Latent Ability Model: A Generative Probabilistic Learning Framework for Workforce Analytics

Zhiling Luo, *Member, IEEE,* Ling Liu, *Fellow, IEEE,* Jianwei Yin, *Member, IEEE,* Ying Li, and Zhaohui Wu

Abstract—As more business workflow systems being deployed in modern enterprises and organizations, more employee-activity log data are being collected and analyzed. In this paper we develop a latent ability model (LAM) as a generative probabilistic learning framework for workforce analytics over employee-activity logs. The LAM development is novel in three aspects. First, we introduce the concept of latent ability variables to model hidden relations between employees and activities in terms of job performance, such as the set of skills provided by an employee and the set of skills required by an activity, and how well they matchup in employee-activity assignment. Second, we construct the latent ability model (LAM) by learning latent ability parameters from the employee-activity log data using expectation-maximization and gradient descent. Finally, we leverage LAM to build inference and prediction models for employee performance prediction, employee ability comparison, and employee-activity matchup quality estimation. We evaluate the accuracy and efficiency of our approach using real log datasets collected from a workflow system deployed in the government of city Hangzhou in China, which consists of 5,287,621 log records over two years involving 744 activities and 1725 employees. We show that LAM approach outperforms existing representative methods in both accuracy and efficiency.

Index Terms—Workforce Analytics, Generative Model, Graphical Model, Latent Variable Model

1 Introduction

Workforce analytics is a data-driven statistical learning methodology that employs statistical models and machine learning algorithms to worker-related data logs, enabling enterprise organizations to optimize their talent pools and transform human resource management [1], [2], [3]. Just like servers in large scale computer systems, employees are the basic operating units in modern enterprises and organizations [4], [5], [6], [7], [8], [9]. The performance of a computer system is measured universally based on the types of workloads using a set of well-known performance metrics, such as throughput and latency. However, unlike computer systems, predicting the performance of employees based on the activities and tasks they have performed is known to be difficult [10] and yet it is on the top of the wish-list for many enterprise leaders. Example problems include: *can we predict how many tasks that one employee can do in the next month? Can we forecast whether a group of three employees is sufficient for a time-sensitive task?* Such employees' performance prediction problems are significantly more challenging for a number of reasons. First, comparing with computer servers, human behavior exhibits a much broader spectrum of uncertainty because human performance is influenced by a wide range of factors, many of which are implicit and hidden variables [11], [12], [13], [14]. Second, the human behavior related to employee's performance and satisfaction is dominated by work-related abilities of individuals, and how well employees' provided abilities and task-required abilities are matched in the current employee-activity assignments. Thus, simple statistical metrics, such as throughput and latency of employee-activity task execution time, are not suitable and insufficient as the core indicators for predicting employee performance. We argue that the problem should be addressed by applying latent variable based statistical learning models over the employee-activity log, and by extracting hidden variables and parameters that have statistical impact on employee-activity performance.

Example 1.1 *Consider the Employee-Activity (EA) service-time in Table 1(b), which is derived from a sample set of employee-activity log records in Table 1(a). We observe that employee E0001 performed two activities of A0001 and A0002. Employee E0002 only performed activity A0001. Even though E0002 has not worked on A0002 before, we are interested in questions like: can we predict the potential service time of employee E0002 for performing activity A0002? Can we estimate the probability that E0002 could finish A0002 use less time than the average service time of all other employees on A0002, i.e.,* $(430+410+400)/3 = 413$ *in this example?*

Problems with existing approaches. This workforce analysis problem belongs to the class of prediction problems based on unsupervised learning. Collaborative Filtering (CF) [15] is the most representative unsupervised learning method. We will illustrate the utility of the CF method and the hidden problems of using CF for such type of workforce analysis problems. In general, a CF method will predict the service time of an employee, say E0002, on a new activity, say A0002, by summarizing the service time data of all other similar employees on this activity. One way to measure the similarity of employees is the pairwise weighted similarity of their performance on the set of common activities. With CF+AVG, one can predict the potential service time by averaging of the weighted sum of similar employees' service time data for this activity. In our example, only E0001 has performed A0002, and both E0001 and E0002 have performed the common activity A0001. Thus, the CF+AVG estimates that E0002 could finish A0002 by the average service time of E0001 over the three log records on A0002: $(430 + 410 + 400)/3 = 413$. However, this CF-based prediction formulation suffers severely, and does not work well in the presence of skewed data distribution [16]. Concretely, if the set of common activities between a pair of employees is significantly smaller compared to the total set of activities performed by these two employees, thus there is highly skewed data distribution exists in employee-activity relations, then such common-set based similarity measure is inaccurate and ineffective for measuring pairwise similarity of employees with respect to their performance on activities.

Example 1.2 *Consider Tab. 1, employee E0001 has three records about A0001 with service time of 181, 803 and 190 respectively. The service time of 803 seconds is clearly abnormal. Many factors may have caused this skewedness, e.g., E0001 took a short break. Assume that employee E0005 has never performed activity A0001. To estimate the possible service time of E0005 for activity A0001, by CF+AVG, we can predict the service time of E0005 for A0001 based on all of its similar employees' service time data. If E0001 is the only one similar to E0005 and has performed activity A0001, then the average service time of E0001 on A0001, i.e., (181+803+190)/3=391.3, is the estimated service time for E0005 to perform A0001. However, this CF averaging leads to incorrect prediction result, which doubled the usual service time (less than 200 seconds) of E0001 on activity A0001, due to the presence of skewed service time of 803 seconds.*

The two examples show that different employees may perform the same activity with varying service times and an employee may perform the same activity with varying service time at different times. This indicates that the service time in the log is a complex feature and its value distribution over its domain exhibits some uncertainty and randomness due to hidden relations between employees and activities. Such latent features contribute to the complexity of predicting employee's service time on new activities. Thus, such highly skewedness and random uncertainty in the employee-activity-service_time dataset can severely degrade the effectiveness and accuracy of the existing methods.

Existing literature studies such randomness in the inference features (e.g., service time) by manually defining some performance indicators. For example, [7] considers the subjective factor, e.g., the diligence of employee and the objective factor, e.g., the complexity of activity. [12], [14] addresses the problem by requiring manually identifying whether an employee satisfies the ability requirement of an activity. For instance, to find out whether an employee is good at communication, one needs to pre-define what the communication ability is and how it is measured and then manually give a score on this ability for each pair of employee and activity. These approaches are clearly subjective and not scalable. In this paper, we show that it is important and feasible to develop a statistical inference model to learn the hidden factors that influence the performance of employees on their assigned activities, e.g., the service time.

In this paper, we present the *Latent Ability Model (LAM)* as a generative probabilistic learning framework by introducing latent variables to capture the randomness of service time by modeling it as a stochastic value generated based on the probability distribution of latent variables. We make three original contributions. First, we introduce latent ability variables to model the types of hidden quantities that may influence the prediction accuracy of service time. We capture the abilities provided by employees in performing assigned activities, and the abilities required by activities performed by employees of different ages and gender. Second, we construct a latent ability model (LAM) to encode the hidden quantities in a joint probability distribution of observed and latent random variables. Given the EA service-time log records as prior, we can uncover the particular hidden quantities by using an inference algorithm to approximate the posterior - the conditional distribution of the hidden variables given the observations. Finally, we use the posterior and the latent parameters learned to construct the predictive distribution, the distribution over future data based on LAM and the observations from the service time log. To the best of our knowledge, the LAM enabled probabilistic learning framework is the first to use the latent ability model as the core of a generative probabilistic process to learn latent features from the employee-activity log data, instead of relying on a predefined subjective set of skills/abilities [7], [12], [14].

2 PROBLEM FORMULATION

2.1 Real Datasets and Example Wish List

The dataset used in the study is obtained from the municipal government of city Hangzhou in south China, which consists of a total of 8 organizations of employee-activity log datasets. A detailed description of these datasets is given in Section 6. Table 1 provides sample log records, Table 2 provides sample employee records and sample activity records, and Table 3 is a summary of some statistical characteristics of the eight datasets. The mission of this workforce analytics project is to help the municipal government to improve the workforce efficiency by mining the employee-activity logs. The desirable example wish list includes questions such as (i) are the current employee-activity assignments effective? (ii) where and what can we do to improve the overall organizational efficiency? (iii) which activities need to allocate more skillful employees? (iv) how to compare the performance of different employees and find out the most skillful employees in our organization? Bearing these example questions in mind, we develop a general-purpose latent ability model based learning framework, which can facilitate the statistical learning process from multiple dimensions by combining both observable and latent variables as well as hidden patterns embedded in the employee-activity log datasets.

Note that *ServiceTime* in Tab. 1(b) is a *derived aggregate feature* based on the *StartTime* and *EndTime* in Tab. 1(a). In the rest of the paper, we refer to the employee-activity log in the format of Tab. 1(b). We use n to denote the number of records in the employee-activity log $\boldsymbol{L}$, n_a and n_e denote the number of activities in the activity table and the number of employees in the employee table respectively. Each record $x \in \boldsymbol{L}$ is defined by a triple (a, e, s) where a is the ActivityID, e is the EmployeeID and s is the ServiceTime.

2.2 Learning Complex Latent Features

From Examples 1.1 and 1.2, we see that the service time is a complex variable which may be modeled as a mixture of latent features. Although the factors that can lead to the variation and skewed distribution of service time can be many, the most common and dominating factors that are related to the service time log are the set of abilities provided by employees in performing the activities, and the set of abilities required by activities for successful execution in an organization. Instead of subjectively and manually defining such abilities as done in existing literature [7], [12], [14], [11], we propose to design a latent ability learning algorithm that can automatically discover and infer such hidden quantities based on the service time log through a LAM based generative probabilistic inference framework. We treat these hidden quantities as the latent ability variables, which may influence both the past and future service time for any pair of employee and activity.

TABLE 1: The Employee-Activity logs

(a) Raw Employee-Activity log data

RecordID	ActivityID	EmployeeID	StartTime	CompleteTime
R0001	A0001	E0001	2014/9/10 15:10:33	2014/9/10 15:13:34
R0002	A0001	E0001	2014/9/10 15:21:10	2014/9/10 15:34:33
R0003	A0001	E0001	2014/9/10 15:40:12	2014/9/10 15:43:22
R0004	A0001	E0002	2014/9/10 15:50:01	2014/9/10 15:54:51
R0005	A0001	E0002	2014/9/10 16:01:03	2014/9/10 16:04:23
R0006	A0001	E0002	2014/9/10 16:12:33	2014/9/10 16:15:33
R0007	A0002	E0001	2014/9/10 16:16:10	2014/9/10 16:23:20
R0008	A0002	E0001	2014/9/10 16:25:12	2014/9/10 16:32:02
R0009	A0002	E0001	2014/9/10 16:32:27	2014/9/10 16:39:07
R0010	A0003	E0001	2014/9/10 17:03:45	2014/9/10 17:09:27
R0011	A0003	E0002	2014/9/10 17:10:06	2014/9/10 17:15:18
R0012	A0003	E0001	2014/9/10 17:16:20	2014/9/10 17:19:44
R0013	A0003	E0003	2014/9/10 17:20:20	2014/9/10 17:23:41

(b) Employee-Activity service time table

ActivityID	EmployeeID	ServiceTime(s)
A0001	E0001	181
A0001	E0001	803
A0001	E0001	190
A0001	E0002	290
A0001	E0002	260
A0001	E0002	240
A0002	E0001	430
A0002	E0001	410
A0002	E0001	400
A0003	E0003	342
A0003	E0003	312
A0003	E0002	204
A0003	E0002	201

TABLE 2: The sample fragment of context information about employees and activities

(a) Activity table

ActivityID	Name	Business
A0001	Application Checking	Real Estate Transaction
A0002	Advanced Review	Real Estate Transaction
A0003	Preliminary Review	Real Estate Transaction
A0004	Application Checking	Land Leasing

(b) Employee table

EmployeeID	Name	Gender	Birthday
E0001	C. Zhou	Female	1985/8/9
E0002	P. Wu	Male	1965/12/10
E0003	J. Wang	Male	1989/9/23
E0004	D. Chen	Female	1990/1/3

TABLE 3: Seven district government datasets and a central department dataset

	ShangCheng District (SC)	XiaCheng District (XC)	XiHu District (XH)	GongShu District (GS)	JiangGan District (JG)	BinJiang District (BJ)	ZhiJiang District (ZJ)	HangZhou Central (HZ)
# of Activity	75	5	92	80	5	175	155	155
# of Employee	51	45	312	91	44	456	435	291
# of Records	4641	741	535570	5375	720	1728413	1378838	1633323
# of Emp per Act (Min,Avg,Max)	1,6.8,20	2,14.0,26	1,14.8,172	1,7.2,26	1,14.1,26	1,25.0,177	1,26.9,259	1,28.2,177
# of Act per Emp (Min,Avg,Max)	1,9.6,45	1,1.6,4	1,4.2,52	1,6.1,45	1,1.5,5	1,9.1,57	1,8.5,40	1,14.4,53
# of Records per Pair of Act and Emp	1.26	3.37	19.29	0.77	3.27	22.97	22.96	37.67
Percentage of Recorded Act and Emp Pair	1.96%	2.22%	0.32%	1.10%	2.27%	0.22%	0.23%	0.34%

2.2.1 Latent Ability Variables

Definition 1 *Let $\boldsymbol{L} =< \boldsymbol{A}, \boldsymbol{E}, \boldsymbol{S} >$ denote the log of n employee-activity service time records of the form (a_i, e_j, s_{ij}), $a_i \in \boldsymbol{A}$, $1 \leq i \leq n_a$, $e_j \in \boldsymbol{E}$, $1 \leq j \leq n_e$, $s_{ij} \in \boldsymbol{S}$ ($|\boldsymbol{S}| = n$). Let system-supplied parameter m denote the number of ability variables ($m \ll n$). The ability set $\boldsymbol{B} = \{b_k\}$ ($1 \leq k \leq m$) contains m ability variables that are required by activities and that are provided by employees for performing activities.*

Although m is a system-defined parameter, the larger m will lead to higher cost of learning. Our experiments show that for a given employee-activity log dataset, one can find a near optimal value of m, which gives stable and high log likelihood (accuracy) for our latent ability model based probabilistic learning framework (see Section 6). Note that the ability set $\boldsymbol{B}$ is neither predefined nor obtained directly from the log $\boldsymbol{L}$. Intuitively, given a set of activities, if an employee had higher conditional probability distribution on the set of abilities required by the activities, then we defined the employee's provided abilities accordingly. Similarly, we can define the required abilities of an activity by the conditional probability distribution on the set of abilities provided by all the employees who have performed this activity. Naturally, if an employee has a better score on the set of abilities than that of another employee, then we can predict that the former employee usually uses less service time to complete the given activity. Thus, ability can be modeled as latent variable to connect activity and employee in the context of service time, to capture the latent relation between employee and service time, as well as the latent relation between activity and service time.

