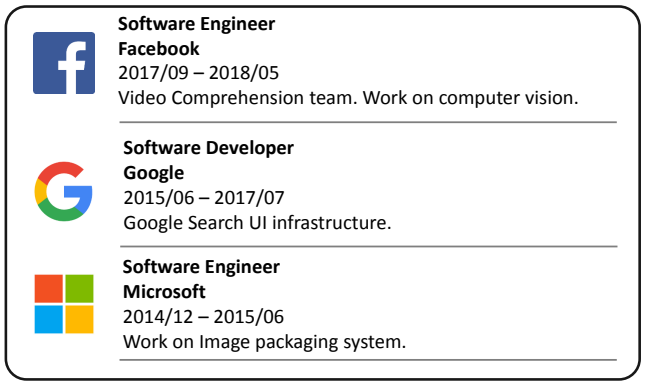


Figure 2: An example of the resume in our dataset.

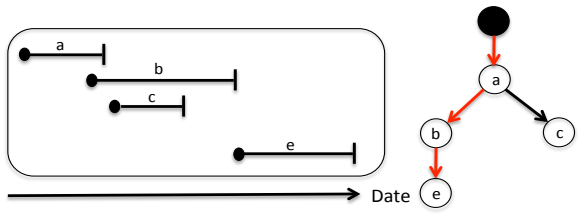


Figure 3: An example of transforming work experiences into job transition records.

construct a career tree where each company represents a node. Intuitively, there may exist multiple paths from the root node to the leaf node, the length of which is set as the sum of corresponding work time. In particular, the longest path will be selected to represent the major career path of the employee, and the adjacent company node on the major path represents a job transition.

Specifically, we use the example shown in Figure 3 to illustrate how to extract job transition records from a resume. In the figure, we use line segments to represent work experiences and use letters to represent the corresponding company names. By checking the working durations of all the work experiences, we can get several job transitions records, including $a \rightarrow b$, $a \rightarrow c$ and $b \rightarrow c$. The corresponding career tree is shown in the right part of Figure 3. Obviously, in this career tree there exist two paths, i.e., $\{a \rightarrow b \rightarrow e\}$ and $\{a \rightarrow c\}$. And because the first path is longer than the other one, we believe it is the major career path of this individual and just keep the job transitions in it, i.e., $[a \rightarrow b, b \rightarrow c]$. The job transition date is set as the start date of the successor and the job title is determined by the title of the predecessor.

2.2 Problem Formulation

Based on the extracted job transition records, here we introduce the problem formulation of talent flow forecast. For facilitating illustration, Table 1 lists some important mathematical notations used throughout this paper.

Formally, we suppose there are N companies and M job positions. Based on the job transition records, we can construct talent flow tensor $\mathcal{R}^t \in \mathbb{R}^{N \times N \times M}$ for each time slice t, where each element

Table 1: Mathematical notations.

Symbol	Description
$\mathcal{R}^t$	The talent flow tensor at time slice t;
U_i^t	The latent vector of origin company i at time slice t;
V_j^t	The latent vector of destination company j at time slice t;
W_k	The latent vector of job position k;
C	The set of companies;
S	The attribute similarity matrix of companies;
N	The number of companies;
M	The number of job position;
T	The number of time windows;
D	The dimension of latent vector.

$\mathcal{R}^t_{ijk}$ is defined as the normalized number of corresponding job transitions, i.e,

$$\mathcal{R}^t_{ijk} = \frac{Num^t_{i,j,k}}{\sum_{j=1}^{N} Num^t_{i,j,k}}, \tag{1}$$

where $Num^t_{i,j,k}$ denotes the aggregated job transitions from the job position k of company i to company j at time slice t. Along this line, the problem of talent flow forecast can be formulated as:

*Definition 2.1 (**Talent Flow Forecast**).* Given a set of talent flow tensors $\{\mathcal{R}^1, ..., \mathcal{R}^T\}$, which is constructed by the transition records as of time T, and the corresponding attributes of companies (e.g., industry, scale, and locations), the goal of talent flow forecast is to predict the value of $\mathcal{R}^{(T+1)}_{ijk}$.

3 TECHNICAL DETAILS

In this section, we first introduce a latent factor based model, named Basic Latent Factor based Evolving Tensor Factorization model (Basic-ETF), for talent flow forecast. Furthermore, we extend it by adding an attribute similarity based constraint for improving its performance.

3.1 Basic-ETF Model

Intuitively, given a talent flow at time slice t, i.e., $\mathcal{R}^t_{ijk}$, its value is influenced by three factors, namely talent outflow company i, talent inflow company j, and job position k. Here we represent them by latent vectors as follows:

- U_i^t represents the latent vector of the origin company i at time slice t.
- V_j^t represents the latent vector of the destination company j at time slice t.
- W_k^t represents the latent vector of the job position k at time slice t.

Without loss of generality, we assume the estimation of R^t_{ijk} equals to $f(U_i^t, V_j^t, W_k^t)$. Accordingly, the conditional distribution over the observed talent flow at time slice t is defined as

$$p(\mathcal{R}|U, V, W, \sigma) = \prod_{i=1}^{N}\prod_{j=1}^{N}\prod_{k=1}^{M}[\mathcal{N}(\mathcal{R}^t_{ijk}|f(U_i^t, V_j^t, W_k^t), \sigma^2)]^{I^t_{ijk}}, \tag{2}$$

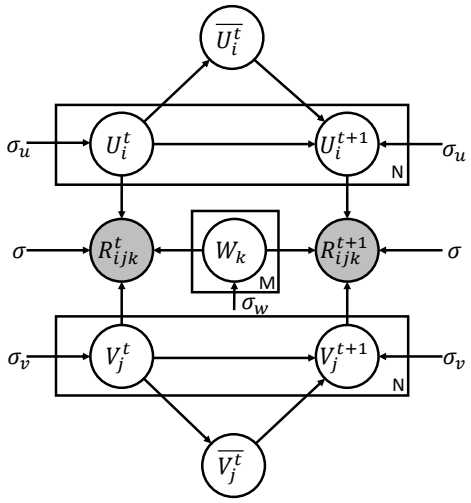


Figure 4: The graphical representation of Basic-ETF.

where $\mathcal{N}(x|\mu, \sigma^2)$ is the probability density function of the Gaussian distribution with mean μ and variance σ^2, and I_{ijk}^t is the indicator function that equals to 1 if talent flow $\mathcal{R}_{ijk}^t > 0$ and equals to 0 otherwise. Specifically, we use interactions to model the relationships between talent flow R_{ijk}^t and latent vectors U_i^t, V_j^t, W_k^t. In other words, we have

$$f(U_i^t, V_j^t, W_k^t) = U_i^t V_j^t + U_i^t W_k^t + V_j^t W_k^t. \tag{3}$$

Next, we will introduce our assumptions about the time-aware relationships among U_i^t, V_j^t, and W_k^t.

