Online Context-Aware Recommendation with Time Varying Multi-Armed Bandit

Chunqiu Zeng, Qing Wang, Shekoofeh Mokhtari, Tao Li
School of Computing and Information Science
Florida International University
Miami, USA
{czeng001,qwang028, smokh004, taoli}@cs.fiu.edu

ABSTRACT

Contextual multi-armed bandit problems have gained increasing popularity and attention in recent years due to their capability of leveraging contextual information to deliver online personalized recommendation services (e.g., online advertising and news article selection). To predict the reward of each arm given a particular context, existing relevant research studies for contextual multi-armed bandit problems often assume the existence of a fixed yet unknown reward mapping function. However, this assumption rarely holds in practice, since real-world problems often involve underlying processes that are dynamically evolving over time.

In this paper, we study the time varying contextual multi-armed problem where the reward mapping function changes over time. In particular, we propose a dynamical context drift model based on particle learning. In the proposed model, the drift on the reward mapping function is explicitly modeled as a set of random walk particles, where good fitted particles are selected to learn the mapping dynamically. Taking advantage of the fully adaptive inference strategy of particle learning, our model is able to effectively capture the context change and learn the latent parameters. In addition, those learnt parameters can be naturally integrated into existing multi-arm selection strategies such as `LinUCB` and `Thompson sampling`. Empirical studies on two real-world applications, including online personalized advertising and news recommendation, demonstrate the effectiveness of our proposed approach. The experimental results also show that our algorithm can dynamically track the changing reward over time and consequently improve the click-through rate.

Keywords

Recommender System; Personalization; Time Varying Contextual Bandit; Probability Matching; Particle Learning

1. INTRODUCTION

Online personalized recommender systems strive to promptly feed the consumers with appropriate items (e.g., advertisements, news articles) according to the current context including both the consumer and item content information, and try to continuously maximize the consumers' satisfaction in the long run. To achieve this goal, it becomes a critical task for recommender systems to track the consumer preferences instantly and to recommend the interesting items to the users from a large item repository.

However, identifying the appropriate match between the consumer preferences and the target items is quite difficult for recommender systems due to several existing challenges in practice [18]. One is the well-known *cold-start* problem since a significant number of users/items might be completely new to the system, that is, they may have no consumption history at all. This problem makes recommender systems ineffective unless additional information about both items and users is collected [9][7]. Second, both the popularity of item content and the consumer preferences are dynamically evolving over time. For example, the popularity of a movie usually keeps soaring for a while after its first release, then gradually fades away. Meanwhile, user interests may evolve over time.

Herein, a context-based *exploration/exploitation* dilemma is identified in the aforementioned setting. A tradeoff between two competing goals needs to be considered in recommender systems: maximizing user satisfaction using the consumption history, while gathering new information for improving goodness of match between user preference and items [16]. This dilemma is typically formulated as a contextual multi-armed bandit problem where each arm corresponds to one item. The recommendation algorithm determines the strategies for selecting an arm to pull according to the contextual information at each trial. Pulling an arm indicates the corresponding item is recommended. When an item matches the user preference (e.g., a recommended news article or ad is clicked), a reward is obtained; otherwise, there is no reward. The reward information is fed back to the algorithm to optimize the strategies. The optimal strategy is to pull the arm with the maximum expected reward with respect to the contextual information on each trial, and then to maximize the total accumulated reward for the whole series of trials.

Recently, a series of algorithms for contextual multi-armed bandit problems have been reported with promising performance under different settings, including unguided exploration (e.g., ϵ-greedy [26] and epoch-greedy [15]) and guided exploration (e.g., LinUCB [16] and Thompson Sampling [8]). These existing algorithms take the contextual information as the input and predict the expected reward for each arm, assuming the reward is invariant under the same context. However, this assumption rarely holds in practice since the real-world problems often involve some underlying processes that are dynamically evolving over time and not all latent influencing factors are included in the context information. As a result, the expected reward of an arm is time varying even though the contextual information does not change.

KDD '16, August 13-17, 2016, San Francisco, CA, USA

DOI: http://dx.doi.org/10.1145/2939672.2939878

A Motivated Example: Here we use a news recommendation example to illustrate the time varying behaviors of the reward. In the example, the click through rate (abbr., CTR) and the news articles correspond to the reward and the arms, respectively. Five news articles are randomly selected and their corresponding user-article interaction records are extracted from the Yahoo! news repository [17, 16]. The context consists of both the user and article information. Although the context information of each article does not change, its average CTR varies dynamically over time as shown in Figure 1. The same contextual information may have different impacts on the CTR at different times.

