AISTAR: An Intelligent System for Online IT Ticket Automation Recommendation

Qing Wang, Chunqiu Zeng, S. S. Iyengar, Tao Li
School of Computing and Information Sciences
Florida International University
Miami, FL, USA
Email: {qwang028,czeng001,iyengar}@cs.fiu.edu

Larisa Shwartz
Cognitive Service Foundations
IBM T.J. Watson Research Center
Yorktown Heights, NY, USA
Email: lshwart@us.ibm.com

Genady Ya. Grabarnik
Dept. Math & Computer Science
St. John's University
Queens, NY, USA
Email: grabarng@stjohns.edu

***Abstract*—An efficient delivery of IT services for increasingly complex IT environments demands an intelligent automated solution for resolving existing and potential issues. An automation recommender system, promptly suggesting the most proper scripted resolution to an arriving IT incident ticket, would play a significant role in IT automation services. Hence, developing a comprehensive framework supporting becomes imperative for continuous improvement of automation recommendation.**

In this paper, we first identify the challenges of IT services followed by a discussion on *AISTAR* (an intelligent system for online IT ticket automation recommendation) designed and developed to provide them. Specifically, we define and formalize automation recommendation procedure as a multi-armed bandit problem with *dependent* arms, which is capable of achieving the optimal tradeoff between exploitation of the system for the best automation recommendation and exploration of automation execution information for future recommendation. Two novel multi-armed bandit models are proposed and integrated to handle the aforementioned challenges in IT automation services. Empirical studies on a large ticket dataset from IBM Global Services demonstrate both the effectiveness and efficiency of our intelligent integrated system. *AISTAR* is earmarked for Cognitive Event Automation for IBM Service delivery.

***Keywords*-Multi-armed bandit model; Interactive recommender system; Cold-start problem; Cognitive IT service management**

I. Introduction

In today's economic climate, business enterprises constantly explore innovative ways in expanding their outreach and gaining competitive advantages in the marketplace. As IT service providers, they are expected to focus on innovation and assisting customers in their core business areas. To minimize both the time and efforts of IT experts when fixing operational issues, enterprise IT automation services (ITAS) [1] has been introduced into IT service management (ITSM) as an engine for automated corrective actions and automated closure of incident records. Incident management is one of the most critical processes in ITSM as it restores normal operations by resolving issues affecting business services. It usually relies on automated monitoring for problem detection, followed by problem diagnosis and resolution by ITAS and human engineers. The growing complexity of IT environment urgently demands a dynamic automation solution driven by artificial intelligence.

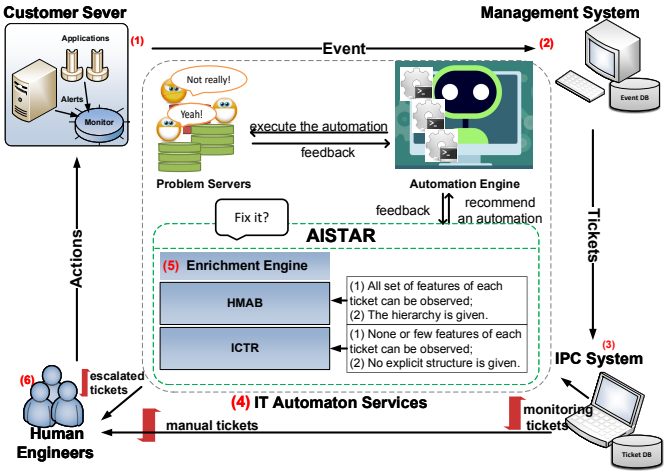


Figure 1: The overview of ITSM workflow.

Figure 1 illustrates the ITSM workflow augmented with our proposed intelligent integrated system, called AISTAR. Typically, the entire workflow consists of six steps. (1) Anomalies are detected by the monitoring system. When the anomaly persists beyond a predefined duration, the system generates an event for further inspection. (2) Events from an entire IT environment are consolidated in an enterprise event management system, which determines whether or not to create an alert and, subsequently, an incident ticket (sometimes referred to as a monitoring ticket). (3) Tickets are collected by an Incident, Problem, and Change (IPC) system. (4) Based on the symptom of the monitoring ticket, ITAS executes an automation (i.e., a scripted resolution). In case it doesn't fix the issue, the ticket is escalated to a human engineer. (5) An enrichment engine uses historical data for continuous enhancement of ITAS. (6) Manually created and escalated tickets are forwarded to human engineers for a labor-intensive problem determination, diagnosis, and resolution.

Figure 2 shows an example of two different ticket problems that were detected and auto-ticketed by the monitoring system, and subsequently closed successfully by the same (process cpu spike) automation. The summary and monitoring class (sometimes called an alert key) of each ticket provide an initial symptom description used by automation

978-1-5386-5035-6/18/$31.00 ©2018 IEEE

services to identify relevant existing automations or lack thereof. When the problem is resolved by the recommended automation, the value of "AUTORESOVLED" is set to be nonzero. In order to improve the efficiency of the recommending strategies in automation engine (AE), it is essential to infer a mapping function between the symptoms and the corresponding scripted resolutions. This is the initial motivation for our study. Based on preliminary studies [2], [3], we have identified three key challenges as follows.

Ticket Problem 1

ALERT_KEY	xxx_cpusum_xuxc_aix		AUTOMATON_NAME		Process CPU Spike Automation		
OPEN_DTTM	CLIENT_ID	HOSTNAME	ORIGINAL SEVERITY	OSTYPE	COMPONENT	SUBCOMPONENT	AUTO RESOVLED
1456383421000	90	XXX	4	AIX	UNIX	UNKNOWN	1
TICKET SUMMARY	XXX CPU Utilization is very high, workloads affected			TICKET RESOLUTION	Alert in question has recovered, hence closing the ticket.		

Ticket Problem 2

ALERT_KEY	xxx_prccpu_rlzc_std		AUTOMATON_NAME		Process CPU Spike Automation		
OPEN_DTTM	CLIENT_ID	HOSTNAME	ORIGINAL SEVERITY	OSTYPE	COMPONENT	SUBCOMPONENT	AUTO RESOVLED
1454900281000	52	XXX	4	UNKNOWN	LINUX	Process	1
TICKET SUMMARY	XXX Process using high cpu (97.30%) nsrdmpix			TICKET RESOLUTION	Alert in question has recovered, hence closing the ticket.		

Figure 2: Two different ticket problems in ITSM.

Challenge 1: How do we properly solve the well-known cold-start problem and adaptively optimize the recommending strategies of the enterprise automation engine using interactive feedback in ITAS?

Most recommender systems suffer from the cold-start problem, since every system could encounter a significant number of users/items that are completely new to the system with no historical records at all. It makes the systems ineffective unless additional information about users/items is collected [4]. This is a crucial problem for AE since an unknown issue would require a significant human effort. AE (see Figure 1) can automatically take action based on the context of a ticket problem and receive the execution feedback (e.g., success or failure) from the problem server. The current strategies of AE have not yet exploited the execution feedback for continuous improvement in an interactive mode. Based on the aforementioned discussion, we focus on the solution to an online learning problem of recommending an appropriate automation that instantly makes use of the up-to-date feedback by adjusting the recommendation strategies. This can be naturally modeled as a multi-armed bandit problem, the widely applied model to various interactive recommender systems [5], [6], [7]. Note that the model also allows us to address the cold-start problem by achieving the optimal balances between *exploration* and *exploitation*.

Challenge 2: How do we efficiently improve the performance of recommendation using the automation hierarchy of ITAS?

In ITSM, domain experts would define a taxonomy [8] to categorize IT problems (see Figure 3). Correspondingly, automations can also be organized into an underlying hierarchical structure of categories. For example, a problem is auto-ticketed due to a failure of DB2 database. The root cause may be database deadlock, high usage, etc. Intuitively, if it is initially treated as a database problem, the automation engine will take less time to fix it by ignoring those automations in other categories (e.g., application and OS). To utilize the taxonomy, we formulate it as a bandit problem with dependent arms organized hierarchically, which could match each arm's feature space from a coarse level first, and then be refined to the next lower level according to the taxonomy.

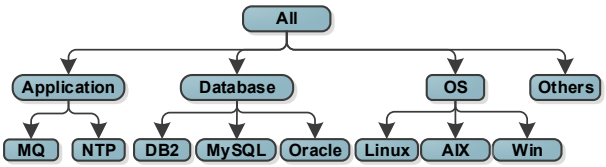


Figure 3: An example of taxonomy in IT tickets.

Challenge 3: How do we effectively recommend a proper automation with no explicit hierarchical information and, in the worse case, with no contextual information of the incident ticket in ITAS?