We below introduce two statistical parameters of ability. The first property is the frequency of ability required by the given set of activities and the frequency of ability provided by the given set of employees. The second property is the probability of activity assigning to (requiring) each ability variable and the probability of employee assigning to (providing) each ability variable. Through statistical learning algorithm, we discover and encode the relationship between $\boldsymbol{B}$ and employees and between $\boldsymbol{B}$ and activities.

Definition 2 *(**Ability's Frequency on Activity or Employee**) Given $\boldsymbol{L}$ and $\boldsymbol{B}$, we define the frequency of b_i required by all the activities in $\boldsymbol{L}$ by $\theta_{a,\{i\}}$, and the frequency of b_i provided by all the employees in $\boldsymbol{L}$ by $\theta_{e,\{i\}}$.*

To smooth these two frequency variables, we introduce the

Dirichlet Distribution with parameter α as the prior distribution. The prior distribution for ability on activity is $Q(\theta_a;\alpha) = \prod^m(\theta_{a\{i\}})^{\alpha-1}/B(\alpha)$ and the prior distribution for ability on employee is $Q(\theta_e;\alpha)$.

Definition 3 *(***Assignment Probability on Ability***) Given $\boldsymbol{L}$ and $\boldsymbol{B}$, the probability of assigning ability b_i to activity a_j is defined by $P(a_j|b_i)$ such that $\forall b_i \in \boldsymbol{B}, 1 \le i \le m, \sum_{j=1}^{n} P(a_j|b_i) = 1$. Similarly, the probability of assigning ability b_i to employee e_j is defined by $P(e_j|b_i)$ such that $\forall b_i \in \boldsymbol{B}, 1 \le i \le m, \sum_{j=1}^{n} P(e_j|b_i) = 1$. For presentation brevity, we denote $P(a_j|b_i)$ by $\beta_{a\{i,j\}}$ and $P(e_j|b_i)$ as $\beta_{e\{i,j\}}$.*

We use the matrix $\boldsymbol{\beta}_a = \{\beta_{a\{i,j\}}\}$ to denote the probability for assigning all m abilities to all n_a activities ($1 \le i \le m$, $1 \le j \le n_a$) and the matrix $\boldsymbol{\beta}_e = \{\beta_{e\{i,j\}}\}$ to denote the probability for assigning all m abilities to all n_e employees. Note that the sum of each ability row i is 1 for both $\boldsymbol{\beta}_a$ and $\boldsymbol{\beta}_e$ and such constraint does not apply for the activity columns in $\boldsymbol{\beta}_a$ or employee columns in $\boldsymbol{\beta}_e$. In fact, the values in a column of $\boldsymbol{\beta}_a$ and $\boldsymbol{\beta}_e$ are independent of each other.

Note that both frequency parameters θ_a, θ_e and assignment parameters $\boldsymbol{\beta}_a, \boldsymbol{\beta}_e$ are unknown in the beginning. There is a number of of ways to assign the initial distribution for $\boldsymbol{\beta}_a$ and $\boldsymbol{\beta}_e$. Different initial settings will result in the same final value though they may have different convergence rate. Their final values are obtained through iterative learning.

Example 2.1 *Consider the running example in Table 1, with $m = 3$, $\boldsymbol{B} = \{b_1, b_2, b_3\}$, $\alpha = 5$, we initialize $\theta_a, \theta_e, \beta_a, \beta_e$ as shown in top left of Figure 1. After iterative learning of these latent ability parameters, the final results of $\theta_a, \theta_e, \beta_a, \beta_e$ are given in the bottom left of Figure 1.*

2.2.2 Latent Parameters for Service Time

To model the actual impact of different ability frequencies and different ability assignments on service time prediction, we identify the following three contributing factors, which are statistical properties of service time. We call them the **service time parameters**. We assume that service time is sampled from an exponential distribution that related to these three factors:

The **complexity factor of activity**, denoted by c_a, represents activity's sophistication factor on service time. c_a is a vector of size n_a, with c_{a_j} as the complexity factor of the j^{th} activity a_j ($1 \le j \le n_a$). Intuitively, an activity has a higher complexity factor than another if it takes more service time no matter which employee performs the activity.

The **complexity factor of employee**, denoted by c_e, represents employee's sophistication factor on service time. c_e is a vector of size n_e, with c_{e_k} as the complexity factor of employee e_k ($1 \le k \le n_e$). An employee has a higher complexity factor than another, if this employee uses less service time no matter which activity he/she is assigned to.

Penalty from ability mismatch, denoted by ω, represents the amount of penalty for employee-activity ability mismatch. The higher penalty is given if an employee is considered mismatched for an activity if his provided ability quantities are not matched well to the required ability quantities of the activity. Consequently this employee takes longer service time than the others. ω is a global parameter contributing to all employees and all activities.

Note that the values of c_a, c_e, ω are positive real numbers. There are several ways to set the initial values for these parameters, all of which will result in the same final value but with different convergence rates.

Let the function ϕ denote the probability density function on exponential distribution with parameter λ, j denote the activity and k denote the employee in the i^{th} record of $\boldsymbol{L}$. We model the relation between latent ability and service time by using the above three parameters and the exponential distribution:

$$\phi(s_i; \lambda_{i,j,k}) = \lambda_{i,j,k} \exp(-\lambda_{i,j,k} s_i) \quad (1)$$

where

$$\lambda_{i,j,k}^{-1} = \begin{cases} c_{a_j} c_{e_k} \text{ if } \beta_{a\{q,j\}} = \beta_{e\{q,k\}}, \forall q \in \{1, ..., m\} \\ c_{a_j} d_{e_k} \omega \text{ otherwise} \end{cases} \quad (2)$$

Consider a situation such that if we have the correct assignment for each employee-activity pair on ability, then all the service time values in the employee-activity log should fit exactly the exponential distribution ϕ with final values of c_a, c_e and ω. Intuitively, these factors contribute to the relationship between service time and employee/activity assignment on ability and thus the distribution of service time values. When the value of **required ability** variable (activity) matches that of **provided ability** variable (employee), we use an exponential distribution, whose expectation is the multiplication of c_a and c_e, as the distribution of **service time**. Otherwise, we use the exponential distribution, whose expectation is the multiplication of all three parameters: c_a, c_e, ω.

2.3 Learning Objectives

To complete the problem formulation, in this section we define our primary learning objectives for workforce analytics over the employee-activity service time log.

Definition 4 (Performance Prediction) *Let $\boldsymbol{E}$ denote the employee table of n_e records, $\boldsymbol{A}$ denote the activity table of n_a records, and $\boldsymbol{L} = \{(e_j, a_k, s_i)|e_j \in \boldsymbol{E}, a_k \in \boldsymbol{A}, s_i \in \mathbb{R}\}$ denote the employee-activity service time log of n records ($1 \le i \le n$, $1 \le j \le n_e$, $1 \le k \le n_a$). We construct the latent ability model (LAM) as a generative probabilistic learning framework, denoted by φ, such that for any pair of employee and activity, i.e., $\forall e_j \in \boldsymbol{E}, a_k \in \boldsymbol{A}$, we can predict the service time s by employing the LAM learning algorithm $\varphi(e_j, a_k)$ over $\boldsymbol{L}$ if $(e_j, a_k, s) \notin \boldsymbol{L}$.*

The next learning objective is employee ability estimation, which is closely related to the employee performance prediction, our primary learning goal.

Definition 5 (Employee Ability Estimation) *Given the EA working log $\boldsymbol{L}$, we want to build a statistical inference model that can estimate the ability of an employee based on the set of activities for which this employee has performed and the service time of other employees on the same set of activities.*

The employee's ability score may help the organization to study the performance and the job satisfactory level of its employees, especially whether an employee's ability score the requirements of the activities assigned to him/her in predicting the performance. The third learning objective is employee-activity ability match-up score, which can be seen as an intuitive extension to the previous two learning objectives.

Definition 6 (Employee-Activity Matchup Score) *Given the EA working log $\boldsymbol{L}$, we want to build a learning model to find out the activity-employee ability matchup score for any*

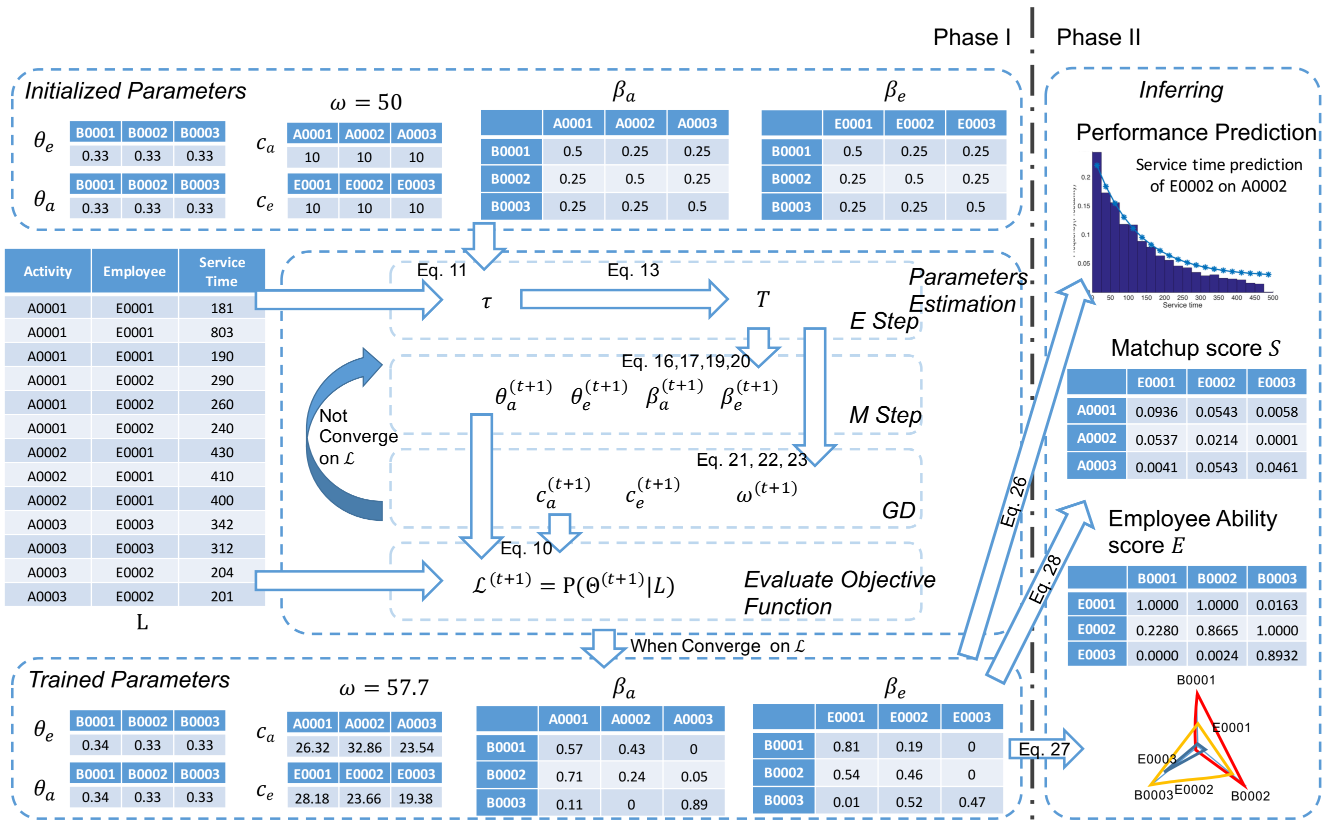


Fig. 1: Illustration of the LAM learning framework by a simple example. Phase I: LAM learns to estimate the set of latent parameters about its m ability variables: $\boldsymbol{\Theta} = \{\boldsymbol{\theta}_a, \boldsymbol{\theta}_e, \boldsymbol{\beta}_a, \boldsymbol{\beta}_e, \boldsymbol{c_a}, \boldsymbol{c_e}, \boldsymbol{\omega}\}$, initialized before iterative learning. Four steps in iteration t: E-step updates the conditional expectation $Q(\boldsymbol{\Theta}|\boldsymbol{\Theta}^{(t)})$; M-step updates $\theta_a^{(t+1)}$, $\theta_e^{(t+1)}$, $\beta_a^{(t+1)}$ and $\beta_e^{(t+1)}$. GD step updates $c_a^{(t+1)}$, $c_e^{(t+1)}$ and $\omega^{(t+1)}$. Evaluation-step computes the objective function $\mathcal{L}$, check if $\mathcal{L}$ coverages, output the extracted latent features, otherwise go to iteration $(t+1)$. Phase II: LAM prediction using the latent features learned. The performance of E0002 on A0002 can be predicted by the probability (y-axis) of service time for a range of increasing values (x-axis), the matchup score of for any employee-activity pair can be estimated, and the employee ability score can be inferred for all m ability components.

activity-employee pair. This score shows how suitable an employee-activity assignment is.