As for W_k^t, in this paper, each job position represents a kind of jobs in all of time slices which play the similar function. Thus, we think it is generally time-independent and use W_k to denote the vector of job position k at all time slices. Here, we add zero-mean Gaussian priors on W, which is similar to the traditional matrix factorization [16]. Then, we have:

$$p(W|\sigma_w) = \prod_{k=1}^{M} p(W_k|\sigma_w) = \prod_{k=1}^{M} \mathcal{N}(W_k|0, \sigma_w^2 I), \tag{4}$$

where I is the D-by-D identity matrix.

As for the latent vectors of companies U_i^t and V_j^t, we think the talent preference of a company would evolve over time. Specifically, we assume there are two factors that influence companies' features in future. The first one is the **previous influence**. Intuitively, the features of company at time slice t are related to its features in the previous time slice $t-1$. In other words, the U_i^t (V_j^t) is related to U_i^{t-1}(V_j^{t-1}). The other one is the **market influence**, which means the company may evolve according to the general market tendency. Specifically, we assume those companies which have similar business construct the market together. And two companies are considered to have similar business when employees are very likely to move among them [3]. Therefore, here we define the business similarity of two companies based on the total number of talent transition between them. In particular, we construct a job transition network, where a node represents a company, a weighted edge represents the number of the corresponding job transitions. Then, for the target company i, at time slice t, we choose the neighbor companies that exist edges with the target company in the job transition network to represent its market, denoted by $\mathcal{M}_i^t$. And the impact of a neighbor node m to company i is denoted by P_{im}^t. In this paper, the P_{im}^t is set as follows:

$$P_{im}^t = \frac{E_{im}^t}{\sum_{m \in \mathcal{M}_i} E_{im}^t}, \tag{5}$$

where E_{im}^t denotes the total number of job transitions between company i and m. Now, we represent the market influence on the feature of origin company i by $\left(\sum_{m \in \mathcal{M}_i^{t-1}} P_{im}^{t-1} U_m^{t-1}\right)$ at time slice t, and for the destination company j by $\left(\sum_{m \in \mathcal{M}_j^{t-1}} P_{jm}^{t-1} V_m^{t-1}\right)$.

Intuitively, the companies may have their own decisions when balancing the two aspects. For example, some large companies prefer to follow their own historical state, while the small companies would like to follow the market. So we define two parameters for each company to balance the two aspects, namely α_i and β_j. Finally, the evolving process is modeled as:

$$\begin{aligned}
&p(U_i^t) = \mathcal{N}(U_i^t|\bar{U}_i^{t-1}, \sigma_u^2 I), \quad p(V_j^t) = \mathcal{N}(V_j^t|\bar{V}_j^{t-1}, \sigma_v^2 I),\\
&where \quad \bar{U}_i^{t-1} = (1-\alpha_i) U_i^{t-1} + \alpha_i \sum_{m \in \mathcal{M}_i^{t-1}} P_{im}^{t-1} U_m^{t-1},\\
&\qquad\qquad \bar{V}_j^{t-1} = (1-\beta_j) V_j^{t-1} + \beta_j \sum_{m \in \mathcal{M}_j^{t-1}} P_{jm}^{t-1} V_m^{t-1},\\
&s.t.\ \forall i, j \in C,\ 0 \le \alpha_i \le 1,\ 0 \le \beta_j \le 1,
\end{aligned} \tag{6}$$

where C denotes the company set. At the initial time $t = 1$, neither previous influence nor market influence is available. So we just assume zero-mean Gaussian distribution for the latent vectors of companies. To be specific,

$$p(U^1) = \prod_{i=1}^{N} \mathcal{N}(U_i^1|0, \sigma_u^2 I), \quad p(V^1) = \prod_{j=1}^{N} \mathcal{N}(V_j^1|0, \sigma_v^2 I). \tag{7}$$

Then, we summarize the prior over companies' latent feature matrix sequence as

$$\begin{aligned}
p(U|\bar{U}, \sigma_u) &= \prod_{i=1}^{N} \mathcal{N}(U_i^1|0, \sigma_u^2 I) \prod_{t=2}^{T} \prod_{i=1}^{N} \mathcal{N}(U_i^t|\bar{U}_i^{t-1}, \sigma_u^2 I),\\
p(V|\bar{V}, \sigma_v) &= \prod_{j=1}^{N} \mathcal{N}(V_i^1|0, \sigma_v^2 I) \prod_{t=2}^{T} \prod_{j=1}^{N} \mathcal{N}(V_j^t|\bar{V}_j^{t-1}, \sigma_v^2 I).
\end{aligned} \tag{8}$$

According to the graphical model shown in Figure 4, we can get the posterior distribution over U, V, W easily. To be specific,

$$\begin{aligned}
&p(U, V, W|R, \sigma, \sigma_u, \sigma_v, \sigma_w) \propto\\
&p(U|\bar{U}, \sigma_u) p(V|\bar{V}, \sigma_v) p(W|\sigma_w) p(R|U, V, W, \sigma).
\end{aligned} \tag{9}$$