The reality of IT environments is such that not all of automations are properly set in a hierarchical structure due to the lack of sufficient information and some may fall into the "Others" category (see Figure 3). Furthermore, as a result of the imperfection of log information, a large number of tickets is logged with an error code only and no detailed symptoms. It lays difficulties in inferring a proper automation. By observing the rich historical data, we find different ticket problems (see Figure 2) that can be resolved by the same automation, while a single ticket problem may be resolved by different automations. Facing the challenge, we propose an interactive collaborative topic regression model capable of learning hidden features for ticket problems and automations, while automatically identifying automation dependencies in the form of clusters.

The remainder of this paper is organized as follows. In Section II, we provide mathematical formalizations of the aforementioned problems. An integrated solution is given in Section III. Section IV describes the comparative experiments and two empirical case studies conducted over a real ticket dataset. The related work is shown in Section V. Finally, we conclude the paper in Section VI.

II. Problem Formulation

A. Online Recommendation of IT Automations

Let $\mathcal{A} = \{a^{(1)}, ..., a^{(N)}\}$ denote a set of automations (i.e., scripted resolutions) feasible in IT automation system. Every time a ticket is reported, AE selects a proper automation according to contextual information (i.e., the ticket symptom) and recommends it as a possible resolution. Specifically, the contextual information of a reported ticket at time t is represented as a feature vector $\mathbf{x}_t \in \mathcal{X}$, where $\mathcal{X}$ denotes the d-dimensional feature space. Every recommended automation $a^{(i)} \in \mathcal{A}$ at time t has a corresponding feedback indicating whether or not the ticket has been successfully resolved.

The aforementioned process can be naturally modeled as a contextual bandit problem, where automations are constantly recommended and the recommendation model is adaptively updated by the feedback (e.g., success or failure) collected over time. Generally, a contextual bandit problem involves a series of decisions over a finite but possibly unknown time horizon T. In our formulation, each arm corresponds to an automation, pulling an arm indicates recommending an automation, and the reward received after pulling an arm is the feedback after recommending an automation.

In the contextual bandit setting, at each time $t = [1, \ldots, T]$, a policy π is defined as a function and used to select an automation $\pi(\mathbf{x}_t) \in \mathcal{A}$, according to the contextual vector $\mathbf{x}_t$ of the current ticket. Let $r_{\mathbf{x}_t,n}$ denote the reward for recommending an automation $a^{(n)}$ at time t, whose value is drawn from an unknown distribution determined by the context $\mathbf{x}_t$. The total reward received by the policy π after T iterations is

$$R_\pi = \sum_{t=1}^{T} r_{\mathbf{x}_t,\pi(\mathbf{x}_t)}. \quad (1)$$

The optimal policy π^* is defined as the one with maximum accumulated expected reward after T iterations,

$$\pi^* = \arg\max_\pi E(R_\pi) = \arg\max_\pi \sum_{t=1}^{T} E(r_{\mathbf{x}_t,\pi(\mathbf{x}_t)}|t). \quad (2)$$

Our goal is to identify the optimal policy for maximizing the total reward. Before selecting the optimal automation at time t, a policy π is updated to refine a model for reward prediction of each automation according to the historical observations $\mathbf{S}_t = \{(\mathbf{x}_i, n(i), r_{\mathbf{x}_i,n(i)})|i = [1, \ldots, t-1]\}$. The reward prediction helps to ensure that the policy π includes decisions to increase the total reward. The reward $r_{\mathbf{x}_t,n}$ is typically modeled as a linear combination of the feature vector $\mathbf{x}_t$ given as follows:

$$r_{\mathbf{x}_t,n} = \mathbf{x}_t^T \theta_n + \xi_n, \quad (3)$$

where θ_n is a d-dimensional coefficient vector and ξ_n denotes an observation noise, a zero-mean Gaussian noise with variance σ_n^2, i.e., $\xi_n \sim \mathcal{N}(0, \sigma_n^2)$. Then,

$$r_{\mathbf{x}_t,n} \sim \mathcal{N}(\mathbf{x}_t^T \theta_n, \sigma_n^2), \quad (4)$$

and our objective function in Equation 2 can be reformulated as:

$$\pi^* = \arg\max_\pi \sum_{t=1}^{T} E_{\theta_{\pi(\mathbf{x}_t)}}(\mathbf{x}_t^T \theta_{\pi(\mathbf{x}_t)}|t). \quad (5)$$

To address the aforementioned problem, contextual bandit algorithms have been proposed and extensively adopted in real applications to balance the tradeoff between exploration and exploitation for arm selection, including ϵ-greedy, Thompson sampling [9], LinUCB [5], etc. However, most of them do not take the dependencies among arms into account. In the IT environment, automations are organized hierarchically. In the following section, we will introduce the proposed model leveraging both the explicit and implicit arm dependencies in the bandit setting for ITAS.

B. Leverage Automation Hierarchy for Online Interactive Recommendation

In general, these automations can be classified using a pre-defined taxonomy of ITAS. It allows us to reformulate the problem as a bandit model with dependent arms in the form of hierarchies [10].

Let $\mathcal{H}$ denote the taxonomy, which contains a set of nodes (i.e., arms) organized in a tree-structured hierarchy. Given a node $a^{(i)} \in \mathcal{H}$, $pa(a^{(i)})$ and $ch(a^{(i)})$ are used to represent the parent and children sets, respectively. If $pa(a^{(i)}) = \emptyset$, node $a^{(i)}$ is the root node. If $ch(a^{(i)}) = \emptyset$, $a^{(i)}$ is a leaf node indicating an automation. Otherwise, $a^{(i)}$ is a category node when $ch(a^{(i)}) \neq \emptyset$. Since the goal is to recommend an automation to a ticket problem and only a leaf node of $\mathcal{H}$ is an automation, the recommendation process cannot be completed until a leaf node is selected at each time t. Therefore, the multi-armed bandit problem of IT automation recommendation is reduced to a selection of a path in $\mathcal{H}$ from the root to a leaf node, and multiple arms along the path are sequentially selected based on the contextual vector $\mathbf{x}_t$ at time t.

Let $pth(a^{(i)})$ be a set of nodes along the path from the root node to the leaf node $a^{(i)}$ in $\mathcal{H}$. Assume that $\pi_{\mathcal{H}(\mathbf{x}_t|t)}$ is the path selected by the policy π in light of the contextual information $\mathbf{x}_t$ at time t. For every arm selection policy π we have:

Property 1: Given the contextual information $\mathbf{x}_t$ at time t, if a policy π selects a node $a^{(i)}$ in the hierarchy $\mathcal{H}$ and receives positive feedback (i.e., success), the policy π receives positive feedback as well by selecting the nodes along the path $pth(a^{(i)})$.

Let $r_{\mathbf{x}_t,\pi_{\mathcal{H}(\mathbf{x}_t|t)}}$ denote the reward obtained by the policy π after selecting the multiple arms along the path $\pi_{\mathcal{H}(\mathbf{x}_t|t)}$ at time t. The reward is computed as follows:

$$r_{\mathbf{x}_t,\pi_{\mathcal{H}(\mathbf{x}_t|t)}} = \sum_{a^{(i)} \in \pi_{\mathcal{H}(\mathbf{x}_t|t)}, ch(a^{(i)}) \neq \emptyset} r_{\mathbf{x}_t,\pi(\mathbf{x}_t|ch(a^{(i)}))}, \quad (6)$$

where $\pi(\mathbf{x}_t|ch(a^{(i)}))$ represents the arm selected from the children of $a^{(i)}$, given the contextual information $\mathbf{x}_t$. After T iterations, the total reward received by the policy π is:

$$R_{\pi_\mathcal{H}} = \sum_{t=1}^{T} r_{\mathbf{x}_t,\pi_{\mathcal{H}(\mathbf{x}_t|t)}}. \quad (7)$$

The optimal policy π^* with respect to $\mathcal{H}$ is determined by

$$\pi^* = \arg\max_\pi E(R_{\pi_\mathcal{H}}) = \arg\max_\pi \sum_{t=1}^{T} E(r_{\mathbf{x}_t}, \pi_{\mathcal{H}(\mathbf{x}_t|t)}). \quad (8)$$

The reward prediction for each arm is inferred by Equation 4, and then the optimal policy can be equivalently

determined by

$$\pi^* = \arg\max_{\pi} \sum_{t=1}^{T} \sum_{\substack{a^{(i)} \in \pi_{\mathcal{H}(\mathbf{x}_t|t)}, \\ ch(a^{(i)}) \neq \emptyset}} E_{\theta_{\pi(\mathbf{x}_t|ch(a^{(i)}))}}(\mathbf{x}_t^T \theta_{\pi(\mathbf{x}_t|ch(a^{(i)}))}|t). \tag{9}$$

We incorporate both Thompson sampling and LinUCB algorithms into our models in Section III-A. Since our models explicitly make use of the hierarchies defined by domain experts, it makes our proposed algorithms converge much faster by exploring automations hierarchically.