Example 2.2 *Consider our running example in Table 1, compared to E0003, employee E0002 has consistently shorter service time for activity A0003. From the Employee Ability Score $\boldsymbol{E}$ shown in Figure 1, based on the log $\boldsymbol{L}$, our LAM prediction algorithm (see Section 5.3) gives E0002 higher score on all abilities than E0003, e.g., for activity A0003, E0002 has a higher matchup score of 0.0543 comparing to E0003, with matchup score of 0.0461.*

3 Framework Overview

We have defined three learning objectives, solving three different problems in the workforce analytics over an employee-activity log. By introducing the latent ability variables and the set of latent parameters $\theta_a, \theta_a, \beta_a, \beta_e, c_a, c_d, \omega$, these seemingly different problems become closely related. Also for our learning objectives, the only important matter regarding m ability variables is the difference of one ability from another with respect to the three types of observations in the log: activity, employee and service-time. We build a latent ability model (LAM) based on observations, latent ability variables, latent parameters, and their dependence relations through a generative probabilistic process. Instead of considering all possible quantities of ability variable, we employ a graphical model on both observations and latent ability variables. This constrains the domain of ability to m mixture components. The mixture model assumes that the log data are clustered and that each data point is drawn from a distribution associated with its assigned cluster. The hidden variables of the model are the cluster assignments and (hyper) parameters to the pre-cluster distributions. The inference algorithm learns the latent variables behind the observable variables, and encode their relationship by a joint probability distribution of hidden and observed variables. Given the observations in $\boldsymbol{L}$, by getting a prior probability for each random variable, we can uncover the particular latent variables through the posterior, the conditional distribution of the latent variables. This enables us to formally describe how the hidden variables and observations interact in a probability distribution. This posterior reveals hidden structure in our service time log data: it clusters the data points into m groups and describes the location (i.e., the mean) of each group. The final goal of this inference is to use the posterior to construct the predictive distribution, derived from the posterior, which provides the distribution of the future data points that the observation and the model imply.

Given the training data with ground truth, the generative model can be trained by tuning the parameters to maximize the posterior probability. This parameter estimation method is called the **Maximizing a Posterior** (MAP). After completion of training, the generative model can infer the probability of future data taking values from its domain.

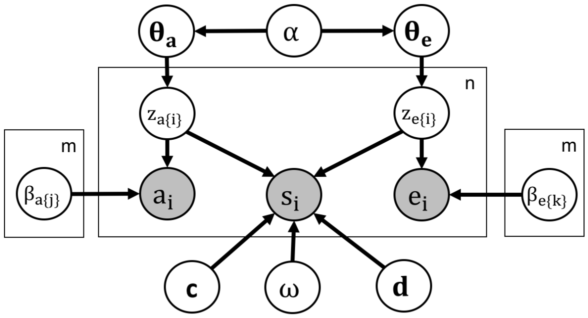


Fig. 2: The graphical model of LAM generative process.

The LAM analytic framework performs probabilistic learning and prediction in two phases. The first phase is dedicated to latent feature extraction through generative probabilistic inference with the objective of building a prediction model. The second phase leverages the latent features and parameters learned from the first phase to predict employee performance, estimate employee's ability and compute employee-activity matchup score. These two phases of inference shares the same probabilistic inference model − LAM on the hidden relations among employee, activity, service time and ability. Fig. 1 provides an illustrative sketch of this two phase probabilistic learning framework.

4 LAM CONSTRUCTION

4.1 The Generative Process

We construct the latent ability model using a generative probabilistic process, as shown in Fig. 2. It describes how the latent ability variables interact with the observation variables to govern the distribution of the observations. The circle represents a variable and the direct edge from one circle to another represents the dependency of the latter variable on the former one. The grey circle represents an observation variable. The rectangle labeled with symbol n or m represents a process that repeats for n or m times. A summary of the key notations is provided in Tab. 4.

In the generative process of LAM, the mixture components $\boldsymbol{\beta}_a$ and $\boldsymbol{\beta}_e$, the mixture proportions $\boldsymbol{\theta}_a$ and $\boldsymbol{\theta}_e$, and the service time parameters c_a, c_e, ω are learned iteratively. The mixture assignments z_a and z_e depend on the mixture proportions $\boldsymbol{\theta}_a$ and $\boldsymbol{\theta}_e$, which parameterize the distribution of the mixture ability assignment to activity $\boldsymbol{\beta}_a$ and the distribution of the mixture ability assignment to employee $\boldsymbol{\beta}_e$ respectively. Given log $\boldsymbol{L}$, an observation $x_i = (\boldsymbol{a_i}, \boldsymbol{e_i}, \boldsymbol{s_i}) \in \boldsymbol{L}$ depends on both the mixture components $\boldsymbol{\beta}_a, \boldsymbol{\beta}_e$, and the mixture assignment $\boldsymbol{z}_{a\{i\}}, \boldsymbol{z}_{e\{i\}}$. The graphical model shown in Fig. 2 illustrates the structure of the factorized joint distribution and the flow of the generative process of LAM.

TABLE 4: Notations and descriptions

Notations	Descriptions
n, m	Record number and ability number.
n_a,n_e	Activity number and employee number.
$\boldsymbol{\theta_a}$	The vector in $\mathbb{R}^m$, whose element $\theta_{a\{i\}}$ denotes the probability of requiring ability i for all activities. $\sum_i \theta_{a\{i\}} = 1$.
$\boldsymbol{\theta_e}$	The vector in $\mathbb{R}^m$, whose element $\theta_{e\{i\}}$ denotes the probability of owning ability i for all employees. $\sum_i \theta_{e\{i\}} = 1$.
$\boldsymbol{\beta}_a$	The matrix in $\mathbb{R}^{m\times n_a}$, whose element $\beta_{a\{i,j\}}$ denotes the probability of activity j requiring ability i. $\forall i, \sum_j \beta_{a\{i,j\}} = 1$.
$\boldsymbol{\beta}_e$	The matrix in $\mathbb{R}^{m\times n_e}$, whose element $\beta_{e\{i,j\}}$ denotes the probability of employee j owns ability i. $\forall i, \sum_j \beta_{e\{i,j\}} = 1$.
z_a,z_e	Ability assignment on activity and employee.
$\boldsymbol{c_a}$, $\boldsymbol{c_e}$, ω	Activity complexity, employee complexity and mismatch penalty factor

4.1.1 Dirichlet Prior

Given $\boldsymbol{\theta_a}$, the frequency of mixture ability assignment on activity, and $\theta_{a\{i\}}$, the fraction of distinct activities requiring ability i, satisfying $\sum_{j=1}^{m} \theta_{a\{j\}} = 1$, we define the prior distribution of $\boldsymbol{\theta_a}$ and its probability density function by **Dirichlet Prior**:

$$\boldsymbol{\theta_a}|\boldsymbol{\alpha} \sim \text{Dirichlet}(\boldsymbol{\alpha}) \tag{3}$$

where $\boldsymbol{\alpha}$ is the prior parameter, which is set to a constant at initialization of the training phase. Dirichlet Prior is the smoothing approach to guarantee that the element value of $\boldsymbol{\theta_a}$ would not be too small or too big. Similarly, we smooth $\boldsymbol{\theta_e}$ by using the same Dirichlet Prior with the same α.

$$\boldsymbol{\theta_e}|\boldsymbol{\alpha} \sim \text{Dirichlet}(\boldsymbol{\alpha}) \tag{4}$$

This design is shown in Fig. 2 by $\boldsymbol{\alpha}$, $\boldsymbol{\theta_a}$, $\boldsymbol{\theta_e}$ and the direct edges between them, each edge conducts the conditional probability with respect to the two end nodes. Concretely, the edge from $\boldsymbol{\alpha}$ to $\boldsymbol{\theta_a}$ represents the conditional probability $P(\boldsymbol{\theta_a}|\boldsymbol{\alpha})$, which is the probability of $\boldsymbol{\theta_a}$ given $\boldsymbol{\alpha}$. The formalization of this conditional probability is exactly the Dirichlet Prior. Similarly the edge from $\boldsymbol{\alpha}$ to $\boldsymbol{\theta_e}$ represents the conditional probability $P(\boldsymbol{\theta_e}|\boldsymbol{\alpha})$.

4.1.2 Ability Sampling

This step is **sampling ability** from its frequency on activity and its frequency on employee. Given a log record $x_i = (a_i, e_i, s_i) \in \boldsymbol{L}$ ($i \in \{1, ..., n\}$), we consider the following sampling process: the required ability is a sample from the ability set of size m, according to the frequency $\boldsymbol{\theta_a}$, i.e., the probability of getting the required ability by a_i is exactly $\theta_{a\{i\}}$. Formally:

$$z_a|\boldsymbol{\theta_a} \sim \text{Discrete}(\boldsymbol{\theta_a}) \tag{5}$$

Similarly, the probability of offering the provided ability by e_i is exactly $\theta_{e\{i\}}$, thus we have

$$z_e|\boldsymbol{\theta_e} \sim \text{Discrete}(\boldsymbol{\theta_e}) \tag{6}$$

This step is repeated by n times with each $x_i \in \boldsymbol{L}$ as the input, as shown in Fig. 2.

4.1.3 Activity/Employee Ability Assignment

This step considers the probability of assigning ability z_a to activity a_i and assigning ability z_e to employee e_i for each log record $x_i = (a_i, e_i, s_i) \in \boldsymbol{L}$. To build the relation between the required ability z_a and a_i, we use the assignment probability $\boldsymbol{\beta_a}$ as the parameter. The activity a_i with ability $z_{a\{i\}}$ is sampled from the following distribution:

$$a_i|z_a, \boldsymbol{\beta_a} \sim \text{Discrete}(\beta_{a\{z_a\}}) \tag{7}$$

It means that the conditional probability of a_i given the required ability $z_{a\{i\}}$ is exactly $\beta_{a\{z_a\}}$. Similarly, we introduce the employee ability assignment probability $\boldsymbol{\beta_e}$ to get the conditional probability:

$$e_i|z_e, \boldsymbol{\beta_e} \sim \text{Discrete}(\beta_{e\{z_e\}}) \tag{8}$$

$\boldsymbol{\beta_a}$ and $\boldsymbol{\beta_e}$ have size $n_a \times m$ and $n_e \times m$ respectively with m independent mixture ability components. This step repeats n times, one for each log record x_i ($1 \leq i \leq n$), as shown in Fig. 2 by the edge from $\beta_{a\{j\}}$ to a_i and the edge $\beta_{e\{k\}}$ to e_i.

4.1.4 Service Time Sampling

The last step is **sampling service time** s_i given the log record $x_i \in \boldsymbol{L}$, the **required ability** z_a and the **provided ability** z_e and the three service time parameters: **activity complexity** $\boldsymbol{c_a}$, **employee complexity** $\boldsymbol{c_e}$ and **mismatch penalty** ω. For a log record $x_i = (a_i, e_i, s_i) \in \boldsymbol{L}$, we use the following conditional probability:

$$s_i|z_a, z_e, \boldsymbol{c_a}, \boldsymbol{c_e}, \omega \sim \phi(s_i; \lambda_{i,j,k}) \tag{9}$$

where ϕ is the exponential distribution whose expectation is λ as defined in Section 2.2.2. This formalization states that the service time follows an exponential distribution in expectation of the multiplication of activity complexity and employee complexity if the provided ability matches the required ability. Otherwise, the mismatch penalty should be taken into consideration.