We aim to find the value of U_i^t, V_j^t, W_k by maximizing the log-posterior of Equation 9, which is equivalent to minimizing the

sum-of-squared-errors loss function:

$$\begin{aligned}\mathcal{L}_b &= \frac{1}{2}\sum_{t=1}^{T}\sum_{i=1}^{N}\sum_{j=1}^{N}\sum_{k=1}^{M} I_{ijk}^t\left(\mathcal{R}_{ijk}^t - f(U_i^t, V_j^t, W_k)\right)^2 \\ &+ \frac{\lambda_u}{2}\sum_{t=2}^{T}\sum_{i=1}^{N}\left\|U_i^t - \bar{U}_i^{t-1}\right\|_F^2 + \frac{\lambda_v}{2}\sum_{t=2}^{T}\sum_{j=1}^{N}\left\|V_j^t - \bar{V}_j^{t-1}\right\|_F^2 \\ &+ \frac{\lambda_w}{2}\sum_{k=1}^{M}\|W_k\|_F^2 + \frac{\lambda_u}{2}\sum_{i=1}^{N}\left\|U_i^1\right\|_F^2 + \frac{\lambda_v}{2}\sum_{j=1}^{N}\left\|V_j^1\right\|_F^2, \\ s.t.\ \forall i, j \in C, 0 \le \alpha_i \le 1, 0 \le \beta_j \le 1,\end{aligned} \tag{10}$$

where $\lambda_u = \sigma^2/\sigma_u^2$, $\lambda_v = \sigma^2/\sigma_v^2$, $\lambda_w = \sigma^2/\sigma_w^2$ and $\|\cdot\|_F^2$ denotes the Frobenius norm.

3.2 Company Attribute Constraint

The basic model only considers the numerical job transition information, here we integrate some extra attribute information of companies into the basic model to further refine the performance of talent flow forecast, and the complete model is called ETF.

Specifically, first we extract some representative attributes for each company, which contain company's industry, scale, location, specialties, type and age. Next, these attributes of a company are integrated as a vector by using one hot encoding. To estimate the attribute similarity among companies, we choose the Cosine similarity of the corresponding vectors. Finally, we use a constraint to model the similarity between companies. We formulate the **Company Similarity Regularizer** as

$$\begin{aligned}\mathcal{L}c &= \sum_{t=1}^{T}\left(\sum_{i=1}^{N}\sum_{i'=1}^{N} S(i,i')\left\|U_i^t - U_{i'}^t\right\|_F^2 + \sum_{j=1}^{N}\sum_{j'=1}^{N} S(j,j')\left\|V_j^t - V_{j'}^t\right\|_F^2\right) \\ &= \sum_{t=1}^{T}\sum_{i=1}^{N}\sum_{i'=1}^{N}\sum_{d=1}^{D} S(i,i')\left(U_i^t(d)^2 - U_i^t(d)U_{i'}^t(d)\right) \\ &+ \sum_{t=1}^{T}\sum_{j=1}^{N}\sum_{j'=1}^{N}\sum_{d=1}^{D} S(j,j')\left(V_j^t(d)^2 - V_j^t(d)V_{j'}^t(d)\right) \\ &= \sum_{t=1}^{T}\sum_{d=1}^{D}\left(U^t(d)^\top(\bar{S}-S)U^t(d) + V^t(d)^\top(\bar{S}-S)V^t(d)\right) \\ &= \sum_{t=1}^{T}\left(tr\left((U^t)^\top(\bar{S}-S)U^t\right) + tr\left((V^t)^\top(\bar{S}-S)V^t\right)\right),\end{aligned} \tag{11}$$

where $tr(\cdot)$ represents the matrix trace, and $S(i,i')$ is the attribute similarity between the origin company i and i'. $\bar{S}$ is the degree matrix of S, which is defined as

$$\bar{S} = \begin{cases}\sum_{i=1}^{N} S(i,i'), & i = i', \\ 0, & \text{otherwise.}\end{cases} \tag{12}$$

Here, we use the company attribute similarity matrices S to regularize the learning process of company latent matrix U^t and V^t, which guarantees the components of U^t (or V^t) should be similar if their corresponding company attributes are similar. By integrating the extra attribute information of companies into the basic model, we can obtain the final optimization objective of our model. To be specific, we have

$$min\mathcal{L} = min(\mathcal{L}_b + \lambda_c\mathcal{L}_c), \tag{13}$$

where λ_c denotes the parameter of company similarity constraint.

Algorithm 1 Parameter Learning of the ETF

Require: $\mathcal{R} = \{\mathcal{R}^1, \mathcal{R}^2, ..., \mathcal{R}^T\}$, attribute matrix F, the learning rate η, the hypeparemeters λ_u, λ_v, λ_w, λ_c
Ensure: U, V, W, α and β
Initialize U, V, W, α and β
Calculate the attribute similarity matrix S base on F
Calculate the corresponding degree matrix $\bar{S}$ of S
while Not Converged **do**
  for each (i, j, k, t) in the $\mathcal{R}$ **do**
    Update latent vector W_k with Equation 15
    Update latent vector U_i^t with Equation 16
    Update latent vector V_j^t with Equation 17
    Update bias vector α_i and β_j according to Equation 18
  end for
end while

3.3 Model Learning and Forecasting

The coupling between U, V, W and the balance parameters make the above loss function $\mathcal{L}$ not-convex. However, a local minimum could be achieved by performing gradient descent on each parameter iteratively. Here, we introduce how to use the gradient descent approach to learn our model. The partial derivative of the balancing coefficient α_i and β_j are:

$$\begin{aligned}\frac{\partial\mathcal{L}}{\partial\alpha_i} &= \lambda_u\sum_{t=2}^{T}\sum_{i=1}^{N}(U_i^t - \bar{U}_i^{t-1})\left(\sum_{m\in\mathcal{M}_i^{t-1}} P_{im}^{t-1}U_m^t - U_i^{t-1}\right), \\ \frac{\partial\mathcal{L}}{\partial\beta_j} &= \lambda_v\sum_{t=2}^{T}\sum_{j=1}^{N}(V_j^t - \bar{V}_j^{t-1})\left(\sum_{m\in\mathcal{M}_j^{t-1}} P_{jm}^{t-1}V_m^t - V_j^{t-1}\right).\end{aligned} \tag{14}$$