C. Learning Hidden Features for Online Interactive Recommendation

In ITSM, IT tickets contain their alert keys (i.e., ticket problems) but often lack useful symptom descriptions. This makes inference of corresponding automations more challenging. In this section, we learn the latent features from historical ticket resolution records for both ticket problems and automations in an online setting.

Assume that there are M ticket problems and N automations in ITAS. Let $\mathbf{R} = \{r_{m,n}\} \in \mathcal{R}^{M \times N}$ be a rating matrix, where the rating $r_{m,n}$ indicates whether ticket problem m can be resolved by automation n. At each time $t = [1, \ldots, T]$, the system, according to the observed rating history, recommends an automation $n(t)$ to an incoming ticket problem m. The feedback (i.e., success or failure) $r_{m,n(t)}$ is collected over time to update the recommendation model. Let $\mathbb{S}(t) = \{(n(1), r_{m,n(1)}), \ldots, (n(t-1), r_{m,n(t-1)})\}$ be the available information at time t collected by the system for the target ticket problem m. Our goal is to identify an optimal policy that maximizes the total ratings. Herein a policy π makes a decision to select a proper automation $\pi(\mathbb{S}(t)) \in \mathcal{A}$ according to the history information $\mathbb{S}(t)$.

Using the latent factor model [11], [12], the rating $r_{m,n}$ can be estimated by a product of ticket problem and automation latent feature vectors $\mathbf{p}_m \in \mathcal{R}^K$ and $\mathbf{q}_n \in \mathcal{R}^K$ in a shared low K-dimension space. Additionally, we introduce an observation noise ξ_n, a zero-mean Gaussian noise with variance σ_n^2 (i.e., $\xi_n \sim \mathcal{N}(0, \sigma_n^2)$) to the rating prediction function. The derived rating prediction is as follows:

$$r_{m,n} = \mathbf{p}_m^\intercal \mathbf{q}_n + \xi_n. \tag{10}$$

The latent features $\{\mathbf{p}_m\}$ and $\{\mathbf{q}_n\}$ can be learned by minimizing the prediction error for all observed ratings in $\mathbf{R}$, while each unobserved rating value can be estimated using Equation (10) with corresponding latent features learned by bayesian matrix factorization technique [12]. In this setting, Equation 5 is reformulated as:

$$\pi^* = \arg\max_{\pi} \sum_{t=1}^{T} \mathbb{E}_{\mathbf{p}_m, \mathbf{q}_{\pi(\mathbb{S}(t))}}(\mathbf{p}_m^\intercal \mathbf{q}_{\pi(\mathbb{S}(t))}|t). \tag{11}$$

Thus, our goal is to optimize the objective function in Equation 11.

To use the Thompson sampling and UCB strategies, we need to learn the distributions for both the random variable $\mathbf{p}_m$ and $\mathbf{q}_n$ from Bayesian learning perspective, so that the latent feature vectors can be easily sampled and the feedback at every time can be reasonably integrated [6], [13]. Furthermore, since the dependencies among some automations are not properly indicated with an explicit hierarchical structure in practice, the challenge herein lies in how to sequentially infer the latent dependencies as well as improve the accuracy of IT ticket troubleshooting while simultaneously providing the feedback over time. In the following section, we describe a generative Bayesian model to address those issues.

D. Identify Latent Dependencies among Automations

Utilizing the idea of topic modeling, a generative model, Interactive Collaborative Topic Regression (ICTR), is proposed aiming at learning the hidden features from the latent topic space and discovering the dependencies among automations according to their corresponding topic distribution [14].

In ICTR model, we consider an automation n as a word and a ticket problem m as a document. All the automations used to resolve a ticket problem indicate the possible resolutions to the ticket as analogous to the scenario in topic modeling where the words contained in a document imply its latent topics. Specifically, K is the number of latent aspects (i.e., topics or clusters). The k-th component of $\mathbf{p}_m$ suggests the preference of a ticket problem over the k-th aspect of automations, while the k-th component value of $\mathbf{q}_n$ represents the probability of automation n to belong to the k-th cluster. The rating $r_{m,n}$ is estimated by the inner product of $\mathbf{p}_m$ and $\mathbf{q}_n$. The corresponding probabilistic graphical model of rating is presented in Figure 4.

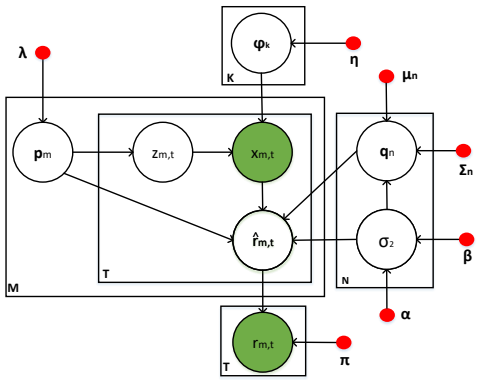


Figure 4: The graphical representation for ICTR model.

In the graphical model above, we assume $\mathbf{p}_m$ follows a Dirichlet prior distribution with a predefined hyper parameter λ:

$$\mathbf{p}_m|\lambda \sim Dir(\lambda). \tag{12}$$

As presented in Equation 10, σ_n^2 is denoted as the variance of the noise for rating prediction, and we assume σ_n^2 is drawn

from the Inverse Gamma (abbr., $\mathcal{IG}$) distribution shown as follows:

$$p(\sigma_n^2|\alpha,\beta) = \mathcal{IG}(\alpha,\beta), \tag{13}$$

where α and β are the predefined hyper parameters for the Inverse Gamma distribution.

We assume that given σ_n^2, $\mathbf{q}_n$ is generated by a Gaussian distribution (see Equation 14). Let $\varphi_k \in \mathcal{R}^N$ be the automation distribution on the topic k. We assume that φ_k is subject to Dirichlet distribution:

$$\mathbf{q}_n|\mu_{\mathbf{q}},\mathbf{\Sigma}_{\mathbf{q}},\sigma_n^2 \sim \mathcal{N}(\mu_{\mathbf{q}},\sigma_n^2\mathbf{\Sigma}_{\mathbf{q}}), \quad \varphi_k|\eta \sim Dir(\eta), \tag{14}$$

where $\mu_{\mathbf{q}}, \mathbf{\Sigma}_{\mathbf{q}}$ and $\eta \in \mathcal{R}^N$ are hyper parameters.

When a ticket problem m arrives to interact with the recommender system at time t, one of K topics, denoted as $z_{m,t}$, is first selected according to the ticket problem's latent vector $\mathbf{p}_m$, indicating that the automations in the topic $z_{m,t}$ are more likely to resolve the ticket problem m at this moment. The topic $z_{m,t}$ follows a multinomial distribution governed by $\mathbf{p}_m$ (see Equation 15).

Without loss of generality, we assume $z_{m,t} = k$, the automation distribution on the topic k (i.e., φ_k) used for generating the automation $x_{m,t}$ recommended to ticket problem m at time t. $x_{m,t}$ follows a multinomial distribution ruled by φ_k, i.e.,

$$z_{m,t}|\mathbf{p}_m \sim Mult(\mathbf{p}_m), \quad x_{m,t}|\varphi_k \sim Mult(\varphi_k). \tag{15}$$

The automation n is assumed to be selected by ticket problem m at time t (i.e., $x_{m,t} = n$) where the latent vector corresponding to automation n is $\mathbf{q}_n$. Let $\hat{r}_{m,t}$ be the predicted rating, given by ticket problem m at time t. The predicted rating $\hat{r}_{m,t}$ can be inferred by

$$\hat{r}_{m,t} \sim \mathcal{N}(\mathbf{p}_m^\intercal \mathbf{q}_n, \sigma_n^2). \tag{16}$$

The ratings of all the automations can be estimated by Equation 16. The policy π selects an automation and recommends it to ticket problem m, considering the trade-off between exploitation and exploration. The actual rating $r_{m,t}$ is collected after executing the recommended automation. The objective of the model is to maximize the expected accumulative ratings in the long run as described in Equation 11.