4.1.5 Summary

The LAM generative process shown in Figure 2 is summarized as follows: Given the log $\boldsymbol{L}$ with n records, n_a activities, n_e employees, m latent ability components, both required ability z_a and provided ability z_e are latent variables of size m ($1 \leq j, k \leq m$). For activity a_i and employee e_i in $x_i =< a_i, e_i, s_i >\in \boldsymbol{L}$:

1. Draw required ability proportion $\boldsymbol{\theta_a}|\boldsymbol{\alpha} \sim \mathrm{Dir}(\boldsymbol{\alpha})$ for a_i.
2. Draw provided ability proportion $\boldsymbol{\theta_e}|\boldsymbol{\alpha} \sim \mathrm{Dir}(\boldsymbol{\alpha})$ for e_i.
3. Draw required ability sampling $z_a|\boldsymbol{\theta_a} \sim \mathrm{Discrete}(\boldsymbol{\theta_a})$.
4. Draw provided ability sampling $z_e|\boldsymbol{\theta_e} \sim \mathrm{Discrete}(\boldsymbol{\theta_e})$.
5. Draw activity assignment to ability $a_i|z_a, \boldsymbol{\beta_a} \sim \mathrm{Discrete}(\beta_{a\{z_a\}})$.
6. Draw employee assignment to ability $e_i|z_e, \boldsymbol{\beta_e} \sim \mathrm{Discrete}(\beta_{e\{z_e\}})$.
7. Draw service time sampling $s_i|z_a, z_e, \boldsymbol{c_a}, \boldsymbol{c_e}, \omega \sim \phi(s_i; \lambda_{i,j,k})$.

The generative process has the following latent parameters: $\boldsymbol{\theta_a}$, $\boldsymbol{\theta_e}$, $\boldsymbol{\beta_a}$, $\boldsymbol{\beta_e}$, $\boldsymbol{c_a}$, $\boldsymbol{c_e}$ and ω. We describe how we learn these latent features in the next section.

4.2 Parameter Estimation

In this section we use $\boldsymbol{\Theta} = (\boldsymbol{\theta_a}, \boldsymbol{\theta_e}, \boldsymbol{\beta_a}, \boldsymbol{\beta_e}, \boldsymbol{c_a}, \boldsymbol{c_e}, \omega)$ to denote the set of latent features involved in the generative process of our latent ability model, which are also the set of parameters to be learned through parameter estimation. Several parameter learning methods can be used to estimate parameters in $\boldsymbol{\Theta}$ from observation $\boldsymbol{L}$, such as *Maximum Likelihood Estimation* (MLE), *Maximum A Posterior* (MAP). In the first prototype of LAM, to smooth $\boldsymbol{\theta_a}$ and $\boldsymbol{\theta_e}$ by the same $\boldsymbol{\alpha}$, we choose MAP, which treats a parameter in $\boldsymbol{\Theta}$ as a random variable, assumes a prior probability of $\boldsymbol{\Theta}$: $P(\boldsymbol{\Theta})$, and uses the observation data $\boldsymbol{L}$ of n records to get posterior probability of $\boldsymbol{\Theta}$: $P(\boldsymbol{\Theta}|\boldsymbol{L})$. Concretely, the posterior probability is defined as follows:

$$\mathcal{L} = P(\boldsymbol{\Theta}|\boldsymbol{L}) = Z \prod_{i=1}^{n} \sum_{j=1}^{m} \sum_{k=1}^{m} \tau_{i,j,k} \phi(s_i; \lambda_{i,j,k}) \tag{10}$$

where

$$\tau_{i,j,k} = \beta_{a\{j,a_i\}} \beta_{e\{k,e_i\}} \theta_{a\{j\}} \theta_{e\{k\}} \frac{1}{B(\boldsymbol{\alpha})} \prod_{i'=1}^{m} (\theta_{a\{i'\}} \theta_{e\{i'\}})^{\alpha-1}. \tag{11}$$

Z is a constant for normalization and it keeps the sum of all probabilities equal to 1. Equation 10 is the product of n observations in $\boldsymbol{L}$, m ability-to-activity assignment probabilities for each of the n_a activities ($j = 1 : m$) and m ability-to-employee assignment probabilities for each of the n_e employees ($k = 1 : m$). The proof of this posterior equation is given in appendix.

To solve this MAP (Maximum A Posterior) problem, we employee the *expectation-maximization*(EM) algorithm to the subset of parameters $\boldsymbol{U} = \{\theta_a, \theta_e, \beta_a, \beta_e\}$, and gradient descent (GD) to the parameters $\boldsymbol{V} = \{\boldsymbol{c_a}, \boldsymbol{c_e}, \boldsymbol{\omega}\}$ at the same time. The EM algorithm searches the space of parameters by maximizing the expectation of log likelihood, denoted by $Q(\boldsymbol{\Theta}|\boldsymbol{\Theta}^{(t)})$. By considering the two latent variables $\boldsymbol{U}$ and $\boldsymbol{V}$, we compute the posterior as follows:

$$\begin{aligned} &P(\boldsymbol{\Theta}|\boldsymbol{L}, \boldsymbol{U}, \boldsymbol{V}) \\ &= Z \prod_{i=1}^{n} \sum_{z_a=1}^{m} \sum_{z_e=1}^{m} I(u_i = j) I(v_i = k) \tau_{i,j,k} \phi(s_i; \lambda_{i,j,k}) \end{aligned} \tag{12}$$

where Z is the same as defined in Equation 10, and $I(\cdot)$ function is an indicator function which returns 1 if the input condition is true, and returns 0 otherwise. Given that Eq. 12 holds the same expectation as Eq. 10, thus, we solve Eq. 12 instead.

The EM-GD algorithm is iterative on t until $\mathcal{L}$ converge. We provide the pseudo-code as Algorithm 1. Before starting the iteration t, all parameters are initialized (recall Figure 1 top portion for an example). In each iteration t, we first perform **E-step** and **M-step** and then perform **GD-step** and **Evaluation-step**. In **E-step**, the expectation on the previously learned parameters $Q(\boldsymbol{\Theta}|\boldsymbol{\Theta}^{(t)})$ is calculated. Then, in **M-step** we get new update of parameters in $\boldsymbol{U}$: $\boldsymbol{\theta_a}^{(t+1)}$, $\boldsymbol{\theta_e}^{(t+1)}$, $\boldsymbol{\beta_a}^{(t+1)}$ and $\boldsymbol{\beta_e}^{(t+1)}$. Followed by **GD-step**, in which we get new update of parameters in $\boldsymbol{V}$: $\boldsymbol{c_a}^{(t+1)}$, $\boldsymbol{c_e}^{(t+1)}$ and $\omega^{(t+1)}$. Finally, we update the objective function $\mathcal{L}^{(t+1)}$ in **Evaluation-step**. We below describe each step in detail.

E-step refers to the expectation step, in which we calculate the conditional distribution of the ability to activity assignment probability u_i and the ability to employee assignment probability v_i by Bayes theorem, given the current estimation of parameters $\boldsymbol{\Theta}^{(t)}$.

$$\begin{aligned} T_{i,j,k}^{(t)} &= P(u_i = j, v_i = k | a_i, e_i, s_i, \boldsymbol{\Theta}^{(t)}) \\ &= \frac{\tau_{i,j,k} \phi(s_i; \lambda_{i,j,k})}{\sum_{j'=1}^{m} \sum_{k'=1}^{m} \tau_{i,j',k'} \phi(s_i; \lambda_{i,j',k'})} \end{aligned} \tag{13}$$

we calculate the conditional expectation as follows:

$$\begin{aligned} Q(\boldsymbol{\Theta}|\boldsymbol{\Theta}^{(t)}) &= \mathbb{E}_{\boldsymbol{U},\boldsymbol{V}|\boldsymbol{L},\Theta^{(t)}} [\log P(\boldsymbol{\Theta}|\boldsymbol{L}, \boldsymbol{U}, \boldsymbol{V})] \\ &= \sum_{i=1}^{n} \sum_{j=1}^{m} \sum_{k=1}^{m} T_{i,j,k}^{(t)} \log(\tau_{i,j,k} \phi(s_i; \lambda_{i,j,k})) \end{aligned} \tag{14}$$

M-step refers to the maximization step, in which we update parameters by maximizing the condition expectation $Q(\boldsymbol{\Theta}|\boldsymbol{\Theta}^{(t)})$. For $\boldsymbol{\theta_a}$ with the constraint $\sum_{j=1}^{m} \theta_{a\{j\}} = 1$,

$$\begin{aligned} \boldsymbol{\theta_a}^{(t+1)} &= \arg_{\boldsymbol{\theta_a}} \max Q(\boldsymbol{\Theta}|\boldsymbol{\Theta}^{(t)}) \\ &= \arg_{\boldsymbol{\theta_a}} \max \sum_{i=1}^{n} \sum_{j=1}^{m} \sum_{k=1}^{m} T_{i,j,k}^{(t)} (\log \theta_{a\{j\}} + \\ &\quad (\alpha - 1) \sum_{i'=1}^{m} \log(\theta_{a\{i'\}})) \end{aligned} \tag{15}$$

Then we have

$$\theta_{a\{j\}}^{(t+1)} = \frac{(\alpha - 1)n + \sum_{i=1}^{n} \sum_{k=1}^{m} T_{i,j,k}^{(t)}}{(m(\alpha - 1) + 1)n}. \tag{16}$$

In a similar way, we can update $\boldsymbol{\theta_e}$ as follows:

$$\theta_{e\{k\}}^{(t+1)} = \frac{(\alpha - 1)n + \sum_{i=1}^{n} \sum_{j=1}^{m} T_{i,j,k}^{(t)}}{(m(\alpha - 1) + 1)n}. \tag{17}$$

Next, we consider $\boldsymbol{\beta_a}$, which is a matrix in the space $\mathbf{R}^{m \times n_a}$, and n_a is the number of activities. $\beta_{a\{j,q\}}$ represents the probability of activity q requiring ability j, with the constraint $\sum_{q=1}^{n_a} \beta_{a\{j,q\}} = 1$. We can update $\boldsymbol{\beta_a}$ by solving

$$\boldsymbol{\beta}_{\boldsymbol{a}}^{(t+1)} = \arg_{\boldsymbol{\beta_a}} \max Q(\boldsymbol{\Theta}|\boldsymbol{\Theta}^{(t)}) \tag{18}$$

which means

$$\beta_{a\{j,q\}}^{(t+1)} = \frac{\sum_{i=1}^{n}\sum_{k=1}^{m} T_{i,j,k}^{(t)} I(a_i = q)}{\sum_{i=1}^{n}\sum_{j'=1}^{m}\sum_{k=1}^{m} T_{i,j',k}^{(t)} I(a_i = q)}. \tag{19}$$

Similarly, we update $\boldsymbol{\beta_e}$, which is a matrix in the space $\mathbf{R}^{m \times n_e}$ and n_e is the number of activities. $\beta_{e\{k,q\}}$ represents the probability of employee p having ability k, with the constraint $\sum_{p=1}^{n_e} \beta_{e\{k,p\}} = 1$ and

$$\beta_{e\{k,p\}}^{(t+1)} = \frac{\sum_{i=1}^{n}\sum_{j=1}^{m} T_{i,j,k}^{(t)} I(e_i = p)}{\sum_{i=1}^{n}\sum_{j=1}^{m}\sum_{k'=1}^{m} T_{i,j,k'}^{(t)} I(e_i = p)}. \tag{20}$$

GD-step estimates the parameters $\boldsymbol{c_a}$, $\boldsymbol{c_e}$ and ω by employing gradient descent (GD). After M-step, we update $\boldsymbol{c_a}$, $\boldsymbol{c_e}$ and ω by following their gradient direction with learning rate γ, which is set in LAM configuration. The following three equations present the gradient direction for parameters $\boldsymbol{c_a}$ $\boldsymbol{c_e}$ and ω respectively.

$$\begin{aligned}\frac{\partial \mathcal{L}}{\partial c_{a\,(q)}} =& \sum_{i=1}^{n}\sum_{j=1}^{m}\sum_{k=1}^{m} \\ & T_{i,j,k}^{(t)}\Big(-\frac{1}{c_q} + \frac{s_i}{c_{e\,(e_i)} c_{a\,q}^{\,2}}\big(I(j=k) + \frac{1}{\omega} I(j \neq k)\big)\Big) I(a_i = q)\end{aligned} \tag{21}$$

Here is the gradient for c_e:

$$\begin{aligned}\frac{\partial \mathcal{L}}{\partial c_{e\,(q)}} =& \sum_{i=1}^{n}\sum_{j=1}^{m}\sum_{k=1}^{m} \\ & T_{i,j,k}^{(t)}\Big(-\frac{1}{d_q} + \frac{s_i}{c_{a\,(a_i)} c_{e\,q}^{\,2}}\big(I(j=k) + \frac{1}{\omega} I(j \neq k)\big)\Big) I(e_i = q)\end{aligned} \tag{22}$$

Then the gradient for ω:

$$\frac{\partial \mathcal{L}}{\partial \omega} = \sum_{i=1}^{n}\sum_{j=1}^{m}\sum_{k=1}^{m} T_{i,j,k}^{(t)}\Big(-\frac{1}{\omega} + \frac{s_i}{c_{a\,(a_i)} c_{e\,(e_i)} \omega}\Big) I(j \neq k) \tag{23}$$

Evaluation-step is the last step in the iterative parameter learning process. In this step, we re-calculate the posterior $\mathcal{L}$ by Eq. 12 and update the objective function $\mathcal{L}^{(t+1)}$. The iteration stops when the objective function $\mathcal{L}$ meets its convergence condition.