The derivatives of job position vector W_k, the origin company's latent vector U_i^t, and destination company's vector V_j^t are

$$\frac{\partial\mathcal{L}}{\partial W_k} = \sum_{i=1}^{N}\sum_{j=1}^{N} I_{ijk}^t\left(\mathcal{R}_{ijk}^t - f(U_i^t, V_j^t, W_k)\right)(U_i^t + V_j^t) + \lambda_w W_k, \tag{15}$$

$$\begin{aligned}\frac{\partial\mathcal{L}}{\partial U_i^t} &= \sum_{j=1}^{N}\sum_{k=1}^{M} I_{ijk}^t\left(\mathcal{R}_{ijk}^t - f(U_i^t, V_j^t, W_k)\right)(V_j^t + W_k) \\ &+ \mathcal{I}[t=1]\cdot\lambda_u U_i^1 + I[t\ge 2]\cdot\lambda_u(U_i^t - \bar{U}_i^t) \\ &+ \lambda_u(1-\alpha_i)(\bar{U}_i^{t+1} - U_i^{t+1}) + \lambda_c(\bar{S}-S)U_i^t \\ &+ \lambda_u\sum_{i\in\mathcal{M}_m^t}\alpha_m\cdot P_{mi}^t(\bar{U}_m^{t+1} - U_m^{t+1}),\end{aligned} \tag{16}$$

$$\begin{aligned}\frac{\partial \mathcal{L}}{\partial V_j^t} = &\sum_{i=1}^{N}\sum_{k=1}^{M} I_{ijk}^t \left(\mathcal{R}_{ijk}^t - f(U_i^t, V_j^t, W_k)\right)(U_i^t + W_k) \\ &+ \mathcal{I}[t=1]\cdot\lambda_v V_j^1 + I[t\geq 2]\cdot\lambda_v(V_i^t - \bar{V}_i^t) \\ &+ \lambda_v(1-\beta_j)(\bar{V}_j^{t+1} - V_j^{t+1}) + \lambda_c(\bar{S} - S)V_j^t \\ &+ \lambda_v \sum_{i\in\mathcal{M}_m^t} \beta_m \cdot P_{mj}^t(\bar{V}_m^{t+1} - V_m^{t+1}),\end{aligned} \tag{17}$$

where $\mathcal{I}[x]$ is an indicator function that equals 1 if x is true and 0 otherwise. Because there is no constraint on U, V, and W at the updating step, we can directly update them by the Stochastic Gradient Descent (SGD) [1] method. As for α_i and β_j, they must be bounded between 0 and 1. So we employee the Projected Gradient (PG) [15] to achieve it. Specifically, by denoting the learning rate as η, the PG method updates α_i and β_j in $l+1$-iteration by the following rules:

$$\begin{aligned}\alpha_i^{l+1} &\leftarrow \mathcal{P}\left(\alpha_i^l + \eta\lambda_u(U_i^t - \bar{U}_i^t)(\sum_{m\in\mathcal{M}_i^{t-1}} P_{im}^{t-1}U_m^t - U_i^{t-1})\right), \\ \beta_j^{l+1} &\leftarrow \mathcal{P}\left(\beta_j^l + \eta\lambda_v(V_j^t - \bar{V}_j^t)(\sum_{m\in\mathcal{M}_j^{t-1}} P_{jm}^{t-1}V_m^t - V_j^{t-1})\right),\end{aligned} \tag{18}$$

where the function $\mathcal{P}(\cdot)$ is defined as follows:

$$\mathcal{P}(x) = \begin{cases} 0, & \text{if } x \leq 0, \\ \alpha_i, & \text{if } 0 \leq x \leq 1, \\ 1, & \text{if } x \geq 1. \end{cases} \tag{19}$$

In conclusion, Algorithm 1 describes the optimization process of the ETF model. After learning the related parameters U, V, W, α, and β, we can forecast the $\mathcal{R}_{ijk}^{T+1}$ by:

$$\begin{aligned}U_i^{T+1} &\approx (1-\alpha_i)U_i^T + \alpha_i \sum_{m\in\mathcal{M}_i^T} P_{im}^T U_m^T, \\ V_j^{T+1} &\approx (1-\beta_j)V_j^T + \beta_j \sum_{m\in\mathcal{M}_j^T} P_{jm}^T V_m^T, \\ \mathcal{R}_{ijk}^{T+1} &\approx f(U_i^{T+1}, V_j^{T+1}, W_k).\end{aligned} \tag{20}$$

4 EXPERIMENTS

In this section, we introduce the experimental results for validating the effectiveness of our approach.

4.1 Data Pre-processing

There are three main steps of data pre-processing, namely *data filtering*, *job title categorization* and *Data Spliting*.

Table 2: Statistics of our dataset.

The number of resumes	2,176,157
The number of companies	4,567
The number of transition records	5,875,626

Data Filtering. In our real-world dataset, there are 670,260 unique companies. To avoid noise, we removed the companies that appeared in less than 1,000 resumes, and removed the resumes that contained only one company in the experience records (i.e., without job-hopping). After that, we finally retained 4,567 target companies for experiments. Meanwhile, as shown in Figure 5, the frequency of job transition has a clear cycle of 12 months. Therefore, in the experiments, we set the length of time window as one year. Moreover, we also find that the frequency of job transition grows steadily before January 2016, while the frequency dropped significantly after then. It is because that, in most cases, the users in OPNs do not update their profiles immediately after changing the job. Therefore, the closer the date of job transition is to the time of data collection (e.g., January 2017 for our dataset), the more incomplete data is obtained. To guarantee the effectiveness of evaluation, we only selected the job transition records between January 2010 and January 2016 for experiments. The detailed statistics of our pre-processed dataset are shown in Table 2. In addition, Figure 6 shows the distribution of the number of companies with respect to different average number of out/in-flow of talents each year. From the results, we can find that both distributions are unbalanced, and most of the companies only have limited rate of talent flows, while only few companies have high rate of talent flows.