III. METHODOLOGY AND SOLUTION

A. Solution with Automation Hierarchy

In this section, we propose the Hierarchical Multi-Armed Bandit (HMAB) algorithms for exploiting the dependencies among hierarchically organized arms. As presented in Equation 4, the reward $r_{\mathbf{x}_t,n}$ depends on random variable $\mathbf{x}_t$, θ_n and σ_n^2. We assume θ_n and σ_n^2 follow a Normal Inverse Gamma (abbr., $\mathcal{NIG}$) conjugate prior distribution. The parameter σ_n^2 is drawn from the Inverse Gamma distribution:

$$p(\sigma_n^2|\alpha_n,\beta_n) \sim \mathcal{IG}(\alpha_n,\beta_n), \tag{17}$$

where α_n and β_n are the predefined hyper parameters for the $\mathcal{IG}$ distribution. Given σ_n^2, the coefficient vector θ_n is generated by a Gaussian prior distribution with the hyper parameter μ_{θ_n} and $\mathbf{\Sigma}_{\theta_n}$:

$$p(\theta_n|\mu_{\theta_n},\mathbf{\Sigma}_{\theta_n},\sigma_n^2) \sim \mathcal{N}(\mu_{\theta_n},\sigma_n^2\mathbf{\Sigma}_{\theta_n}). \tag{18}$$

At each time t, a policy π will select a path $\pi_{\mathcal{H}(\mathbf{x}_t|t)}$ from $\mathcal{H}$ according to the context $\mathbf{x}_t$. Assuming $a^{(p)} \in \pi_{\mathcal{H}(\mathbf{x}_t|t)}$ is the leaf node (i.e., an automaton), then we have $pth(a^{(p)}) = \pi_{\mathcal{H}(\mathbf{x}_t|t)}$. After recommending the automation $a^{(p)}$, a reward $r_{p,t}$ is obtained. Since the reward $r_{p,t}$ is shared by all the arms along the path $pth(a^{(p)})$, a set of triples $F = \{(\mathbf{x}_t, a^{(n)}, r_{\mathbf{x}_t,n}) | a^{(n)} \in pth(a^{(n)}), r_{\mathbf{x}_t,n} = r_{p,t}\}$ is acquired. A new sequence $S_{\pi,t}$ is generated by incorporating the triple set F into $S_{\pi,t-1}$. The posterior distribution for every $a^{(n)} \in pth(a^{(n)})$ needs to be updated with the new feedback sequence $S_{\pi,t}$. The posterior distribution of θ_n and σ_n^2 given $S_{\pi,t}$ is a $\mathcal{NIG}$ distribution with the hyper parameters μ_{θ_n}, $\mathbf{\Sigma}_{\theta_n}$, α_n and β_n. The hyper parameters at time t, i.e., $\mathbf{\Sigma}'_{\theta_n}$, μ'_{θ_n}, α'_n and β'_n are updated based on their values at time $t-1$:

$$\mathbf{\Sigma}'_{\theta_n} = (\mathbf{\Sigma}_{\theta_n}^{-1} + \mathbf{x}_t\mathbf{x}_t^T)^{-1}, \quad \mu'_{\theta_n} = \mathbf{\Sigma}'_{\theta_n}(\mathbf{\Sigma}_{\theta_n}^{-1}\mu_{\theta_n} + \mathbf{x}_t r_{\mathbf{x}_t,n})$$
$$\alpha'_n = \alpha_n + \frac{1}{2}, \ \beta'_n = \beta_n + \frac{1}{2}[r_{\mathbf{x}_t,n}^2 + \mu_{\theta_n}^T\mathbf{\Sigma}_{\theta_n}^{-1}\mu_{\theta_n} - \mu_{\theta_n}'^T\mathbf{\Sigma}_{\theta_n}'^{-1}\mu'_{\theta_n}]. \tag{19}$$

Algorithm 1 The algorithms for HMAB model

procedure MAIN($\mathcal{H}, \pi, \lambda$) ▷ Main entry, π is the policy.
  for $t \leftarrow 1, T$ **do**
    Initialize parameters of $a^{(m)} \in \mathcal{H}$ to α_m, β_m, $\mathbf{\Sigma}_{\theta_m} = \mathbf{I}_d$, $\mu_{\theta_m} = \mathbf{0}_{d\times 1}$
    Get contextual vector $\mathbf{x}_t \in \mathcal{X}$
    for each path P of $\mathcal{H}$ **do**
      Compute the reward of P using Equation 6, by calling EVAL($\mathbf{x}_t, a^{(n)}, \pi$) for each arm $a^{(n)} \in P$
    end for
    Choose the path P^* with maximum reward
    Recommend the automation $a^{(*)}$ (leaf node of P^*)
    Receive reward $r_{\mathbf{x}_t,*}$ by recommending the automation $a^{(*)}$
    UPDATE($\mathbf{x}_t, P^*, r_{\mathbf{x}_t,*}, \pi$)
  end for
end procedure

procedure EVAL($\mathbf{x}_t$, $a^{(n)}$, π) ▷ Get a score for $a^{(n)}$, given $\mathbf{x}_t$.
  if π is TS **then**
    Sample σ_n^2 according to Equation 17
    Sample θ_n according to Equation 18
    return $\hat{r}_{\mathbf{x}_t,n} = \mathbf{x}_t^T\theta_n$
  end if
  if π is LinUCB **then**
    return $\hat{r}_{\mathbf{x}_t,n} = \mathbf{x}_t^T\mu_{\theta_n} + \frac{\lambda}{\sigma_n}\sqrt{\mathbf{x}_t^T\mathbf{\Sigma}_{\theta_n}^{-1}\mathbf{x}_t}$
  end if
end procedure

procedure UPDATE($\mathbf{x}_t, P, r_t, \pi$) ▷ update the inference. ▷ P is the path in $\mathcal{H}$, r_t is the reward.
  for each automation $a^{(n)} \in P$ **do**
    Update $\mathbf{\Sigma}'_{\theta_n}$, μ'_{θ_n}, α'_n and β'_n using Equation 19
  end for
end procedure

Note that the posterior distribution of θ_n and σ_n^2 at time t is considered as the prior distribution of time $t+1$. We propose HMAB algorithms presented in Algorithm 1 with different sampling strategies (Thompson sampling and UCB) as HMAB-TS($\mathcal{H}$, α, β) and HMAB-LinUCB($\mathcal{H}$, λ).

Online inference of our hierarchical bandit problem starts with MAIN procedure. As a ticket $\mathbf{x}_t$ arrives at time t, the EVAL procedure computes a score for each automation of different levels. In each level, the automation with the highest score is selected to be recommended. After receiving reward, the new feedback is used to update the HMAB algorithms by the UPDATE procedure.

B. Solution with ICTR Model

In this section, we present the methodology for online inferences of ICTR model. The posterior distribution inference involves five random variables, i.e., $\mathbf{p}_m$, $z_{m,t}$, φ_k, $\mathbf{q}_n$, and σ_n^2. According to the graphical model in Figure 4, these five random variables belong to two categories: parameter random variable and latent state random variable. φ_k, $\mathbf{p}_m$, $\mathbf{q}_n$, and σ_n^2 are parameter random variables since they are assumed to be fixed but unknown, and their values do not change with time. Instead, $z_{m,t}$ is referred to as a latent state random variable since it is not observable and its value is time dependent. After recommending the automation $n(t)$, where $n(t) = x_{m,t}$ according to Equation 15 at time t, a rating is observed as $r_{m,t}$. Thus, $x_{m,t}$ and $r_{m,t}$ are referred to as observed random variables.

Our goal is to infer both latent parameter variables and latent state random variables to sequentially fit the observed data at time $t-1$, and predict the ratings for automation selection with respect to the incoming ticket problem at time t. However, since the inference of our model cannot be conducted by a simple closed-form solution, we adopt the sequential sampling-based inference strategy that is widely used in sequential Monte Carlo sampling [15], particle filtering [16], and particle learning [17], [18] thus learning the distribution of both the parameter and the state random variables. Specifically, a particle learning method that allows both state filtering and sequential parameter learning simultaneously is a perfect solution to our proposed model inference. In particle learning, each particle corresponds to a sample for modeling inference status information. At each time stamp, the particles are re-sampled according to their fitness to the current observable data. Then, the re-sampled particles are propagated to new particles and obtain the status information for the next time stamp.

We interpret particle learning as follows. A particle for predicting the reward $\hat{r}_{m,t}$ is a container that maintains the current status information for both ticket problem m and automation $x_{m,t}$. The status information comprises random variables such as $\mathbf{p}_m$, σ_n^2, φ_k, $\mathbf{q}_n$, and $z_{m,t}$, as well as the hyper parameters of their corresponding distributions, such as λ, α, β, η, $\mu_{\mathbf{q}}$ and $\mathbf{\Sigma}_{\mathbf{q}}$.