5 LAM Prediction: Case Studies

5.1 Performance Prediction

Given a pair of employee e' and activity a' and $(e', a') \notin \boldsymbol{L}$, we can predict the service time s' of e' on a' by inferring the following conditional probability.

$$\begin{aligned}& P(s'|a', e') \\ =& Z_p \sum_{z_a=1}^{m}\sum_{z_e=1}^{m} \phi(s'; \lambda_{i,z_a,z_e}) \beta_{a\{z_a,a'\}} \theta_{a\{z_a\}} \beta_{e\{z_e,e'\}} \theta_{e\{z_e\}}\end{aligned} \tag{24}$$

where Z_p is a constant and can be inferred as the normalization factor as follows:

$$\begin{aligned} Z_p^{-1} =& \int_0^{\infty} \sum_{z_a=1}^{m}\sum_{z_e=1}^{m} \phi(s; \lambda_{i,z_a,z_e}) \beta_{a\{z_a,a'\}} \theta_{a\{z_a\}} \beta_{e\{z_e,e'\}} \theta_{e\{z_e\}} ds \\ =& \sum_{z_a=1}^{m}\sum_{z_e=1}^{m} \beta_{a\{z_a,a'\}} \theta_{a\{z_a\}} \beta_{e\{z_e,e'\}} \theta_{e\{z_e\}}\end{aligned} \tag{25}$$

Algorithm 1 LAM EM-GD Algorithm for Parameter Estimation

Input: $\boldsymbol{L}$: observation records where $\boldsymbol{L}_i = < a_i, e_i, s_i >$. Each Employee s_i is indexed from 1 to M and each activity a_i is indexed from 1 to N.
α : the prior parameter.
γ : learning rate in GD.
m : the number of latent ability mixture components.
Output: $\boldsymbol{\Theta}$.

$t = 1$
$\mathcal{L}^{(1)} = Inf$
$\mathcal{L}^{(0)} = 0$
while $\|\mathcal{L}^{(t)} - \mathcal{L}^{(t-1)}\| < \epsilon$ **do**
  //E-step
  for $i = 1$ to n **do**
    for $j = 1$ to m **do**
      for $k = 1$ to m **do**
        $T_{i,j,k}^{(t)} = \frac{\tau_{i,j,k} \phi(s_i;\lambda_{i,j,k})}{\sum_{j'=1}^{m}\sum_{k'=1}^{m} \tau_{i,j',k'} \phi(s_i;\lambda_{i,j',k'})}$
      end for
    end for
  end for
  //M-step
  for $i = 1$ to m **do**
    $\theta_{a\{i\}}^{(t+1)} = \frac{(\alpha-1)T_s^{(t)} + \sum_{j=1}^{n}\sum_{k=1}^{m} T_{j,i,k}^{(t)}}{(m(\alpha-1)+1) * T_s^{(t)}}$
    $\theta_{e\{i\}}^{(t+1)} = \frac{(\alpha-1)T_s^{(t)} + \sum_{k=1}^{n}\sum_{j=1}^{m} T_{k,j,i}^{(t)}}{(m(\alpha-1)+1) * T_s^{(t)}}$
    for $q = 1$ to n_a **do**
      $\beta_{a\{i,q\}}^{(t+1)} = \frac{\sum_{j=1}^{n}\sum_{k=1}^{m} T_{j,i,k}^{(t)} I(a_j=q)}{\sum_{j=1}^{n}\sum_{j'=1}^{m}\sum_{k=1}^{m} T_{j,j',k}^{(t)} I(a_j=q)}$
    end for
    for $q = 1$ to n_e **do**
      $\beta_{e\{i,q\}}^{(t+1)} = \frac{\sum_{k=1}^{n}\sum_{j=1}^{m} T_{k,j,i}^{(t)} I(e_k=q)}{\sum_{k=1}^{n}\sum_{j=1}^{m}\sum_{k'=1}^{m} T_{k,j,k'}^{(t)} I(e_k=q)}$
    end for
  end for
  //GD-step
  for $i = 1$ to n_a **do**
    ${c_a}_{(i)}^{(t+1)} = c_i^{(t)} + \gamma * \frac{\partial \mathcal{L}^{(t)}}{\partial {c_a}_{(i)}^{(t)}}$ //by Eq.21
  end for
  for $i = 1$ to n_e **do**
    ${c_e}_{(i)}^{(t+1)} = d_i^{(t)} + \gamma * \frac{\partial \mathcal{L}^{(t)}}{\partial {c_e}_{(i)}^{(t)}}$ //by Eq.22
  end for
  $\omega^{(t+1)} = \omega^{(t)} + \gamma * \frac{\partial \mathcal{L}^{(t)}}{\partial \omega^{(t)}}$ //by Eq.23
  //Evaluating-step
  $\mathcal{L}^{(t+1)} = P(\boldsymbol{\Theta}|\boldsymbol{L})$
  t=t+1
end while
return $\boldsymbol{\Theta}$

Note that the service time is a continuous variable. Thus, we use the probability density $\Psi(s'|a', e')$ to present the probability that employee e' can finish activity a' in s' seconds.

$$\begin{aligned} & \Psi(s'|a', e') \\ =& \int_0^{s'} P(s|a', e') ds \\ =& Z_p \sum_{z_a=1}^{m}\sum_{z_e=1}^{m} \beta_{a\{z_a,a'\}} \theta_{a\{z_a\}} \beta_{e\{z_e,e'\}} \theta_{e\{z_e\}} (1 - exp(-\lambda_{i,z_a,z_e} s'))\end{aligned} \tag{26}$$

Here Z_p is a constant to normalize the probability density.

Example 5.1 *Consider the running example in Fig. 1, we want to infer the probability of employee E0002 completing activity A0002 in 413 seconds (the average service time on E0002). We initialize* $c_a = 10, c_e = 10, \omega = 50$ *as shown in Figure 1. By Eq. 25, we get* $Z_p^{-1} = 0.0214$*. Then by Eq. 26, we calculate the probability density* $\Psi = 0.443$*. This indicates that there is 44.3% probability that employee E0002 can complete activity A0002 in 413 seconds.*

5.2 Employee Ability Prediction

Let $\boldsymbol{E}$ denote the employee-ability set and $E_{i,j}$ denote the ability score of employee i with provided ability j. Given that $\boldsymbol{\beta_e}$

represents the distribution of ability on employee and $\beta_{e,\{i,j\}}$ denotes the probability of employee i with provided sampling ability j. Thus, we can compute the ability score using the following normalized probability:

$$E_{i,j} = \frac{\beta_{e\{j,i\}}}{max(\beta_{e\{j\}})} \tag{27}$$

where $\beta_{e\{j\}}$ is the j'th row of $\boldsymbol{\beta_e}$. $max(\beta_{e\{j\}})$ is the max factor for all employees on ability j. Note that $E_{i,j}$ is the score of employee i on ability j. With this employee-ability set $\boldsymbol{E}$, we can compare any pair of employees in term of their ability. There are two possible situations. First, one employee prevails over another on all ability scores, namely, for any two employees, e_i and $e_{i'}$, if $\forall b_j \in \boldsymbol{B}, j \in \{1, ..., m\}$, we have $E_{i,j} > E_{i',j}$. It says that employee e_i did better than employee $e_{i'}$ on all of the activities that they both participated. In another situation, an employee has at least one ability score larger than that of the other.

Example 5.2 *In our running example of Fig. 1, the ability score of employee E0002 is the normalization of the weight vector* $(0.19, 0.46, 0.52)$ *by the max known weight for each ability. For B0001, B0002, B0003, the max known weight is 0.81, 0.54, 0.52 respectively. Thus, the ability score is* $(0.19, 0.46, 0.52)./(0.81, 0.54, 0.52) = (0.23, 0.87, 1)$*, which is represented by the triangle in the bottom right of Fig. 1.*

5.3 Matchup Score Estimation

A well-defined matchup score is an indicator of the degree of employee's satisfaction with respect to the activity's required ability. We define the matchup score of employee e_i to activity a_i by the probability that employee e_i satisfies all the ability requirements $\{z\}$ of activity a_i, namely

$$S_{i,j} = \sum_{z}^{m} P(z|i)P(z|j) = \sum_{z}^{m} \beta_{a\{z,i\}}\beta_{e\{z,j\}}\theta_{a\{z\}}\theta_{e\{z\}} \tag{28}$$

One way to provide better matching is to introduce the candidate activity group $G(i) = \{j|S_{i,j} > \delta\}$, where δ is a constraint constant. A large $|G(i)|$ shows that employee i is an all-around employee.

Example 5.3 *Consider the matchup score of employee E0002 on activity A0001 in Fig. 1 with* $\beta_{a,A0001} = (0.57, 0.71, 0.11)$ *and* $\beta_{e,E0002} = (0.19, 0.46, 0.52)$*. By Eq. 28 we get the matchup score of E0002, namely* $S_{2,1} = 0.19 * 0.57 * 0.34 * 0.33 + 0.46 * 0.71 * 0.33 * 0.33 + 0.52 * 0.11 * 0.33 * 0.33 = 0.00543$*. Furthermore, we can get the matchup scores of employee E0002 on all three activities, which are* $(0.0543, 0.0214, 0.0543)$*, shown in Phase 2 Matchup score* $\boldsymbol{S}$ *in Fig. 1. We infer that A0001 and A0003 are the most appropriate activities for E0002.*

6 Experiments

6.1 Datasets and Experiments Setup

The employee-activity service log datasets are collected from an operational workflow system deployed by the municipal government of Hangzhou City in China. This workflow system was deployed in seven district government departments and a central department, namely, ShangCheng (**SC**), XiaCheng (**XC**), XiHu (**XH**), Gongzhu (**GS**), BinJiang (**BJ**), ZhiJiang (**ZJ**) and HangZhou Central (**HZ**). We collect the log from May. 2013 to Apr. 2015, consisting of a total of 5,287,621 records, involving 1725 employees, 742 activities. This log collection is about the department of Land Examination and Approval from all eight departments. Table 3 shows the statistics about these log datasets. We remove the two small log datasets: **XC** and **JG**, which only involve 5 activities. In all experiments, we divide the whole log dataset into training set and testing set by 7:3 ratio. We also ensure that the pairs of employee and activity in the testing set do not appear in the training set.

All experiments are conducted on Mac OS X EI Capitan with 16GB 1867MHz DDR3 memory and 3.1GHz Intel Core i7. We implement all algorithms in MATLAB 2015b.

6.2 Evaluation Models and Metrics

To evaluate the effectiveness of our approach in terms of prediction accuracy and efficiency, we compare it with three existing representative approaches in Latent Dirichlet Allocation (LDA) [17] and Collaborative Filtering (CF) [18]. We choose LDA because both LDA and LAM use a generative statistical model. LDA creates a separate feature spaces for each observation variable and explains each type of observations by a set of unobserved features (quantity groups) to capture some latent structure of the data that is similar. LAM uses a unified latent feature space of m latent ability variables, and extracts hidden quantities as latent parameters that describe relations and interactions between observation variables and latent ability variables. We choose CF because it is the most popular approach to mine correlations between two sets of entities.

The first approach is (LDA+GLM), which fits LDA on the observations of activities and employees separately with the same number of ability groups and then fits the service time observations with a generalized linear model. The second approach is (LDA+SVR), which fits LDA on the log data first and then use the support vector regression with RBF kernel [19]. The third approach is called (AVG+CF), which pre-processes the raw log data into the service time matrix with employee and activity as rows and columns, and the average service time as the element value given an employee and an activity, and uses the collaborative filtering (CF) to predict the unknown service time.

We use the log likelihood to measure the accuracy/quality of prediction, which is defined by

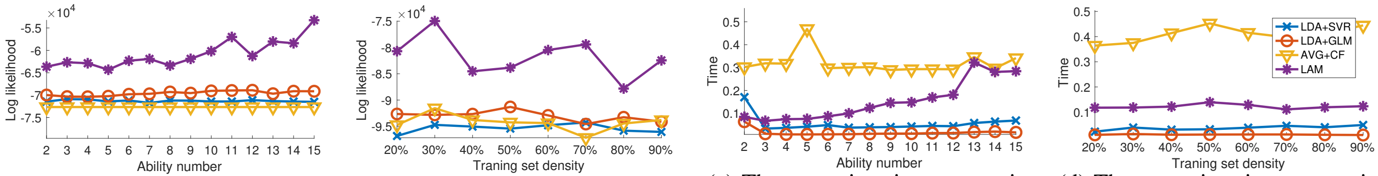
$$Lg = \sum_{i=1} log(P(s_i|a_i, e_i; \boldsymbol{\Theta})) \tag{29}$$

where $< a_i, e_i, s_i >$ is a record in the testing set.

Given an employee-activity pair, a_i and e_i, LAM outputs a probability distribution on the values of service time s_i as shown in Figure 1, whereas LDA+SVR, LDA+GLM and AVG+CF estimate the service time s_i using their own method as the prediction result. For comparison, we apply an exponential distribution whose expectation is the predicted service s_i as the output distribution for LDA+SVR, LDA+GLM and AVG+CF respectively. We set the Dirichlet parameter α to 5.0 for all four models. The model that gives higher probability to the unseen employee-activity pair better captures the hidden interactions between observation variables and latent ability variables. We measure and compare the effectiveness of the four models in terms of accuracy by log likelihood and efficiency by execution time.