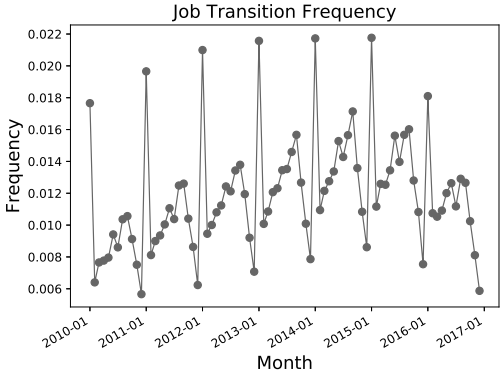


Figure 5: The normalized frequency of job transitions over different months.

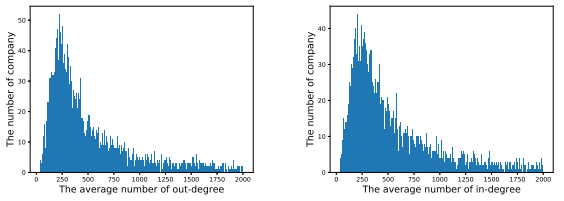
Job Title Categorization. In our dataset, there are total 448,309 unique job titles, which were filled in by users without specification. In our experiments, to normalize the job titles, we used an online API called MonkeyLearn [1] to categorize the original job titles into 26 classes according to the job descriptions written in the resumes. The detailed results of job title categorization in our dataset is shown in Table 3.

Data Spliting. After the above data-processing, we further standardized the date of job transitions into different time slices with year-based granularity. In the evaluation, we split four experimental datasets according to the time slice, the statistics of which are shown in Table 4. Specifically, we used the data as of time slice T for model training in the data splitting process, e.g., T=5 (T=3) in DATA10-15 (DATA12-15), and the data at the last time slice for test, e.g., T=6 (T=4) in DATA10-15 (DATA12-15). Finally, to initialize

[1] https://app.monkeylearn.com/main/classifiers

Table 3: Job title category and distribution

Category	# Title	# Transition	Category	# Title	# Transition
Accounting	50,543	565,151	Arts-and-design	25,931	310,469
Administrative	26,465	316,872	Business-development	44,158	488,703
Consulting	51,399	495,402	Community-and-social-services	10,902	128,140
Engineering	13,133	157,250	Education	43,518	491,046
Entrepreneurship	15,726	158,285	Finance	14,649	175,400
Healthcare-services	1,284	95,383	Human Resources	4,290	81,368
Information-technology	3,230	58,675	Legal	10,834	129,715
Military-and-protective-services	1,626	79,463	Marketing	6,646	79,576
Media-and-communication	8,819	105,592	Operations	8,815	105,545
Program-and-project-management	26,940	322,550	Purchasing	11,940	142,961
Product-management	27,127	336,772	Quality-assurance	9,348	111,934
Research	2,566	60,726	Real-estate	10,974	131,394
Sales	1,836	50,014	Support	15,803	189,215



(a) Average Number of Out-degree (b) Average Number of In-degree

Figure 6: The distributions of the number of companies w.r.t the average number of in/out-flows of talents each year.

Table 4: The Statistics of Experimental Dataset

DataSet	Density	# Training	# Test
DATA10-15	9.5787 ‰	3,116,686	514,582
DATA11-15	9.2922 ‰	2,519,555	514,582
DATA12-15	8.6874 ‰	1,884,447	514,582
DATA13-15	7.6677 ‰	1,247,452	514,582

the talent flow tensor, we aggregated all transition records by the origin company, destination company, job category and time slice simultaneously.

4.2 Evaluation on Talent Flow Forecast

In this subsection, we introduce the evaluation of our approach in talent flow forecast.

Baselines. To evaluate the performance of our model in talent flow forecast, we selected a number of state-of-art methods as baselines. Specifically, we first chose two popular tensor factorization based approaches, and several latent factor based evolutionary approaches. Then, to further validate the performance of each component in our model, we also designed some simplified variants of our model.

- **CP (CANDECOMP/PARAFAC)** [11], which decomposes the tensor into same rank of latent factors, i.e., an n-dimensional tensor is factorized into a sum of outer products of n vectors.
- **Tucker** [5], which considers the tensor factorization problem as the high order PCA, i.e., an n-dimensional tensor is factorized into a small core tensor and n matrices.
- **BPTF** [22], which extends PMF [16] to tensor factorization for modeling the temporal relational data by creating an additional latent feature vector for each time slice.
- **CKF** [19], which models the temporal dynamics by allowing users' factors to evolve through the linear state space model.
- **BTMF** [28], which extends the conventional matrix factorization by assuming the user preferences change over time, and models the dynamics by learning a transition matrix for user latent vectors between consecutive time windows.
- **Pure-ETF**, which is a simple version of ETF, by ignoring the influence of neighbor when modeling the evolution process and the attribute information of company. It equals to minimizing the loss function defined in Equation 10 with $\alpha_i = \beta_j = 0$, for each $i, j \in C$.
- **Basic-ETF**, which is a simplified version of ETF, by considering the evolutionary process with loss function defined in Equation 10.

Evaluation Metrics. In our experiments, we chose two metrics for evaluating the performance of talent forecast, namely Root Mean Squared Error (RMSE) and Mean Absolute Error (MAE). Specifically, the two metrics are defined as

$$RMSE(y, \hat{y}) = \sqrt{\frac{1}{n}\sum_{t=1}^{n}(y_i - \hat{y}_i)^2}, \tag{21}$$

$$MAE(y, \hat{y}) = \frac{1}{n}\sum_{t=1}^{n}|y_i - \hat{y}_i|, \tag{22}$$

where y_i is the actual value, while $\hat{y}_i$ is the estimated value, and n is the number of test instances. Generally speaking, the best prediction has a value of 0 of both RMSE and MAE. And the lower values of these metrics are, the better forecast is obtained. Both metrics were selected since they are widely adopted in evaluating

Table 5: The RMSE performance of each model.