1) Re-sample Particles with Weights: At time $t-1$, a fixed-size set of particles is maintained for the rating prediction for each automation $n(t-1)$ given the ticket problem m. We denote the particle set at time $t-1$ as $\mathcal{P}_{m,n(t-1)}$ and assume the number of particles in $\mathcal{P}_{m,n(t-1)}$ is B. Let $\mathcal{P}^{(i)}_{m,n(t-1)}$ be the i^{th} particles given both ticket problem m and automation $n(t-1)$ at time $t-1$, where $1 \le i \le B$. Each particle $\mathcal{P}^{(i)}_{m,n(t-1)}$ has a weight, denoted as $\rho^{(i)}$, indicating its fitness for the new observed data at time t. Note that $\sum_{i=1}^{B} \rho^{(i)} = 1$. The fitness of each particle $\mathcal{P}^{(i)}_{m,n(t-1)}$ is defined as the likelihood of the observed data $x_{m,t}$ and $r_{m,t}$. Therefore,

$$\rho^{(i)} \propto p(x_{m,t}, r_{m,t} | \mathcal{P}^{(i)}_{m,n(t-1)}). \tag{20}$$

Further, $\hat{r}_{m,t}$ is the predicted value of $r_{m,t}$. The distribution of $\hat{r}_{m,t}$, determined by $\mathbf{p}_m$, $\mathbf{q}_n$, $z_{m,t}$, φ_k , and σ_n^2, described in Section II-D.

We compute $\rho^{(i)}$ as proportional to the density value given $\hat{r}_{m,t} = r_{m,t}$ and $x_{m,t} = n$. Thus, we obtain

$$\rho^{(i)} \propto \sum_{z_{m,t}=1}^{K} \{\mathcal{N}(r_{m,t} \,|\, \mathbf{p}_m^{\intercal}\mathbf{q}_n, \sigma_n^2) p(z_{m,t} = k, x_{m,t} = n | \mathcal{P}^{(i)}_{m,n(t-1)})\},$$

After further deriving

$$p(z_{m,t} = k, x_{m,t} = n \,|\, \mathcal{P}^{(i)}_{m,n(t-1)}) = E(\mathbf{p}_{m,k}|\lambda)\ E(\varphi_{k,n}|\eta),$$

we have:

$$\rho^{(i)} \propto \sum_{z_{m,t}=1}^{K} \{\mathcal{N}(\mathbf{r}_{m,t} \,|\, \mathbf{p}_m^{\intercal}\mathbf{q}_n, \sigma_n^2)\ E(\mathbf{p}_{m,k}|\lambda)\ E(\varphi_{k,n}|\eta)\}, \tag{21}$$

where $E(\mathbf{p}_{m,k}|\lambda) = \frac{\lambda_k}{\sum_{k=1}^{K}\lambda_k}$ and $E(\varphi_{k,n}|\eta) = \frac{\eta_{k,n}}{\sum_{n=1}^{N}\eta_{k,n}}$ represent the conditional expectations of $\mathbf{p}_{m,k}$ and $\varphi_{k,n}$ given the observed λ and η of $\mathcal{P}^{(i)}_{m,n(t-1)}$.

Before updating any parameters, a re-sampling process is conducted. We replace the particle set $\mathcal{P}_{m,n(t-1)}$ with a new set $\mathcal{P}_{m,n(t)}$, where $\mathcal{P}_{m,n(t)}$ is generated from $\mathcal{P}_{m,n(t-1)}$ using sampling with replacement based on the weights of particles. Then sequential parameter updating is based on $\mathcal{P}_{m,n(t)}$.

2) Latent State Inference: Provided with the new observation $x_{m,t}$ and $r_{m,t}$ at time t, the random state $z_{m,t}$ can be one of K topics, and the posterior distribution of $z_{m,t}$ is shown as follows:

$$z_{m,t} | x_{m,t}, r_{m,t}, \mathcal{P}^{(i)}_{m,n(t-1)} \sim Mult(\theta), \tag{22}$$

where $\theta \in \mathcal{R}^K$ is the parameter specifying the multinominal distribution. According to Equation 21, since

$$p(z_{m,t} | x_{m,t}, r_{m,t}, \lambda, \eta) \propto p(z_{m,t}, x_{m,t} | r_{m,t}, \lambda, \eta),$$

θ can be computed as

$$\theta_k \propto E(\mathbf{p}_{m,k} | r_{m,t}, \lambda) E(\varphi_{k,n} | r_{m,t}, \eta).$$

Further, $E(\mathbf{p}_{m,k}|r_{m,t}, \lambda)$ and $E(\varphi_{k,n}|r_{m,t}, \eta)$ can be obtained

as follows,

$$E(\mathbf{p}_{m,k}|r_{m,t},\lambda) = \frac{\mathcal{I}(z_{m,t}=k)r_{m,t}+\lambda_k}{\sum_{k=1}^{K}[\mathcal{I}(z_{m,t}=k)r_{m,t}+\lambda_k]},$$
$$E(\varphi_{k,n}|r_{m,t},\eta) = \frac{\mathcal{I}(x_{m,t}=n)r_{m,t}+\eta_{k,n}}{\sum_{n=1}^{N}[\mathcal{I}(x_{m,t}=n)r_{m,t}+\eta_{k,n}]}, \quad (23)$$

where $\mathcal{I}(\bullet)$, an indicator function, returns 1 when the input Boolean expression is true, otherwise returns 0. Specifically, if $r_{m,t} \in \{0,1\}$, the value of $r_{m,t}$ indicates whether $x_{m,t}$ should be included in the preferred automation list for the ticket problem m. If $r_{m,t} \in [0,+\infty)$, the value of $r_{m,t}$ implies the probability of the automation $x_{m,t}$ can fix the ticket problem m. Therefore, our solution can effectively handle the non-negative rating score at different scales.

3) Parameter Statistics Inference: At time $t-1$, the sufficient statistics for the parameter random variables ($\mathbf{q}_n$, σ_n^2, $\mathbf{p}_m$, φ_k) are ($\mu_{\mathbf{q}}$, $\mathbf{\Sigma}_{\mathbf{q}}$, α, β, λ, η). Assume $\mu'_{\mathbf{q}}$, $\mathbf{\Sigma}'_{\mathbf{q}}$, α', β', λ', η' are the sufficient statistics at time t, which are updated on the basis of both the sufficient statistics at time $t-1$ and the new observation data (i.e, $x_{m,t}$ and $r_{m,t}$). The sufficient statistics for parameters are updated as follows:

$$\mathbf{\Sigma}'_{\mathbf{q}_n} = (\mathbf{\Sigma}_{\mathbf{q}_n}^{-1} + \mathbf{p}_m\mathbf{p}_m^{\intercal})^{-1},\ \mu'_{\mathbf{q}_n} = \mathbf{\Sigma}'_{\mathbf{q}_n}(\mathbf{\Sigma}_{\mathbf{q}_n}^{-1}\mu_{\mathbf{q}_n} + \mathbf{p}_m r_{m,t})$$
$$\alpha' = \alpha + \frac{1}{2},\ \beta' = \beta + \frac{1}{2}(\mu_{\mathbf{q}_n}^{\intercal}\mathbf{\Sigma}_{\mathbf{q}_n}^{-1}\mu_{\mathbf{q}_n} + r_{m,t}^{\intercal}r_{m,t} - \mu_{\mathbf{q}_n}'^{\intercal}\mathbf{\Sigma}_{\mathbf{q}_n}'^{-1}\mu'_{\mathbf{q}_n})$$
$$\lambda'_k = \mathcal{I}(z_{m,t}=k)r_{m,t} + \lambda_k,\ \eta'_{k,n} = \mathcal{I}(x_{m,t}=n)r_{m,t} + \eta_{k,n}.$$

At time t, the sampling process for the parameter random variables σ_n^2, $\mathbf{q}_n$, $\mathbf{p}_m$ and φ_k is summarized as below:

$$\sigma_n^2 \sim \mathcal{IG}(\alpha',\beta'), \quad \mathbf{q}_n|\sigma_n^2 \sim \mathcal{N}(\mu'_{\mathbf{q}_n}, \sigma_n^2\mathbf{\Sigma}'_{\mathbf{q}_n}),$$
$$\mathbf{p}_m \sim Dir(\lambda'), \quad \varphi_k \sim Dir(\eta').$$

4) Integration with Policies: In this section, two policies (i.e., Thompson sampling and UCB) for interactive collaborative filtering are integrated with ICTR model. The latent vectors $\mathbf{p}_m$ and $\mathbf{q}_n$, sampled from their corresponding posterior distributions by Equation 24 at time $t-1$, are used to predict the rating for each automation.