6.3 Employee Performance Prediction

We evaluate and compare the effectiveness of the four models for employee performance prediction in both accuracy and efficiency, first on a combined collection of the six log datasets and then on



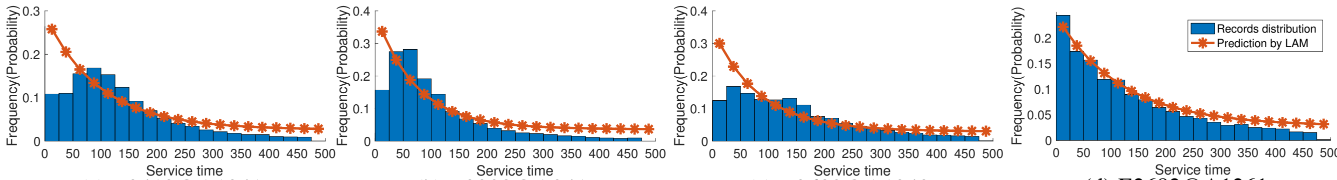
(a) The log likelihood comparison with varying m

(b) The log likelihood comparison with varying data density

(c) The execution time comparison with varying m

(d) The execution time comparison with varying data density.

Fig. 3: The log likelihood and prediction execution time on combined log collection.



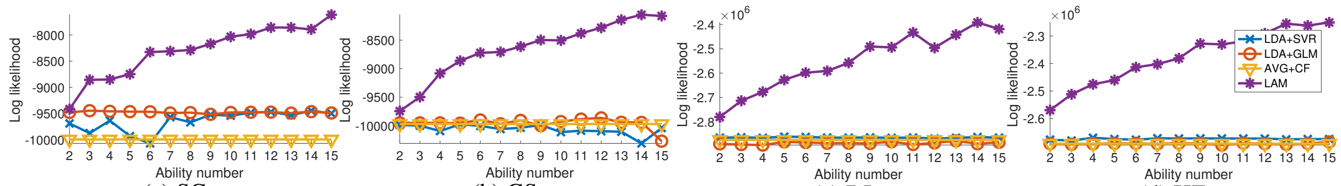
(a) E2419@A1241

(b) E2308@A941

(c) E2682@A1242

(d) E2682@A1261

Fig. 4: The comparison between the statistical result and predicted distribution of service time.



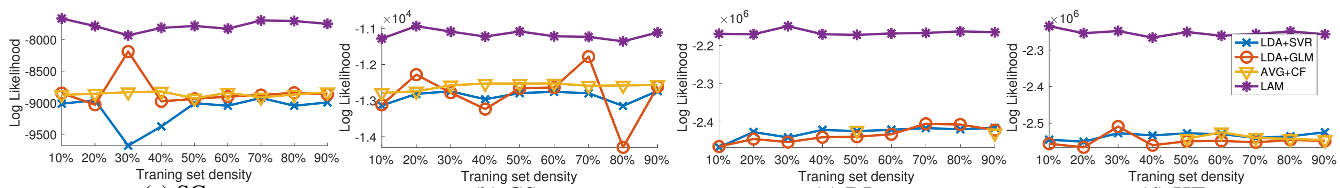
(a) SC

(b) GS

(c) BJ

(d) HZ

Fig. 5: The log likelihood comparison on 4 log datasets by varying m, the number of ability groups



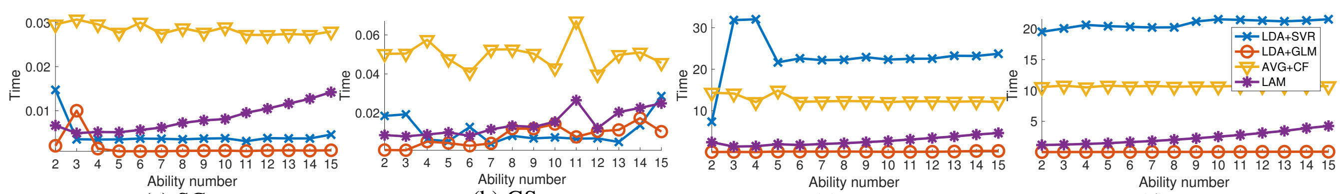
(a) SC

(b) GS

(c) BJ

(d) HZ

Fig. 6: The log likelihood comparison on 4 log datasets with varying dataset density.



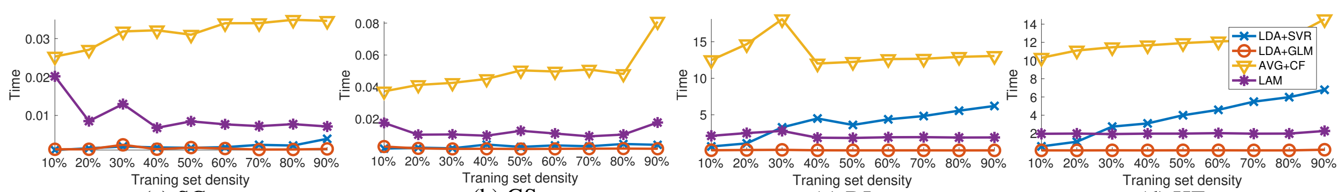
(a) SC

(b) GS

(c) BJ

(d) HZ

Fig. 7: The execution time comparison on 4 log datasets with varying m, the number of ability groups.



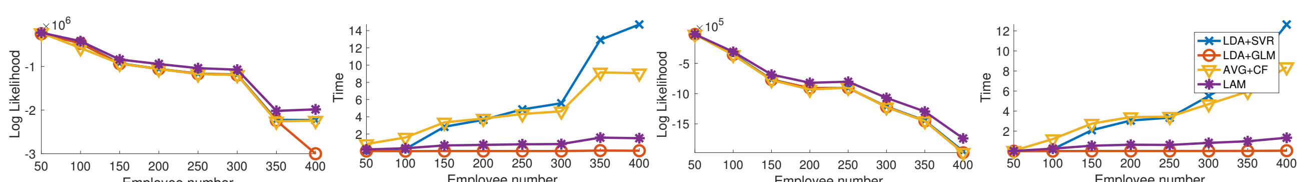
(a) SC

(b) GS

(c) BJ

(d) HZ

Fig. 8: The execution time comparison on 4 log datasets with varying dataset density.



(a) Execution time of different contexts from BJ

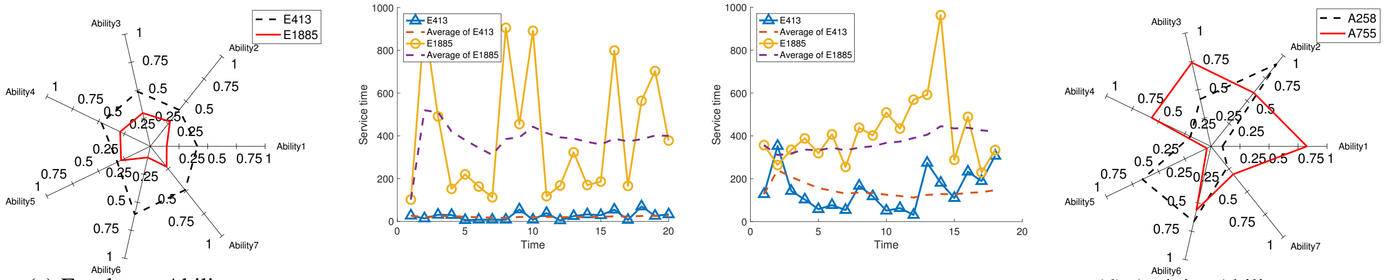
(b) Log likelihood of different contexts from BJ

(c) Execution time of different contexts from HZ

(d) Log likelihood of different contexts from HZ

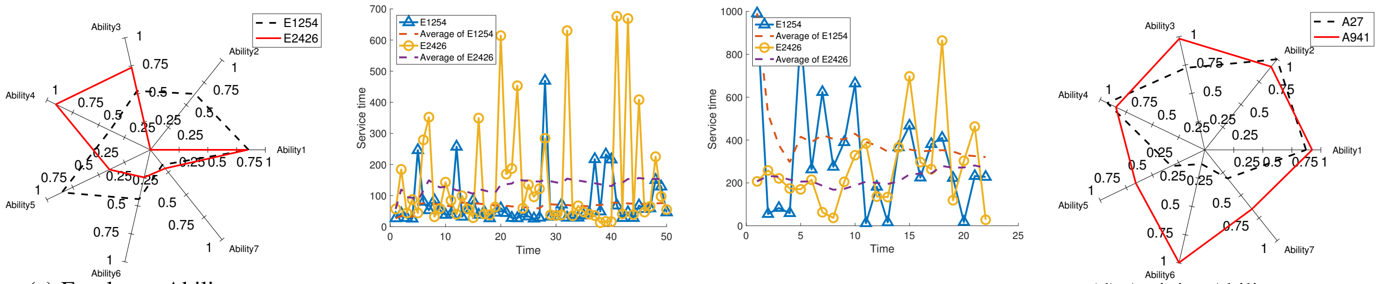
Fig. 9: The performance comparison among different contexts.

each of the six log datasets. Fig. 3 shows the comparison results on the combined log collection. LAM outperforms all other three models in quality. To vary the density of the training set, we remove some data in training set to make it sparse. Concretely, the training set density of $x\%$ refers to $(1 - x\%)$ of training dataset was randomly removed. When we vary the density of the



(a) Employee Ability scores (b) E413, E1885@A775 (c) E413, E1885@A258 (d) Activity Ability scores

Fig. 10: The comparison of employees E0415, E1885 on activity A775, A258.



(a) Employee Ability scores (b) E1254, E2426@A941 (c) E1245, E2426@A27 (d) Activity Ability scores

Fig. 11: The comparison employee E1254, E2426 on activity A941, A27.

training dataset, we use the default setting of $m = 7$ and when we vary m, we use the default density setting of 100%. Fig. 3 (a) and (b) show the log-likelihood measures of all four models by varying m and the density of the training set respectively. In both cases, LAM performs significantly better than the other three models in log-likelihood performance. LDA+GLM slightly better than LDA+SVR and AVG+CF. Figure 3 (c) and (d) show the execution time comparison by varying m and the density of the training set respectively. In both case, LAM consistently outperforms AVG+CF with the shortest time to finish. LDA+GLM is faster than other models while worse in quality. Also we observe that LAM slows down when m reaches 13 or higher. It urges us to choose a trade-off $m = 7$.

In order to explain the reason that LAM outperforms all other three models, We conduct the next set of experiments to further illustrate the high accuracy of LAM prediction performance by comparing the distribution of actual service time in original log data and the prediction by LAM on four employee-activity pairs. Figure 4 shows the results. We observe that the quality of LAM prediction on the probability distribution of service time closely approximates the actual distribution in the original log dataset.

In the next sets of experiments, we compare the four approaches in terms of the accuracy and efficiency on six independent datasets, namely **SC**, **XH**, **GS**, **BJ**, **ZJ** and **HZ**. Due to the space limit, we omit **XH** and **ZJ** in this paper as they are similar to **GS**. Fig. 5 measures the log likelihood by varying m. We observe that (1) LAM has the highest log likelihood as m increases for all six log datasets; (2) LDA+SVR, LDA+GLM and AVG+CF have a similar log likelihood independent of m; and (3) the log likelihood increases with m. Given that a larger m requires more time spent in training phase, thus we can trade-off accuracy and efficiency by finding local optimal setting of m. Fig. 5 shows that all six datasets exhibit a stable log likelihood when m is around 7 or 8. Thus, the default setting of m is 7.

We also measure the log likelihood on six datasets by varying the density percentage of training dataset. Figure 6 shows the results. We observe that LAM consistently delivers high accuracy even when the density of training dataset is as low as 10%. The performance of LAM is dis-sensitive to the data density which means it does not facing the cold-start challenge. Also, we can see that in big dataset, BJ and HZ in (c) and (d), the accuracy gap between LAM with other models is quite big. One reason that the other three models perform poorly for BJ and HZ datasets is the low ratio of their recorded employee-activity pairs in the log over all possible pairs.

Recall Table 3, the column "Percentage of Recorded Activity and Employee Pair" shows that for both in BJ and ZJ, the ratio of the employee-activity pairs in the log dataset over is the smallest (0.23%) among all datasets. Such low ratio indicates the serious sparseness in the log dataset, resulting in log likelihood for LDA+SVR, LDA+GLM, AVG+CF, worse than LAM.

Next, we measure the execution time of all four approaches on the six datasets by varying m. Figure 7 shows the result. In small datasets, SC and GS, in (a) and (b), LDA+SVR and LDA+GLM take the least execution time. In big datasets, BJ and HZ, see (c) and (d), LAM is more faster than LDA+SVR. Figure 8 shows the running time with varying training set density. LAM and AVG+CF have the shortest execution time consistently for big datasets, e.g. BJ and HZ, when the dataset density is 30% or higher.