Model	DATA10-15	DATA11-15	DATA12-15	DATA13-15
CP	0.251807	0.251775	0.251941	0.252031
Tucker	0.251866	0.251922	0.252118	0.252291
BPTF	0.191331	0.195110	0.208030	0.232327
CKF	0.180299	0.185846	0.194905	0.209273
BTMF	0.180384	0.187388	0.197022	0.211710
Pure-ETF	0.159148	0.161018	0.163138	0.163533
Basic-ETF	0.146928	0.148899	0.150921	0.156031
ETF	**0.145037**	**0.146495**	**0.150856**	**0.148303**

Table 6: The MAE performance of each model.

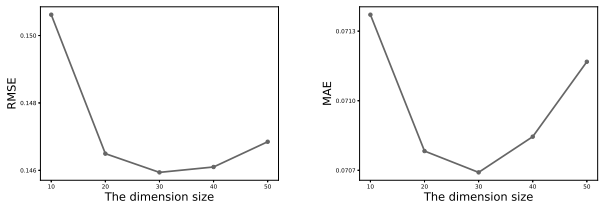
Model	DATA10-15	DATA11-15	DATA12-15	DATA13-15
CP	0.131997	0.131939	0.132112	0.132262
Tucker	0.132012	0.132075	0.132318	0.132572
BPTF	0.092110	0.098019	0.104780	0.118702
CKF	0.086739	0.090773	0.096204	0.104698
BTMF	0.087472	0.091340	0.096623	0.104959
Pure-ETF	0.079467	0.081352	0.083581	0.084650
Basic-ETF	0.072298	0.073935	0.077193	0.086700
ETF	**0.067981**	**0.069183**	**0.071247**	**0.073079**

forecasting models [9]. Meanwhile, the RMSE is a part of our loss function (i.e., defined in Equation 10), so that it is suitable to validate whether our model is over-fitting in the training stage.

Evaluation Setup. In our experiments, each method was evaluated at four datasets, as shown in Table 4. Specifically, 10% records of the training data were selected randomly as validation data for parameter tuning. The rest part of training data were used to train models. And the final performance was evaluated on the test set.

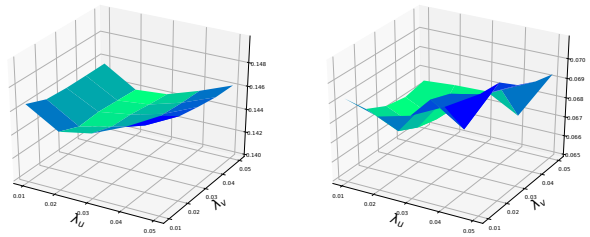
For the tensor factorization based approaches to forecast, such as CP and Tucker, we trained these models on the training tensors directly. After getting the representation of each time slice in the training data, the average representation of each time slice, such as $1, 2, \ldots, T$, was used to denote the representation of $T+1$ [7]. Finally, we could predict the value at time $T+1$ based on the specific decomposition strategy of each method. For the evolution based approaches, such as BPTF, CKF and BTMF, which only consider two dimensions (e.g., user- and item-dimension) at a specific time, we changed the form of our dataset. Specifically, we merged the `Original Company` and `Job Position` dimensions of our tensor at time t to construct an extended matrix. Then in our experiments, we treated the job position of a original company as "user" and treated the destination company as the "item", for executing each method. In particular, all the parameters of models were tuned in the validation set to ensure the best performance.

Overall Performance. The overall RMSE and MAE results of different approaches are shown in Table 5 and 6 respectively. Obviously, our ETF method achieves the better performance than all the baselines in all cases. And from the results, we can have the following observations. First, the tensor factorization based models always get the worst performance, which is because they treat time as a separate element and ignore the evolutionary relationship of latent factors. Second, the Pure-ETF model, which is the simplified version of ETF achieves the better performance than the latent factor based evolutionary models. i.e., BPTF, CKF and BTMF. This means considering the interaction of three factors is more effective



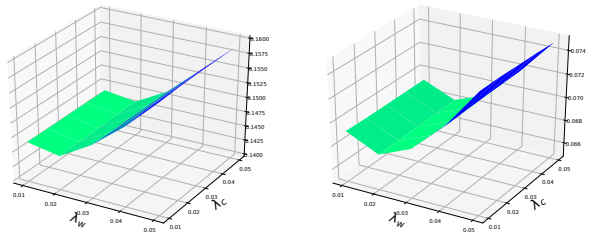
(a) RMSE (b) MAE

Figure 7: The evaluation on the impact of the dimension size.



(a) RMSE (b) MAE

Figure 8: The evaluation on the impact of the λ_u and λ_v.



(a) RMSE (b) MAE

Figure 9: The evaluation on the impact of the λ_w and λ_c.

than only considering two factors. Third, Basic-ETF outperforms Pure-ETF, which means that considering the market influence in evolution process is effective. Meanwhile, we find the attribute similarity based constraints can improve the model by 1% at least in RMSE metric and 5% at least in MAE. Finally, the prediction performance of methods which model the evolutionary process is related to the length of time slices. The longer the length of time slices is, the better performance is obtained. It may because that the longer time slices can help to capture the regular pattern of evolution. Meanwhile the performance of ETF drops slowly than all baselines as the time slices decrease, which validates the robustness of our model.

4.3 Evaluation on Parameter Sensitivity

Here, we turn to discuss the sensitivity of parameters in our approach. Specifically, several parameters will be discussed, namely the dimension of latent factor D, the coefficient of original company vector λ_u, the coefficient of destination company vector λ_v, the coefficient of job vector λ_w and the coefficient of company similarity

(a) Ground Truth-2015 (b) Ground Truth-2016. (c) Forecast-2016.

Figure 10: (a)-(b) The ground truth of the most attractive companies to engineers of Google in 2015 and 2016. (c) The forecast of the most attractive companies to engineers of Google in 2016 by our model.

constraints λ_c. All the experiments were conducted on DATA11-15, and we executed the model for three times and used the average of result to represent the final performance.