In our model, given ticket problem m, every automation has B independent particles. Each particle i contains its latent variables and parameters, and produces an independent reward prediction $r_{m,t}^{(i)}$. The rating of recommending automation n is predicted with the average value of reward from B particles. The policy based on Thompson sampling selects an automation $n(t)$ based on the following equation,

$$n(t) = \arg\max_n (\bar{r}_{m,n}), \quad (24)$$

where $\bar{r}_{m,n}$ denotes the average reward, i.e.,

$$\bar{r}_{m,n} = \frac{1}{B}\sum_{i=1}^{B}\mathbf{p}_m^{(i)\intercal}\mathbf{q}_n^{(i)}.$$

Moreover, the UCB policy selects an automation based on the upper bound of the predicted rating. Assuming that

$$r_{m,t}^{(i)} \sim \mathcal{N}(\mathbf{p}_m^{(i)\intercal}\mathbf{q}_n^{(i)}, {\sigma^{(i)}}^2),$$

the UCB based policy is developed by the mean and variance of predicted rating, i.e.,

$$n(t) = \arg\max_n (\bar{r}_{m,n} + \gamma\sqrt{\nu}), \quad \nu = \frac{1}{B}\sum_{i}^{B}{\sigma^{(i)}}^2, \quad (25)$$

where $\gamma \geq 0$ is a predefined threshold, and ν is the variance.

5) Algorithm: Putting all the aforementioned inferences together, an algorithm based on ICTR model is provided below.

Online inference for the interactive collaborative filtering problem starts with MAIN procedure, as presented in Algorithm 2. As ticket problem m arrives at time t, the EVAL procedure computes a score for each automation, where we define the score as the average rating. The automation with the highest score is selected to pull. After receiving a rating by recommending an automation, the new feedback is used to update ICTR model by the UPDATE procedure. Especially in the UPDATE procedure, we use the $resample\text{-}propagate$ strategy in particle learning [17] rather than the $propagate\text{-}resample$ strategy in particle filtering [16]. With the $resample\text{-}propagate$ strategy, the particles are re-sampled by taking $\rho^{(i)}$ as the i^{th} particle's weight, where $\rho^{(i)}$ indicates the fitness for the observation at time t given the particle at time $t-1$. The $resample\text{-}propagate$ strategy is considered as an optimal and fully adapted strategy, avoiding an importance sampling step.

Algorithm 2 The algorithm for ICTR model

procedure MAIN(B) ▷ Main entry.
  Initialize B particles, i.e., $\mathcal{P}_{m,n(0)}^{(1)} \cdots \mathcal{P}_{m,n(0)}^{(B)}$.
  for $t \leftarrow 1, T$ **do**
    Ticket problem m arrives for automation recommendation.
    $n(t) = \arg\max_{n=1,N}$ EVAL(m,n) by Equation 24 or Equation 25.
    Receive $r_{m,t}$ by rating automation $n(t)$.
    UPDATE(m, $n(t)$, $r_{m,t}$).
  end for
end procedure

procedure EVAL(m, n) ▷ Get a rating score for item n given ticket problem m.
  for $i \leftarrow 1, B$ **do** ▷ Iterate on each particle.
    Get the ticket problem latent vector $\mathbf{p}_m^{(i)}$.
    Get the automation latent vector $\mathbf{q}_n^{(i)}$.
    Predict i^{th} rating $r_{m,t}^{(i)}$.
  end for
  Compute the average rating as the final rating $r_{m,t}$.
  return the score.
end procedure

procedure UPDATE(m, $n(t)$, $r_{m,t}$) ▷ Update the inference.
  for $i \leftarrow 1, B$ **do** ▷ Compute weights for each particle.
    Compute weight $\rho^{(i)}$ of particle $\mathcal{P}_{m,n(t)}^{(i)}$ by Equation 20.
  end for
  Re-sample $\mathcal{P}'_{m,n(t)}$ from $\mathcal{P}_{m,n(t)}$ according to the weights $\rho^{(i)}$s.
  for $i \leftarrow 1, B$ **do** ▷ Update statistics for each particle.
    Update the sufficient statistics for $z_{m,t}$ by Equation 23.
    Sample $z_{m,t}$ according to Equation 22.
    Update the statistics for $\mathbf{q}_n$, σ_n^2, $\mathbf{p}_m$, φ_k by Equation 24.
    Sample $\mathbf{q}_n$, σ_n^2, $\mathbf{p}_m$, φ_k according to Equation 24.
  end for
end procedure

In addition, existing algorithms [6], [13] consider all the arms independently, while our model takes the clusters of arms into account by learning the topic-related random variables (e.g., φ_k).

IV. Empirical Studies

Our proposed solution is deployed into the ITSM workflow. Extensive studies are conducted on a real ticket dataset to indicate the efficiency of proposed algorithms. We first describe the dataset and evaluation method. Then we analyze the comparative study results of the proposed and baseline algorithms. To further demonstrate the merits of our algorithms, we conduct two case studies.

A. Evaluation Dataset

The evaluation dataset was collected by IBM Tivoli Monitoring system [19] from July 2016 to March 2017, and contains 332,211

historical records. After filtering out those unqualified ones for recommendation algorithms, $116,429$ records are available for empirical studies. The dataset contains 1,091 alert keys (e.g., cpusum_xuxc_aix, prccpu_rlzc_std) and 62 automations (e.g., NFS automation, process CPU spike automation) in total. The execution feedback (i.e., reward or rating) indicating whether the ticket has been resolved by an automation or needs to be escalated to human engineers, is collected and utilized for our proposed model inference. Each record is stamped with the reporting time of the ticket. Additionally, a three-layer hierarchy $\mathcal{H}$ (see Figure 5) given by domain experts is introduced to depict the dependencies among automations.

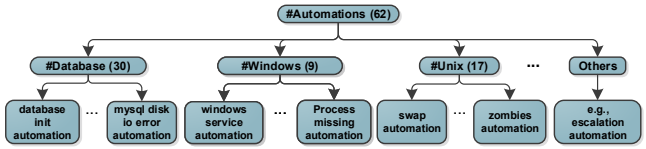


Figure 5: An automation hierarchy defined by domain experts.

We now discuss how to construct the training datasets for HMAB and ICTR model, respectively. To reduce the noise of the dataset, domain experts selected the categorical attributes (e.g., ALERT_ KEY, CLIENT_ ID, etc.) with high representative information of each ticket. These categorical information is encoded into a binary vector [20]. In addition, we augmented a constant feature with value 1 for all vectors. Therefore, for training HAMB model, a ticket is represented as a binary feature vector with $1,182$ dimensions, while for ICTR model, a training record is composed of the alert key id, automation id, rating and timestamp. We refer the dataset for ICTR model as a rating dataset.

B. Evaluation Method

The *replayer* method, an unbiased offline evaluation via the historical logs [20], is applied to evaluate our proposed models. The main idea of *replayer* is to replay the reporting process for each ticket to the testing algorithm for evaluation. If the recommended automation is identical to the one used to resolve the ticket in the historical record, this is referred to as a match. The number of matches where tickets are successfully resolved by the recommended automation is referred as to the accumulated rewards in total. The ratio between accumulated rewards and the total number of matches is used to compute the success rate. The study in [20] indicates that the success rate estimated by the unbiased *replayer* method approaches the real success rate of the deployed online system.

C. Relative Success Rate Optimization

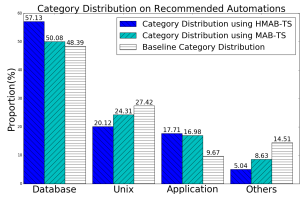
We use success rate as the evaluation metric in our empirical studies. The higher the success rate, the better the performance of the algorithm. To avoid the leakage of business-sensitive information, RSR (relative success rate) is reported. The RSR is the overall success rate of an algorithm divided by the overall success rate of random selection method where an automation is randomly recommended for ticket resolving. The following outlines the experimental performance of HMAB and ICTR models.