Finally, we measure the execution time and accuracy on different contexts. Two biggest datasets, BJ and HZ are used in this experiment. We removed the original dataset by involving only a few employee number. For example, an 100-employee context of BJ means 100 employees from BJ are randomly extracted. Also, the activities retained are those participated by the randomly selected 100 employees. The training set and testing set are randomly divided by 7:3. The experimental results are reported in Fig. 9. In (a) and (c), we can see that the execution time of all models is increasing as the context size (number of employees per context) increases. This is because the larger context means more data records to process. In (b) and (d), we observe that the accuracy is decreasing for all methods as the context size increases. This is because more employees are involved in the model training and testing, thus more diversity, and the accuracy measure Lg takes the sum of log likelihood of all records. Regarding the execution time, LAM grows much slower than LDA+SVR and AVG+CF as the context size increases, and LDA+GLM shows slightly shorter execution time than LAM as the context size grows at the cost of lower in accuracy than LAM. This set of experiments further shows that LAM is more effective than the existing methods, especially in large and complex contexts.

6.4 Employee Ability Comparison

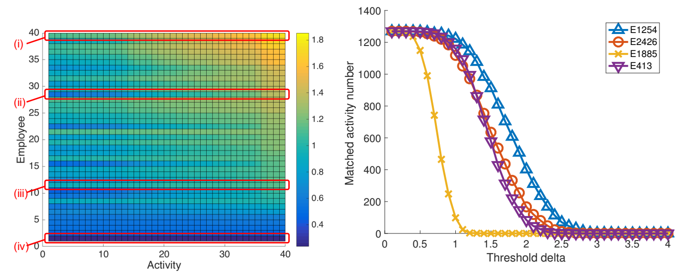
The employee ability comparison should consider two typical scenarios: (1) An employee has had higher score in all m ability groups than another. Thus, for the set of common activities that they both have participated, the former employee should have better performance than the latter for all activities. (2) For any two employees, each of the two has had higher score in at least one of the m ability groups. In this case, among the set of common activities, we can always find one activity that the former employee does better and find another activity that the latter employee performs better.

For each employee, we obtain the ability scores for all m ability components, denote by $\boldsymbol{E}$, which can be obtained by Eq. 27. Figure 10 evaluates the effectiveness of LAM for employee ability comparison by considering the first scenario, i.e., the ability comparison on two employees: E413 and E1885. Figure 10 (a) shows that employee E413 has higher ability score than employee E1885 for all m ability groups ($m = 7$). The black color dashed polygon shows the 7 ability scores of employee E413 with respect to the 7 ability groups, which are around 0.5, much higher than the 7 ability scores of employee E1885, whose ability scores are lower than 0.25 as shown in red color solid polygon. Next, we sample two activities A775 and A258 from the log dataset, in which both employees E413 and E1885 have participated for a number of times. Figure 10 (b) illustrates the service time comparison of employees E413 and E1885 on activity A775. We observe that employee E413 takes significantly less time and thus is more effective than employee E1885. This result is consistent with the employee ability score comparison in Figure 10 (a). Fig. 10 (c) shows the ability comparison of the same pair of employees on activity A258. Again we observe that employee E413 takes shorter service time than employee E1885, consistent with the fact that employee E413 has higher ability scores than employee E1885 on activity A258. Fig. 10 (d) shows the activity required ability comparison. We observe that activities A775 and A258 have different ability scores for $m = 7$ ability groups.

Figure 11 illustrates the second scenario. From Figure 11 (a), employee E1254 has higher score in ability 2 and ability 5 but lower score in ability 3 and 4, compared to employee E2426. We sample two activities, A941 and A27, from the log dataset, in which both employees E1254 and E2426 have participated for several times and have different service times. Fig. 11 (b) and (c) show the service time on activity A941 and activity A27 respectively. We observe that employee E1254 has shorter service time on activity A941 but longer service time on activity A27, comparing with employee E2426. This is consistent with the employee ability scores shown in Fig. 11 (a) and the activity required ability scores shown in Fig. 11 (d).

6.5 Employee-Activity Matchup Prediction

We want to evaluate the effectiveness of our model in predicting how well the employee's provided ability set matches up with the activity's required ability set for any given pair of employee and activity. Recall Section 5, we have defined the matchup score



(a) The grid color graph of matching score on first 40 activities and 40 employees.

(b) The candidate activity number reducing diagram of E1254, E2426, E1885 and E413 with the growth of threshold δ.

Fig. 12: Matching score and candidate activity

$S_{i,j}$ in Equation 28 for employee i and activity j, which uses $\boldsymbol{\beta}_a, \boldsymbol{\beta}_e, \boldsymbol{\theta}_a, \boldsymbol{\theta}_e$.

Figure 12 (a) shows the matching score $S_{i,j}$ on 40 activities and 40 employees. The color of the grid represents the matching score and the x-axis represents activity id and y-axis represents employee id. The lighter color denotes the higher matching score. We arrange the 40 employees by their highest provided ability scores on the 40 activities. Then we arrange the 40 activities by the highest required ability score on the 40 employees. We observe that for most of the employees, the color varies with different activities. Thus, we sort both the set of employees and the set of activities such that the right-top portion of Fig. 12 (a) is light color and left-bottom portion is dark. We obtain the following intuitions.

First, Some employees have either consistently high matchup scores on many activities, or have very different matchup scores on different activities, such as those marked (i) and (ii) in Fig. 12 (a). Specifically, matchup scores of employees in the group marked by (i) are varying significantly with respect to the 40 activities. It means that employees in the region marked (i) are relatively more flexible and can work effectively for most of the activities. In comparison, most matchup scores of employees in the region marked by (ii) are relatively lower compared to those in group (i) for most of the 40 activities. It means that the employees in the group (ii) is suitable for only a few activities out of the 40 activities in comparison.

Second, a few employees have very similar scores on most of the activities, such as employees in group (iii) and group (iv) in Fig. 12 (a), with employees in group (iv) have the darkest color and thus lowest matchup scores on all 40 activities. It means that employees in group (iv) performs poorly in comparison to the others in the set of 40 employees.

Recall Section 5.3, we introduce the concept of candidate activity group $G(i)$ with threshold δ: $G(i) = \{j | S_{i,j} \geq \delta\}$. Given an employee i, $G(j)$ finds the set of activities with the matchup score $S_{i,j}$ larger than the system-defined threshold δ. In the next set of experiments, we vary the threshold δ and measure the size of $G(i)$, the number of activities with matchup score higher than the threshold, on the four employees used in experimental case studies: E1254, E2426, E1885 and E413. Figure 12 (b) shows the results. We make several interesting observations. First, as δ increases, different employees show different decreasing rate with respect to the size of their candidate activity group. Also this deceasing rate is tightly related to their ability scores. Recall that employee E1885 has the lowest average score, which is below 0.25, compared to the others, especially employee E413. Thus, the curve of employee E1885 is sharply declined, indicating that the

TABLE 5: First 3 match up activities

EmployeeID	Name	TopActivity1 (ID)	Score	TopActivity2 (ID)	Score	TopActivity3 (ID)	Score	Shortest Service Time Activity
E413	X. C.	Final Review on Affordable Housing Department (A621)	1.870	Applying Agreement from Internet (A561)	1.868	Acceptance Checking (A259)	1.861	A775
E1885	Q. W.	Applying Agreement from Internet (A561)	1.637	Acceptance Checking (A259)	1.637	Acceptance Checking on Consulting File (A941)	1.625	A258
E1254	M. Z.	Acceptance on Consulting File (A941)	1.585	Acceptance Checking (A259)	1.584	Applying Agreement from Internet (A561)	1.583	A941
E2426	Y. Y.	Acceptance Checking (A259)	1.619	Applying Agreement from Internet (A561)	1.618	Acceptance Checking on Consulting File (A941)	1.611	A941

size of his/her candidate activity group reduces the fastest, as the threshold δ increases. It approaches 0 when δ is set to 1.5, which implies that no activity is suitable when $\delta \geq 1.5$. In comparison, other three employees can still matchup much more activities (400 or higher).

Table 5 lists the most appropriate three activities for the four employees in our experimental case studies. We rank the activities for each employee by the matchup score and show the top-3 for each employee. We can see that for employee E1254 and E2426, the shortest-time activity hit in the top-3 results. It means that our matchup score is really close to reality. While for employee E413 and E1885, the shortest-time activities do not appear in top-3 results. By checking the data, we found that there is no any records about the employees on our top-3 activities. Therefore we can recommend these three activities to them.

7 Related Works

Workforce analytics has received attentions from three research areas: (a) human efficiency, (b) operator allocation and (c) e-Government.

Human efficiency. This line of research focuses on finding out the factors that influence human efficiency with some well-known work. Hockey [8] studied the noise, which produces a narrowing of attention. [20] emphasizes the importance of delegation in enhancing the work efficiency. Paarlberg [9] introduced the impact of customer orientation on government employee performance. Elena et. al. [5] developed the schematic scientifically grounded criteria to evaluate the effectiveness of the employees. [21] discussed the Employee Participation in Profit and Ownership. These existing approaches, however, lack of either quantitative analysis or experimental proof. They are pre-defined and subjective in nature. In comparison, our LAM approach iteratively learn the latent factors qualitatively and the results of our model is provable on practical datasets.

Operator Allocation. This line of research centers on the problem of operator allocation or assignment to employees. The ultimate goal of finding out the optimal operator allocation plan to improve efficiency and increase productivity of both employees and organization as a collection of employees. The early work [22] solves the problem using single criterion by mixed integer programming (MIP). Then, [12] consider the operator allocation on multi-dimensions by introducing the skill category, each denotes one of possible skill combinations and different work requires different operation skills. [11] proposed seven manually defined ability criteria: quality, planning, initiative, teamwork, communication and external factors. [13] reviews the literature on the multiple ability criteria decision in recent two decades. All these existing efforts to date define and model workers' ability skills manually and the ability score is defined subjectively. In contrast, LAM infers the ability set $\boldsymbol{B}$ from the real employee-activity log data automatically and provides employee ability scores statistically.

E-Government. The research on e-Government [23][24] employs information technology to public administrations. [25] shows that efficiency improvement is one of the major challenges for e-Government. Virile [26] studied the e-Government plan in Italy and emphasized the attention on efficiency. Liang [27] proposed the models and selection strategies to promote the efficiency of e-Government by Cloud computing techniques. This line of works emphasize the importance of efficiency in e-Government. However, their focus is primarily on the efficiency of computing instead of employees. The techniques proposed ignore the delays brought by the human factors in measuring workforce efficiency. Our model predicts the distribution of service time from the employee-activity log by extracting the latent ability variables to unify the ability set required by activities and the ability set provided by employees in performing their assigned activities.

8 Conclusion

We have presented a generative probabilistic inference and prediction framework, Latent Ability Model (LAM). Using LAM, we map employee and activity to a unified latent ability space, and for a vector of abstract abilities, we represent an employee by his/her ability vector and an activity by its requirement on the set of abilities. Then we employ LAM based unsupervised learning on this employ-activity latent space initialized by service time. LAM model enhances the existing unsupervised prediction approaches from two novel perspectives: (1) Based the linear relationships and observations from the EA service_time datasets, LAM explores the latent relationships between employee and activities through service time and extract interesting structural patterns from the random and skewed data. (2) LAM derives the prediction model based on both semantic and linear correlation features and the hidden and complex latent correlation features extracted from the real datasets provided by the Hangzhou City government in China. Our extensive experimental results show that LAM approach significantly outperforms existing representative prediction methods in both accuracy and efficiency. We are currently working on extending LAM to handle more complex datasets with more attributes in workforce data analysis. Furthermore, to incorporate new data records, we need to retrain LAM model. We are also interested in exploring incremental approaches to construct and update the LAM model.

References

[1] TechTarget-EGuide, "An hr analytics essential guide collection: Measuring employee engagement and maximizing human resource roles," *pro.techtarget.com*, 2016.

[2] A. Mojsilović and D. Connors, “Workforce analytics for the services economy,” in *Handbook of service science*. Springer, 2010, pp. 437–460.

[3] J. Hota and D. Ghosh, “Workforce analytics approach: An emerging trend of workforce management,” *Workforce management*, pp. 167–179, 2013.

[4] J. R. Campbell and K. N. Kuttner, “Macroeconomic effects of employment reallocation,” in *Carnegie-Rochester Conference Series on Public Policy*, vol. 44. Elsevier, 1996, pp. 87–116.

[5] E. I. Danilina and D. V. Gorelov, “Development of the municipal employee’s performance efficiency and effectiveness indicators on the basis of functional and process approaches,” *American Journal of Economics and Business Administration*, vol. 6, no. 4, p. 133, 2014.

[6] S. J. Davis and J. Haltiwanger, “Gross job creation, gross job destruction, and employment reallocation,” *The Quarterly Journal of Economics*, vol. 107, no. 3, pp. 819–863, 1992.

[7] H. A. Eiselt and V. Marianov, “Employee positioning and workload allocation,” *Computers & operations research*, vol. 35, no. 2, pp. 513–524, 2008.

[8] G. Hockey, “Effects of noise on human efficiency and some individual differences,” *Journal of Sound and Vibration*, vol. 20, no. 3, pp. 299–304, 1972.

[9] L. E. Paarlberg, “The impact of customer orientation on government employee performance,” *International Public Management Journal*, vol. 10, no. 2, pp. 201–231, 2007.

[10] L. M. Saari and T. A. Judge, “Employee attitudes and job satisfaction,” *Human resource management*, vol. 43, no. 4, pp. 395–407, 2004.