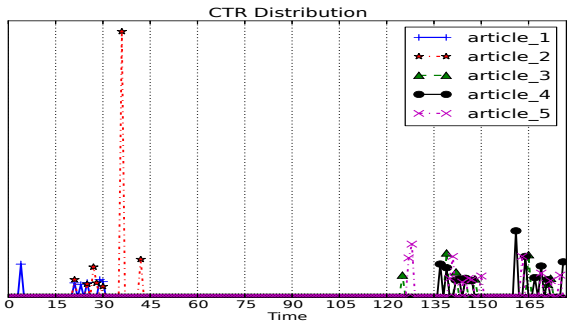


Figure 1: Given the same contextual information for each article, the average CTR distribution of five news articles from Yahoo! news repository is displayed. The CTR is aggregated by every hour.

In this paper, to capture the time varying behaviors of the reward in contextual multi-armed bandit problems, we propose a dynamical context drift model based on particle learning and develop effective on-line inference algorithms. The dynamic behaviors of the reward is explicitly modeled as a set of random walk particles. The fully adaptive inference strategy of particle learning allows our model to effectively capture the context change and learn the latent parameters. In addition, the learnt parameters can be naturally integrated into existing multi-arm selection strategies such as `LinUCB` and `Thompson sampling`. We conduct empirical studies on two real-world applications, including online personalized advertising and news recommendation and the experimental results demonstrate the effectiveness of our proposed approach.

The rest of this paper is organized as follows. In Section 2, we describe a brief summary of prior work relevant to the contextual multi-armed bandit problem and the online inference with particle learning. We formulate the problem in Section 3. The solution to the problem is presented in Section 4. Extensive empirical evaluation results are reported in Section 5. Finally, we reach the conclusion in Section 6.

2. RELATED WORK

In this paper, we come up with a context drift model to deal with the contextual multi-armed bandit problem, where the dynamic behaviors of reward is explicitly considered. A sequential online inference method is developed to learn the latent unknown parameters and infer the latent states simultaneously. In this section, we highlight existing literature studies that are related to our proposed approach for online context-aware recommendation.

2.1 Contextual Multi-armed Bandit

Our work is primarily relevant to the research area in the multi-armed bandit problem which was first introduced in [22]. The multi-armed bandit problem is identified in diverse applications, such as online advertising [20, 14], web content optimization [21, 1], and robotic routing [4]. The core task of the multi-armed bandit problem is to balance the tradeoff between exploration and exploitation. A series of algorithms have been proposed to deal with this problem including ϵ-greedy [26], upper confidence bound (UCB) [5, 19], EXP3 [3], and Thompson sampling [2].

Contextual multi-armed bandit problem is an instance of bandit problem, where the contextual information is utilized for arm selection. It is widely used for personalized recommendation service to address the `cold-start` problem [9]. Lots of existing multi-armed bandit algorithms have been extended to incorporating the contextual information.

Contextual ϵ-greedy algorithm has been introduced by extending the ϵ-greedy strategy with the consideration of context [5]. This algorithm chooses the best arm based on current knowledge with the probability $1 - \epsilon$, while chooses one arm uniformly with the probability ϵ.

Both LinUCB and LogUCB algorithms extend the UCB algorithm to contextual bandits [5, 19]. $LinUCB$ assumes a linear mapping function between the expected reward of an arm and its corresponding context. In [19], the LogUCB algorithm is proposed to deal with the contextual bandit problem based on logistic regression.

Thompson sampling [8], one of earliest heuristics for the bandit problem, belongs to the probability matching family. Its main idea is to randomly allocate the pulling chance according to the probability that an arm gives the largest expected reward given the context.

A most recent research work on the contextual bandit problem in [25] comes up with a novel parameter-free algorithm based on a principled sampling approach. This approach makes use of the online bootstrap sample to derive the distribution of estimated models in an on-line manner. In [24], an ensemble strategies combined with a meta learning paradigm is proposed to stabilize the output of contextual bandit algorithms.

These existing algorithms make use of the contextual information to predict the expected reward for each arm, with the assumption that the reward is invariant under the same context. However, this assumption rarely holds in real applications. Our paper proposes a context drift model to deal with the contextual multi-armed bandit problem by taking the dynamic behaviors of reward into account.

2.2 Sequential Online Inference

Our proposed model makes use of sequential online inference to infer the latent state and learn unknown parameters. Popular sequential learning methods include sequential monte carlo sampling [12], and particle learning [6].