We demonstrate the performance of our proposed algorithms by comparing with the baselines including ϵ-greedy [21], Thompson sampling [22], UCB [23], and LinUCB [5]). Figure 6a, Figure 6b, and Figure 6c show the performance comparison between HMABs and the corresponding baselines configured with different parameter settings. To clarify, we only list the performance for LinUCB and HMAB-LinUCB with the parameter $\lambda > 1$ in Figure 6c to reveal the merits of HMAB-LinUCB since both algorithms perform similarly when $\lambda < 1$. By observing the results, HMABs perform much better than the baselines and HMAB-LinUCB outperforms all other algorithms. After grid search for different parameter settings, we list parameter settings with the best performance for ICTRs as well as the baseline algorithms over the rating dataset, and compare the best performance of them in Figure 6d. We find that both ICTRTS $(5, 3)$ and ICTRUCB $(5, 5, 1.0)$ can achieve the best performance as well as learn the latent features of ticket problem and automation, where the first input parameter of ICTRTS and ICTRUCB is the dimension of latent feature vector, the second one represents the number of particles, and the third one of ICTRUCB is used to balance the tradeoff between exploration and exploitation. It is worth noting that RSR sometimes goes down in the next bucket due to the arrival of a large number of new ticket problems. Throughout these empirical studies, we conclude that the proposed algorithms (i.e., HMABs, ICTRs) outperform all the baselines that assume arms are independent.

D. Case Study

In order to illustrate the merits of the proposed solution, we conduct two case studies: 1) how to solve the cold-start problem; and 2) how to infer an automation's latent category in ITSM.

Since there are no historical records for resolving an escalated ticket, recommending an automation for such a ticket problem can be regarded as a cold-start problem. Note that both our proposed algorithms and classical MABs are able to handle the cold-start problem by *exploration*. Herein HMAB-TS is used in this case study. To compare their performance for the cold-start problem, we calculate the distribution of their recommended automations over different categories (e.g., database, Unix, etc.). An escalated ticket repeatedly reported over time (see Figure 7b), describes a database problem. Intuitively, the automations of database category should have a higher chance to be recommended. According to the predefined hierarchy, the category distributions of these recommended automations by our proposed HMABs and conventional MABs are provided in Figure 7a as well as the baseline category distribution, which is the prior category distribution obtained from the ticket dataset. From Figure 7a, we conclude that 1) HMAB-TS explores more automations of the database category compared with TS; and 2) in HMAB-TS, the database category has the highest percentage among all the automation categories. The results show that our proposed HMABs can achieve better performance powered by the hierarchical information. To further illustrate the effectiveness of HMABs, the category distribution on the recommended automations for the escalated ticket is provided in detail. As shown in Figure 7b, the automations from the database category (e.g., database instance down automation, db2 database inactive automation) are frequently recommended according to the context of the ticket, which clearly indicates the issue is due to the inactive database. By checking the recommended results, domain experts figure out that the **database instance down automation**, one of the top recommended automations, can successfully fix such a cold-start ticket problem, clearly demonstrating the effectiveness of our proposed model.

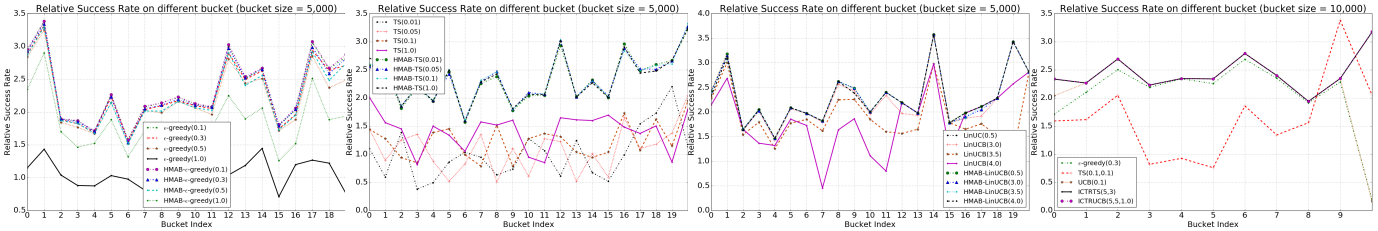


(a)

ALERT KEY	XXX_dbinact_grzc_std	AUTOMATION NAME	Escalation Automation
TICKET SUMMARY	Database xxx status is inactive.	TICKET RESOLUTION	The database is down. It has been restarted, hence closing the ticket.
RECOMMENDED CATEGORY	(%)	RECOMMENDED AUTOMATION	
DATABASE*	57.13	(1) **database** instance **down** automation; (2) db2 **database inactive** automation; (3) mysql **database offline** automation.	
UNIX	20.12	(1) asm space check diskgroup **dbautomation;** (2) **hostdown** automation; (3) certification **expiration** automation.	
APPLICATION	17.71	(1) ntp **restart** automation; (2) mq manager **down** automation.	
OTHERS	5.04	(1) system load automation; (2) others.	

(b)

Figure 7: (a) The comparison of category distribution on the recommended automations; (b) The *exploration* by HMAB-TS on an escalated ticket.



(a) ϵ-greedy and HMAB-ϵ-greedy. (b) TS and HMAB-TS. (c) LinUCB and HMAB-LinUCB. (d) ϵ-greedy, TS, UCB and ICTR.

Figure 6: The RSR of different algorithms on the dataset is given along different time buckets with diverse parameter settings.

Considering Challenge 3, another case study is carried out with an attempt to identify the category for an automation named "process missing" using ICTR model. Based on our previous discussion, ICTRTS $(5, 3)$ can achieve the best performance on this dataset, where the dimension of the latent feature vector is 5 and the number of particles is 3. Through iteratively running on the rating dataset, ICTRTS $(5, 3)$ fully learns the latent feature vector of each automation which also performing the recommendation. Four automations are randomly selected from the automation pool and listed in Figure 8. Different from the automation "process missing," the other four automations can be easily figured out their correct categories according to their names. We compute the Euclidean distances between the latent feature vector of "process missing" and the ones of the other four categorized automations. We consider it as an automation related to "WINDOWS" category as the minimum distance indicated in Figure 8. The correctness of categorization is verified by the domain experts, thus demonstrating ICTR's capability of discovering the automation categories by clustering the learned latent features.

UNCATEGORIZED AUTOMATION	process missing	
CATEGORIZED AUTOMATION	**CATEGORY**	**EUCLIDEAN DISTANCE**
(1) db2 percent db connection executing is to high automation	DATABASE	1.086
(1) process cpu spike automation	UNIX	1.014
(2) swap automation		0.858
(1) windows service automation	**WINDOWS***	**0.565***

Figure 8: An example of inferring the latent category of an automation.

V. Related Work

In this section, we highlight existing research relevant to our work. The automation of ITSM is largely achieved through service-providing facilities in combination with automation of routine procedures such as *problem detection*, *problem determination*, and *resolution recommendation*. System monitoring software, e.g., IBM Tivoli Monitoring [19], is typically used for automated problem detection. In [24], the authors describe an implementation of an integrated framework for minimizing the number of false positive events in monitoring. For automated problem determination, Zeng et al. [8] devised a hierarchical multi-label classification method over ticket data to classify problem types. Discovery of temporal causal relationships between historical events for root cause determination described in [25], [26]. However, automated resolution recommendation [27] is still a challenge since it requires vast domain knowledge of the targeted services. Previous studies [2], [3] focus on recommending proper resolutions to a manually created ticket. Wang et al. [2] proposed a cognitive framework with an attempt to improve this procedure using ontology modeling. A deep neural network ranking model [3] was utilized for recommending the top-n matching resolutions by quantifying the quality of each historical resolution. However, few existing works has attempted to address the aforementioned challenges for auto-generated tickets by the monitoring system.

As abundant online services emerged, interactive recommender systems have become increasingly critical. For an individual user, they continuously refine the recommendation results using feedback [6]. The tradeoff between *exploration* and *exploitation* inherent in learning from interactive feedback has been well dealt with bandit algorithms [22], [5], [28], [7]. However, most prior works (e.g., UCB (upper confidence bound) [23] and Thompson sampling [22]) assume arms are independent, which rarely holds true in reality. Since the real-world items tend to be correlated with each other, a delicate framework [29] is developed to study the bandit problem with dependent arms. Pandey et al. [30] used the taxonomy structure to exploit dependent arms in the context-free bandit setting. The CoFineUCB approach [31] utilized a coarse-to-fine feature hierarchy to reduce the cost of exploration, where the hierarchy was estimated by a small number of existing user profiles. Different from these studies, we proposed an HMAB algorithm, where the hierarchy is constructed by domain experts based on the features of items.