First, we discuss the sensitiveness of the dimension of latent factor D, which is summarized in Figure 7. By setting D from 10 to 50, we find that the generally the increasing D will improve the performance, which could be reasonable as more dimensions may probably keep more useful information. However, the performance of the model drops a little, which may because that the more parameters cause the model over-fitting at the training stage. Second, we discuss the sensitiveness of the evolving parameter λ_u and λ_v, which is shown in Figure 8. By setting parameters from 0.01 to 0.05, we find that the model gets the best performance when $\lambda_u = 0.02$ and $\lambda_v = 0.04$ on both metrics. Finally, we set λ_w and λ_c from 0.01 to 0.05. As the result shown in Figure 9, we find that the model gets the best performance when $\lambda_w = 0.02$ and $\lambda_c = 0.02$.

4.4 Case Study

Here we introduce a case study of talent flow forecast for further validating our approach. Specifically, we trained the ETF model based on the job transition records from January 2010 to December 2015, and forecasted the out-flow for engineers (i.e., employees with engineering related jobs) of Google in 2016 by Equation 20. The ground truth results in 2015/2016 and the forecast results in 2016 are all shown in Figure 10, where the size of words are proportional to the number of talent flows (i.e., more attractive). Based on the results, we can obtain two findings. First, comparing with the ground truth in 2016, the forecast results have very similar distributions, which clearly validates the performance of our approach. Second, comparing the ground truth in 2015 and 2016, we can find that the distributions of companies have obvious difference, which indicates the dynamic characteristics of talent flows. For example, Facebook becomes more and more attractive for the engineers from Google. Indeed, our approach can clearly capture such change of talent flows, which validates the effectiveness of using dynamic latent factors for modeling the evolving of talent flows.

5 RELATED WORK

In this section, we will summarize the related works in the following three categories, namely the talent flow analysis, time-dependent collaboration filtering and tensor factorization.

Talent Flow Analysis. As a crucial issue of human resource management, talent management has attracted wide attention in the past decade. On the one hand, the studies of talent flow analysis, especially those designed by researchers with sociology background, mainly discussed the potential causes of talent flows. For instance, at the national level, [2] and [20] proposed several factors that might influence the decision to migrate, such as economic, family and so on. Besides, a few studies investigated the national talent flow issues with the consideration of "brain drain" in several countries [2], [10], [31]. In addition to these macro-level talent flow analysis, [8] analyzed the factors for influencing individual job hopping, such as the salary level, job satisfaction and so on. On the other hand, with the prevalence of online professional networks (OPNs), a large number of digital resumes of talents can be collected. With these kind of data, there emerges a tend of conducting talent-related studies with machine learning methods. For instance, [21] and [23] predicted the individual job changes for understanding the professional careers of talents. [27] aimed to find the professional similarity between talents. [3] predicted the salary at company level by using talent transition data. [24] proposed the concept of job transition network (JTN) and extracted talent circle for talent recruitment. Recently, [25] extended the JTN with a dynamic setting and focused on the prediction of edge weights in dynamic JTN. Different from the above literatures, in this paper, we aim to introduce a big data-driven approach for predictive talent flow analysis, i.e., forecasting the future talent flows in a fine-grained manner.

Time-Dependent Collaborative Filtering. Latent factor model are widely used [29] [30], and the technical solution of this paper follows the idea of the model based collaborative filtering, which maps both preferences of users and items into the low-dimensional vector space, and conducts recommendations based on the learned latent factors [13]. However, since user preferences usually change over time, it is important to take the dynamics of preferences into account [6] [18] [26]. For instance, [12] proposed the TimeSVD++, which modeled the temporal dynamics of user latent vectors by summarizing three factors. [14] assumed the users' preferences would change smoothly over time, and the current preference of a user was the prior to future preference. [28] proposed the BTMF, which modeled the dynamics by learning the transition matrix for each user's latent vector between consecutive time windows. [19] modeled the dynamics of user's latent vector through a linear state

space model. In fact, it is equal to learning a shared transition matrix for all users' latent factors evolving while each user has a specific bias evolving vector. In this paper, we also try to learn the vectors for three components, i.e., origin company, destination company and job position. Meanwhile, we assume the vectors of companies are time-dependent. Different from the above works, we model the evolving process of companies' vector by considering two kinds of influence, i.e., previous influence and market influence, which is more suitable for the scenario of talent flow forecast.

Tensor Factorization. There are two kinds of popular tensor factorization models, namely the CANDECOMP/PARAFAC (CP) model [11] and the Tucker model [5]. Specifically, the CP model factorizes a tensor into the sum of outer products of n vectors, and the Tucker model factorized an n-dimensional tensor into a small core tensor and n matrices. In the literatures, there are some works on integrating the temporal information for tensor factorization. For instance, [22] created an additional latent feature vector for each time slice and modeled each entry of the tensor as the inner product of latent factors of user, item and time slice. [17] defined a new measure of user preference dynamics to capture the shifting rate for each user. And it further modeled users' preference dynamics and users' side information in coupled tensor factorization. [7] considered the bipartite graphs that evolved over time and demonstrated that tensor based methods were effective for temporal data with varying periodic patterns. Different from these methods, in this paper, we propose a novel approach to model the evolution of talent flows, instead of learning the vector of time slice directly.

6 CONCLUSION

In this paper, we studied the problem of talent flow forecast from a data-driven perspective. Specifically, we first constructed a time-aware job transition tensor by mining the large-scale job transition records of digital resumes in online professional networks (OPNs). Then, we designed a dynamic latent factor based Evolving Tensor Factorization (ETF) model for predicting the future talent flows. In particular, a novel evolving feature by jointly considering the influence of previous talent flows and global market was introduced for modeling the evolving principles of each company. Furthermore, to improve the predictive performance of ETF, we also integrated the company similarity as side information for regulating the model inference. Finally, extensive experiments conducted on a large-scale real-world dataset clearly validate the effectiveness of our approach.

7 ACKNOWLEDGMENTS

This research was partially supported by grants from the National Natural Science Foundation of China (Grant No. 91746301, U1605251, 71531001 and 61703386), the Anhui Provincial Natural Science Foundation (Grant No. 1708085QF140) and K.C.Wong Education Foundation.