[11] R. Islam and S. bin Mohd Rasad, “Employee performance evaluation by the ahp: A case study,” *Asia Pacific Management Review*, vol. 11, no. 3, 2006.

[12] Y. Kuo and T. Yang, “Optimization of mixed-skill multi-line operator allocation problem,” *Computers & Industrial Engineering*, vol. 53, no. 3, pp. 386–393, 2007.

[13] A. Mardani, A. Jusoh, and E. K. Zavadskas, “Fuzzy multiple criteria decision-making techniques and applications–two decades review from 1994 to 2014,” *Expert Systems with Applications*, vol. 42, no. 8, pp. 4126–4148, 2015.

[14] C. G. Şen and G. Çınar, “Evaluation and pre-allocation of operators with multiple skills: A combined fuzzy ahp and max–min approach,” *Expert Systems with Applications*, vol. 37, no. 3, pp. 2043–2053, 2010.

[15] Y. Koren and R. Bell, “Advances in collaborative filtering,” in *Recommender systems handbook*. Springer, 2015, pp. 77–118.

[16] J. B. Schafer, D. Frankowski, J. Herlocker, and S. Sen, “Collaborative filtering recommender systems,” in *The adaptive web*. Springer, 2007, pp. 291–324.

[17] D. M. Blei, A. Y. Ng, and M. I. Jordan, “Latent dirichlet allocation,” *the Journal of machine Learning research*, vol. 3, pp. 993–1022, 2003.

[18] F. Ricci, L. Rokach, and B. Shapira, *Introduction to recommender systems handbook*. Springer, 2011.

[19] C.-C. Chang and C.-J. Lin, “Libsvm: A library for support vector machines,” *ACM Transactions on Intelligent Systems and Technology (TIST)*, vol. 2, no. 3, p. 27, 2011.

[20] S. Kamal and J. Raza, “Enhancing work efficiency through skillful delegation,” *Interdisciplinary Journal of Contemporary Research in Business*, vol. 3, no. 2, p. 241, 2011.

[21] M. Kozłowski, “Employee participation in profit and ownership–impact on work efficiency,” *Comparative Economic Research*, vol. 16, no. 1, pp. 71–86, 2013.

[22] S. Vembu and G. Srinivasan, “Heuristics for operator allocation and sequencing in product-line-cells with manually operated machines,” *Computers & industrial engineering*, vol. 32, no. 2, pp. 265–279, 1997.

[23] V. Peristeras, G. Mentzas, K. A. Tarabanis, and A. Abecker, “Transforming e-government and e-participation through it,” *IEEE Intelligent Systems*, vol. 24, no. 5, pp. 14–19, 2009.

[24] A. Tripathi and B. Parihar, “E-governance challenges and cloud benefits,” in *2011 IEEE International Conference on Computer Science and Automation Engineering*, vol. 1. IEEE, 2011, pp. 351–354.

[25] C. Moulin and M. L. Sbodio, “Improving the accessibility and efficiency of e-government processes,” in *9th IEEE International Conference on Cognitive Informatics (ICCI’10)*. IEEE, 2010, pp. 603–610.

[26] F. Virili, “The italian e-government action plan: from gaining efficiency to rethinking government,” in *Proceedings of the 12th International Workshop on Database and Expert Systems Applications*. IEEE, 2001, pp. 329–333.

[27] J. Liang, “Government cloud: enhancing efficiency of e-government and providing better public services,” in *Proceedings of the 2012 International Joint Conference on Service Sciences*. IEEE, 2012, pp. 261–265.

Zhiling Luo is a research assistant professor in Computer Science at Zhejiang University, China. He received his B.S. and Ph.D. degree in Computer Science from Zhejiang University in 2012 and 2017, respectively. He was the visiting scholar of Georgia Institute of Technology, US, in 2016. His research interests include service computing, machine learning and data mining.

Ling Liu is a Professor in the School of Computer Science at Georgia Institute of Technology. She directs the research programs in Distributed Data Intensive Systems Lab (DiSL). Prof. Liu is an elected IEEE Fellow, a recipient of IEEE Computer Society Technical Achievement Award in 2012, and a recipient of the best paper award from a dozen of top venues, including ICDCS 2003, WWW 2004, 2005 Pat Goldberg Memorial Best Paper Award, IEEE Cloud 2012, IEEE ICWS 2013, ACM/IEEE CCGrid 2015, IEEE Symposium of Big Data 2016. In addition to service as general chair and PC chairs of numerous IEEE and ACM conferences in data engineering, Prof. Liu has served on editorial board of over a dozen international journals. Prof. Liu’s current research is primarily sponsored by NSF, IBM and Intel.

Jianwei Yin is currently a professor in the College of Computer Science, Zhejiang University, China. He received his Ph.D. in Computer Science from Zhejiang University in 2001. He is the visiting scholar of Georgia Institute of Technology, US, in 2008. His research interests include service computing, cloud computing and information integration. Currently Prof. Yin is the AE of IEEE Transactions on Service Computing.

Ying Li received the B.S., M.S. and Ph.D. degrees in computer science from Zhejiang University, China, in 1994, 1997 and 2000, respectively. He is currently an associate professor with the College of Computer Science, Zhejiang University. He is currently leading some research projects supported by National Natural Science Foundation of China and National High-tech RD Program of China (863 Program). His research interests include service computing, business process management and compiler.

Zhaohui Wu received the B.S. and Ph.D. degrees in computer science from Zhejiang University, Hangzhou, China, in 1988 and 1993, respectively. He is currently a Professor with the College of Computer Science, Zhejiang University. His research interests include distributed artificial intelligence, semantic grid, and pervasive computing. Dr. Wu is a Standing Council Member of the China Computer Federation.

APPENDIX: DEDUCTION AND PROOF

In this section, we show some important deductions in LAM. At first, recall that we calculate the posterior to get train parameters. According to the MAP, the posterior Eq. 12 is derived by Bayesian rules. The key idea is to represent the posterior by the prior distributions $Q(\boldsymbol{\theta_a}|\boldsymbol{\alpha})$, $Q(\boldsymbol{\theta_e}|\boldsymbol{\alpha})$, and conditional distributions $P(a_i|z_a, \boldsymbol{\beta_a})$, $P(e_i|z_e, \boldsymbol{\beta_e})$, $P(z_a|\boldsymbol{\theta_a})$ and $P(z_e|\boldsymbol{\theta_e}))$. Note that the prior distributions and conditional distributions have been well defined in generative model. Therefore, we replace them by their definitions. In this way, we get a simple presentation of the posterior.

$$\begin{aligned}
\mathcal{L} =& P(\boldsymbol{\Theta}|\boldsymbol{L}) \\
=& \prod_{i=1}^{n} P(\boldsymbol{\Theta}|a_i, e_i, s_i) \\
=& \prod_{i=1}^{n} P(\boldsymbol{\theta_a}, \boldsymbol{\theta_e}, \boldsymbol{\beta_a}, \boldsymbol{\beta_e}, \boldsymbol{c_a}, \boldsymbol{c_e}, \omega|a_i, e_i, s_i) \\
=& Z \prod_{i=1}^{n} P(a_i, e_i, s_i|\boldsymbol{\theta_a}, \boldsymbol{\theta_e}, \boldsymbol{\beta_a}, \boldsymbol{\beta_e}, \boldsymbol{c_a}, \boldsymbol{c_e}, \omega) Q(\boldsymbol{\theta_a}|\boldsymbol{\alpha}) Q(\boldsymbol{\theta_e}|\boldsymbol{\alpha}) \\
=& Z \prod_{i=1}^{n} (\sum_{z_a=1}^{m} \sum_{z_e=1}^{m} P(s_i|\boldsymbol{\theta_a}, \boldsymbol{\theta_e}, z_a, z_e, \boldsymbol{c_a}, \boldsymbol{c_e}, \omega) \\
& P(a_i|z_a, \boldsymbol{\beta_a}) P(e_i|z_e, \boldsymbol{\beta_e}) P(z_a|\boldsymbol{\theta_a}) \\
& P(z_e|\boldsymbol{\theta_e})) Q(\boldsymbol{\theta_a}|\boldsymbol{\alpha}) Q(\boldsymbol{\theta_e}|\boldsymbol{\alpha}) \\
=& Z \prod_{i=1}^{n} \sum_{j=1}^{m} \sum_{k=1}^{m} \phi(s_i; \lambda_{i,j,k}) \beta_{a\{j,a_i\}} \\
& \beta_{e\{k,e_i\}} \theta_{a\{j\}} \theta_{e\{k\}} \frac{1}{B(\boldsymbol{\alpha})} \prod_{i'=1}^{m} (\theta_{a\{i'\}} \theta_{e\{i'\}})^{\alpha-1} \\
=& Z \prod_{i=1}^{n} \sum_{j=1}^{m} \sum_{k=1}^{m} \tau_{i,j,k} \phi(s_i; \lambda_{i,j,k})
\end{aligned} \tag{30}$$

Besides posterior, another important formula is the conditional expectation, namely Eq. 14, which is the objective to maximized in EM algorithm. Recall that, in EM, we need to re-calculate the parameters $\boldsymbol{\Theta}$ according to the parameters in last iteration $\boldsymbol{\Theta}^{(t)}$. In following deduction, we introduce two temporal variable $\boldsymbol{U}$ and $\boldsymbol{V}$ for simplification.

$$\begin{aligned}
& Q(\boldsymbol{\Theta}|\boldsymbol{\Theta}^{(t)}) \\
=& \mathbb{E}_{\boldsymbol{U},\boldsymbol{V}|\boldsymbol{L},\Theta^{(t)}} [\log P(\boldsymbol{\Theta}|\boldsymbol{L}, \boldsymbol{U}, \boldsymbol{V})] \\
=& \mathbb{E}_{\boldsymbol{U},\boldsymbol{V}|\boldsymbol{L},\Theta^{(t)}} [\log \prod_{i=1}^{n} P(\boldsymbol{\Theta}|a_i, e_i, s_i, u_i, v_i)] \\
=& \mathbb{E}_{\boldsymbol{U},\boldsymbol{V}|\boldsymbol{L},\Theta^{(t)}} [\sum_{i=1}^{n} \log P(\boldsymbol{\Theta}|a_i, e_i, s_i, u_i, v_i)] \\
=& \sum_{i=1}^{n} \mathbb{E}_{\boldsymbol{U},\boldsymbol{V}|\boldsymbol{L},\Theta^{(t)}} [\log P(\boldsymbol{\Theta}|a_i, e_i, s_i, u_i, v_i)] \\
=& \sum_{i=1}^{n} \sum_{j=1}^{m} \sum_{k=1}^{m} P(u_i = j, v_i = k|a_i, e_i, s_i; \boldsymbol{\Theta}^{(t)}) \\
& \log P(\boldsymbol{\Theta}|a_i, e_i, s_i, u_i, v_i) \\
=& \sum_{i=1}^{n} \sum_{j=1}^{m} \sum_{k=1}^{m} T_{i,j,k}^{(t)} \log(\tau_{i,j,k} \phi(s_i; \lambda_{i,j,k}))
\end{aligned} \tag{31}$$

Given the learned features $\boldsymbol{\Theta}$, we'd like to predict the probability $P(s'|a', e')$ namely an employee e' can finish an activity a' on service time s'. Note that $P(s'|a', e')$ means the probability of service time equaling to s'. In practice, we use Eq. 26 to calculate the probability that service time not larger than s'. Here, we present the basic deduction on $P(s'|a', e')$ based on the Bayesian rule.

$$\begin{aligned}
& P(s'|a', e') \\
=& \sum_{z_a=1}^{m} \sum_{z_e=1}^{m} P(s'|z_a, z_e, a', e') P(z_a|a') P(z_e|e') \\
=& Z_p \sum_{z_a=1}^{m} \sum_{z_e=1}^{m} P(s'|z_a, z_e, a', e') P(a'|z_a) Q(z_a) P(e'|z_e) Q(z_e) \\
=& Z_p \sum_{z_a=1}^{m} \sum_{z_e=1}^{m} \phi(s'; \lambda_{i,z_a,z_e}) \beta_{a\{z_a,a'\}} \theta_{a\{z_a\}} \beta_{e\{z_e,e'\}} \theta_{e\{z_e\}}
\end{aligned} \tag{32}$$

Recall that we mention that we can use the features, namely the employees' abilities and the activities' requirement to estimate how goodness an employee fitting an activity. Here we introduce the formalization of the matchup score $S_{i,j}$, on activity i with employee j. The intuitively understanding is that we calculate the probability that employee j have all abilities $z = \{1...m\}$ that required by i.

$$\begin{aligned}
& S_{i,j} \\
=& \sum_{z}^{m} P(z|i) P(z|j) \\
=& \sum_{z}^{m} P(i|z) P(z|\theta_a) P(j|z) P(z|\theta_e) \\
=& Z_z \sum_{z}^{m} \beta_{a\{z,i\}} \beta_{e\{z,j\}} \theta_{a\{z\}} \theta_{e\{z\}}
\end{aligned} \tag{33}$$

Here Z_z is a constant for normalization. We can see that $S_{i,j}$ is larger if the probability that employee j have all abilities $z = \{1...m\}$ that required by i, is larger. In other words, the larger matchup score $S_{i,j}$ denotes that j is fitting activity i.