To address Challenge 3, we studied the interactive collaborative filtering model [32], [33], [34] that integrates CF (collaborative filtering) [11], [12] and topic modeling [35] in an offline setting. In [13], an efficient Thompson sampling algorithm named PTS (particle Thompson sampling) addresses the problem in a completely online mode. To differentiate from the above mentioned work, we proposed ICTR model adopting a generative process to explicitly formulate the dependencies as the clusters on arms. In addition, some studies [36], [37] consider the user-side clustering in the collaborative bandit setting. In contrast, our work is orthogonal to these work, since we mainly focus on items soft clustering (i.e. arm dependencies).

VI. Conclusion and Future Work

We inspected the current procedures in ITAS and identified three challenges for optimization of the online automation recommendation. To address these challenges, we formulated the online automation recommendation problem as a multi-armed bandit problem. We developed an intelligent integrated system to promptly suggest the most matched automations for resolving a ticket, where both explicit and implicit automation dependencies can be effectively exploited. Empirical studies in real IT environment are conducted to show the advantages of our solutions.

We pursue two main future research directions. One direction uses multi-document summarization methods based on the problem inference to create a scripted resolution for a new ticket and automatically create the automation in ITAS. Second, we're working

on employing DNN technology [38] for ticket representation to effectively process both categorical and non-categorical attributes of IT tickets.

VII. Acknowledgements

The work was supported in part by the National Science Foundation under Grant Nos. CNS-1461926 and FIU Dissertation Year Fellowship.

References

[1] IBM, "Enterprise it automation services," 2017, visited on January 30, 2018. [Online]. Available: http://www.redbooks.ibm.com/redpapers/pdfs/redp5363.pdf

[2] Q. Wang, W. Zhou, C. Zeng, T. Li, L. Shwartz, and G. Y. Grabarnik, "Constructing the knowledge base for cognitive itsm," in *SCC*. IEEE, 2017, pp. 410–417.

[3] W. Zhou, W. Xue, R. Baral, Q. Wang, C. Zeng, T. Li, J. Xu, Z. Liu, L. Shwartz, and G. Ya Grabarnik, "Star: A system for ticket analysis and resolution," in *SIGKDD*. ACM, 2017, pp. 2181–2190.

[4] A. I. Schein, A. Popescul, L. H. Ungar, and D. M. Pennock, "Methods and metrics for cold-start recommendations," in *SIGIR*. ACM, 2002, pp. 253–260.

[5] L. Li, W. Chu, J. Langford, and R. E. Schapire, "A contextual-bandit approach to personalized news article recommendation," in *WWW*. ACM, 2010, pp. 661–670.

[6] X. Zhao, W. Zhang, and J. Wang, "Interactive collaborative filtering," in *CIKM*. ACM, 2013, pp. 1411–1420.

[7] C. Zeng, Q. Wang, S. Mokhtari, and T. Li, "Online context-aware recommendation with time varying multi-armed bandit," in *SIGKDD*. ACM, 2016, pp. 2025–2034.

[8] C. Zeng, W. Zhou, . Li, L. Shwartz, and G. Y. Grabarnik, "Knowledge guided hierarchical multi-label classification over ticket data," *TNSM*, 2017.

[9] S. Agrawal and N. Goyal, "Thompson sampling for contextual bandits with linear payoffs," in *ICML*, 2013, pp. 127–135.

[10] Q. Wang, T. Li, S. Iyengar, L. Shwartz, and G. Y. Grabarnik, "Online IT ticket automation recommendation using hierarchical multi-armed bandit algorithms," in *Proceedings of the 2018 SIAM International Conference on Data Mining*. SIAM, 2018, pp. 657–665.

[11] R. Salakhutdinov and A. Mnih, "Probabilistic matrix factorization," in *NIPS*, 2007.

[12] R. Salakhutdinov and A. Mnih, "Bayesian probabilistic matrix factorization using markov chain monte carlo," in *Proceedings of the 25th International Conference on Machine Learning*. ACM, 2008, pp. 880–887.

[13] J. Kawale, H. H. Bui, B. Kveton, L. Tran-Thanh, and S. Chawla, "Efficient Thompson sampling for online matrix-factorization recommendation," in *NIPS*, 2015.

[14] Q. Wang, C. Zeng, W. Zhou, T. Li, S. S. Iyengar, L. Shwartz, and G. Grabarnik, "Online interactive collaborative filtering using multi-armed bandit with dependent arms," *IEEE Transactions on Knowledge and Data Engineering*, 2018.

[15] A. Smith, A. Doucet, N. de Freitas, and N. Gordon, *Sequential Monte Carlo methods in practice*. Springer Science & Business Media, 2013.

[16] P. M. Djurić, J. H. Kotecha, J. Zhang, Y. Huang, T. Ghirmai, M. F. Bugallo, and J. Miguez, "Particle filtering," *Signal Processing, IEEE*, vol. 20, no. 5, pp. 19–38, 2003.

[17] C. Carvalho, M. S. Johannes, H. F. Lopes, and N. Polson, "Particle learning and smoothing," *Statistical Science*, vol. 25, no. 1, pp. 88–106, 2010.

[18] C. Zeng, Q. Wang, W. Wang, T. Li, and L. Shwartz, "Online inference for time-varying temporal dependency discovery from time series," in *Bigdata*. IEEE, 2016, pp. 1281–1290.

[19] I. Tivoli, "Integrated Service Management," 2016, visited on January 30, 2018. [Online]. Available: http://ibm.com/software/tivoli/

[20] L. Li, W. Chu, J. Langford, T. Moon, and X. Wang, "An unbiased offline evaluation of contextual bandit algorithms with generalized linear models," *JMLR*, vol. 26, pp. 19–36, 2012.

[21] M. Tokic, "Adaptive ε-greedy exploration in reinforcement learning based on value differences," in *KI 2010: Advances in AI*. Springer, 2010, pp. 203–210.

[22] O. Chapelle and L. Li, "An empirical evaluation of thompson sampling," in *NIPS*, 2011, pp. 2249–2257.

[23] P. Auer, "Using confidence bounds for exploitation-exploration trade-offs," *JMLR*, vol. 3, no. Nov, pp. 397–422, 2002.

[24] L. Tang, T. Li, L. Shwartz, F. Pinel, and G. Y. Grabarnik, "An integrated framework for optimizing automatic monitoring systems in large it infrastructures," in *SIGKDD*. ACM, 2013, pp. 1249–1257.

[25] C. Zeng and T. Li, "Event pattern mining," *Event Mining: Algorithms and Applications*, pp. 71–121, 2015.

[26] C. Zeng, L. Tang, W. Zhou, T. Li, L. Shwartz, G. Y. Grabarnik *et al.*, "Integrated framework for mining temporal logs from fluctuating events," *TSC*, 2017.

[27] L. Tang, T. Li, L. Shwartz, and G. Y. Grabarnik, "Recommending resolutions for problems identified by monitoring," in *IFIP/IEEE IM*. IEEE, 2013, pp. 134–142.

[28] L. Tang, Y. Jiang, L. Li, C. Zeng, and T. Li, "Personalized recommendation via parameter-free contextual bandits," in *SIGIR*. ACM, 2015, pp. 323–332.

[29] S. Pandey, D. Chakrabarti, and D. Agarwal, "Multi-armed bandit problems with dependent arms," in *ICML*. ACM, 2007, pp. 721–728.

[30] S. Pandey, D. Agarwal, and V. Chakrabarti, D.and Josifovski, "Bandits for taxonomies: A model-based approach," in *SDM*. SIAM, 2007, pp. 216–227.

[31] Y. Yue, S. A. Hong, and C. Guestrin, "Hierarchical exploration for accelerating contextual bandits," *preprint arXiv:1206.6454*, 2012.

[32] C. Wang and D. M. Blei, "Collaborative topic modeling for recommending scientific articles," in *SIGKDD*. ACM, 2011, pp. 448–456.

[33] S. Purushotham, Y. Liu, and C.-C. J. Kuo, "Collaborative topic regression with social matrix factorization for recommendation systems," *arXiv:1206.4684*, 2012.

[34] K. Wang, W. Zhao, H. Peng, and X. Wang, "Bayesian probabilistic multi-topic matrix factorization for rating prediction," 2017.

[35] D. M. Blei, A. Y. Ng, and M. I. Jordan, "Latent dirichlet allocation," *JMLS*, vol. 3, no. Jan, pp. 993–1022, 2003.

[36] Q. Wu, H. Wang, Q. Gu, and H. Wang, "Contextual bandits in a collaborative environment," in *SIGIR*. ACM, 2016, pp. 529–538.

[37] L. Zhou and E. Brunskill, "Latent contextual bandits and their application to personalized recommendations for new users," *preprint arXiv:1604.06743*, 2016.

[38] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in Neural Information Processing Systems*, 2012, pp. 1097–1105.