REFERENCES

[1] Léon Bottou. 2010. Large-scale machine learning with stochastic gradient descent. In *Proceedings of COMPSTAT'2010*. Springer, 177–186.
[2] Stuart C Carr, Kerr Inkson, and Kaye Thorn. 2005. From global careers to talent flow: Reinterpreting ?brain drain? *Journal of World Business* 40, 4 (2005), 386–398.
[3] Xi Chen, Yiqun Liu, Liang Zhang, and Krishnaram Kenthapadi. 2018. How LinkedIn Economic Graph Bonds Information and Product: Applications in LinkedIn Salary. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. ACM, 120–129.
[4] Thomas H Davenport, Jeanne Harris, and Jeremy Shapiro. 2010. Competing on talent analytics. *Harvard business review* 88, 10 (2010), 52–58.
[5] Lieven De Lathauwer, Bart De Moor, and Joos Vandewalle. 2000. A multilinear singular value decomposition. *SIAM journal on Matrix Analysis and Applications* 21, 4 (2000), 1253–1278.
[6] Yi Ding and Xue Li. 2005. Time weight collaborative filtering. In *Proceedings of the 14th ACM international conference on Information and knowledge management*. ACM, 485–492.
[7] Daniel M Dunlavy, Tamara G Kolda, and Evrim Acar. 2011. Temporal link prediction using matrix and tensor factorizations. *ACM Transactions on Knowledge Discovery from Data (TKDD)* 5, 2 (2011), 10.
[8] Daniel C Feldman and Thomas WH Ng. 2007. Careers: Mobility, embeddedness, and success. *Journal of management* 33, 3 (2007), 350–377.
[9] Tilmann Gneiting. 2011. Making and evaluating point forecasts. *J. Amer. Statist. Assoc.* 106, 494 (2011), 746–762.
[10] Kerr Inkson. 2005. Exploring the dynamics of New Zealand's talent flow. *New Zealand Journal of Psychology* 34, 2 (2005), 110–116.
[11] Tamara G Kolda and Brett W Bader. 2009. Tensor decompositions and applications. *SIAM review* 51, 3 (2009), 455–500.
[12] Yehuda Koren. 2009. Collaborative filtering with temporal dynamics. In *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 447–456.
[13] Yehuda Koren, Robert Bell, and Chris Volinsky. 2009. Matrix factorization techniques for recommender systems. *Computer* 8 (2009), 30–37.
[14] Ruijiang Li, Bin Li, Cheng Jin, Xiangyang Xue, and Xingquan Zhu. 2011. Tracking User-Preference Varying Speed in Collaborative Filtering.. In *AAAI*.
[15] Chih-Jen Lin. 2007. Projected gradient methods for nonnegative matrix factorization. *Neural computation* 19, 10 (2007), 2756–2779.
[16] Andriy Mnih and Ruslan R Salakhutdinov. 2008. Probabilistic matrix factorization. In *Advances in neural information processing systems*. 1257–1264.
[17] Dimitrios Rafailidis and Alexandros Nanopoulos. 2014. Modeling the dynamics of user preferences in coupled tensor factorization. In *Proceedings of the 8th ACM Conference on Recommender systems*. ACM, 321–324.
[18] Steffen Rendle, Christoph Freudenthaler, and Lars Schmidt-Thieme. 2010. Factorizing personalized markov chains for next-basket recommendation. In *Proceedings of the 19th international conference on World wide web*. ACM, 811–820.
[19] John Z Sun, Dhruv Parthasarathy, and Kush R Varshney. 2014. Collaborative Kalman Filtering for Dynamic Matrix Factorization. *IEEE Trans. Signal Processing* 62, 14 (2014), 3499–3509.
[20] Ibraiz Tarique and Randall S Schuler. 2010. Global talent management: Literature review, integrative framework, and suggestions for further research. *Journal of world business* 45, 2 (2010), 122–133.
[21] 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.
[22] Liang Xiong, Xi Chen, Tzu-Kuo Huang, Jeff Schneider, and Jaime G Carbonell. 2010. Temporal collaborative filtering with bayesian probabilistic tensor factorization. In *Proceedings of the 2010 SIAM Internationxal Conference on Data Mining*. SIAM, 211–222.
[23] Huang Xu, Zhiwen Yu, Hui Xiong, Bin Guo, and Hengshu Zhu. 2015. Learning career mobility and human activity patterns for job change analysis. In *2015 IEEE International Conference on Data Mining (ICDM)*. IEEE, 1057–1062.
[24] 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.
[25] Huang Xu, Zhiwen Yu, Jingyuan Yang, Hui Xiong, and Hengshu Zhu. 2018. Dynamic Talent Flow Analysis with Deep Sequence Prediction Modeling. *IEEE Transactions on Knowledge and Data Engineering* (2018).
[26] Tong Xu, Hengshu Zhu, Hao Zhong, Guannan Liu, Hui Xiong, and Enhong Chen. 2018. Exploiting the Dynamic Mutual Influence for Predicting Social Event Participation. *IEEE Transactions on Knowledge and Data Engineering* (2018).
[27] Ye Xu, Zang Li, Abhishek Gupta, Ahmet Bugdayci, and Anmol Bhasin. 2014. Modeling professional similarity by mining professional career trajectories. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 1945–1954.
[28] Chenyi Zhang, Ke Wang, Hongkun Yu, Jianling Sun, and Ee-Peng Lim. 2014. Latent factor transition for dynamic collaborative filtering. In *Proceedings of the 2014 SIAM International Conference on Data Mining*. SIAM, 452–460.
[29] Chen Zhu, Hengshu Zhu, Yong Ge, Enhong Chen, and Qi Liu. 2014. Tracking the evolution of social emotions: A time-aware topic modeling perspective. In *2014 IEEE International Conference on Data Mining*. IEEE, 697–706.
[30] 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.
[31] David Zweig. 2006. Competing for talent: China's strategies to reverse the brain drain. *International labour review* 145, 1-2 (2006), 65–90.