Sequential Monte Carlo (SMC) methods consist of a set of Monte Carlo methodologies to solve the filtering problem [11]. It provides a set of simulation based methods for computing the posterior distribution. These methods allow inference of full posterior distributions in general state space models, which may be both nonlinear and non-Gaussian.

Particle learning provides state filtering, sequential parameter learning and smoothing in a general class of state space models [6]. Particle learning is for approximating the sequence of filtering and smoothing distributions in light of parameter uncertainty for a wide class of state space models. The central idea behind particle learning is the creation of a particle algorithm that directly samples from the particle approximation to the joint posterior distribution of states and conditional sufficient statistics for fixed pa-

rameters in a fully-adapted `resample-propagate` framework. We borrow the idea of particle learning for both latent state inference and parameter learning.

3. PROBLEM FORMULATION

In this section, we formally define the contextual multi-armed bandit problem first, and then model the time varying contextual multi-armed bandit problem. Some important notations mentioned in this paper are summarized in Table 1.

Table 1: Important Notations

Notation	Description
$a^{(i)}$	the i-th arm.
$\mathcal{A}$	the set of arms, $\mathcal{A} = \{a^{(1)}, ..., a^{(K)}\}$.
$\mathbf{x}_t$	the context at time t, and represented by a vector.
$r_{k,t}$	the reward of pulling the arm $a^{(k)}$ at time t, $a^{(k)} \in \mathcal{A}$.
$y_{k,t}$	the predicted reward for the arm $a^{(k)}$ at time t.
$\mathcal{P}_k$	the set of particles for the arm $a^{(k)}$ and $\mathcal{P}_k^{(i)}$ is the i^{th} particle of $\mathcal{P}_k$.
$\mathbf{S}_{\pi,t}$	the sequence of $(\mathbf{x}_i, \pi(\mathbf{x}_i), r_{\pi(\mathbf{x}_i)})$ observed until time t.
$\mathbf{w}_k$	the coefficient vector used to predict reward of the arm $a^{(k)}$.
$\mathbf{c}_{\mathbf{w}_k}$	the constant part of $\mathbf{w}_k$.
$\delta_{\mathbf{w},t}$	the drifting part of $\mathbf{w}_k$ at time t.
$\eta_{k,t}$	the standard Gaussian random walk at time t, given $\eta_{k,t-1}$.
θ_k	the scale parameters used to compute $\delta_{\mathbf{w},t}$.
π	the policy for pulling arm sequentially.
R_π	the cumulative reward of the policy π.
$f_{a^{(k)}}(\mathbf{x}_t)$	the reward prediction function of the arm $a^{(k)}$, given context $\mathbf{x}_t$.
σ_k^2	the variance of reward prediction for the arm $a^{(k)}$.
α, β	the hyper parameters determine the distribution of σ_k^2.
$\mu_\mathbf{w}$, $\mathbf{\Sigma_w}$	the hyper parameters determine the distribution of $\mathbf{w}_k$.
$\mu_\mathbf{c}$, $\mathbf{\Sigma_c}$	the hyper parameters determine the distribution of $\mathbf{c}_{\mathbf{w}_k}$.
μ_θ, $\mathbf{\Sigma}_\theta$	the hyper parameters determine the distribution of θ_k.
μ_η, $\mathbf{\Sigma}_\eta$	the hyper parameters determine the distribution of $\eta_{k,t}$.

3.1 Basic Concepts and Terminologies

Let $\mathcal{A}$ be a set of arms, denoted as $\mathcal{A} = \{a^{(1)}, a^{(2)}..., a^{(K)}\}$, where K is the number of arms. A d-dimensional feature vector $\mathbf{x}_t \in \mathcal{X}$ represents the contextual information at time t, and $\mathcal{X}$ is the d-dimensional feature space. The contextual multi-armed problem involves a series of decisions over a finite but possibly unknown time horizon T. A policy π makes a decision at each time $t \in [1, T]$ to select the arm $\pi(\mathbf{x}_t)$, one of K arms, to pull based on the contextual information $\mathbf{x}_t$. After pulling an arm, the policy receives a reward from the selected arm. The reward of an arm $a^{(k)}$ at time t is denoted as $r_{k,t}$, whose value is drawn from an unknown distribution determined by the context x_t presented to arm $a^{(k)}$. However the reward $r_{k,t}$ is not available unless arm $a^{(k)}$ is pulled. The total reward received by the policy π is

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

and the goal of the policy π is to maximize the total reward R_π.

Before selecting one arm at time t, a policy π typically learns a model to predict the reward for every arm according to the historical observation, $S_{\pi,t-1} = \{(\mathbf{x}_i, \pi(\mathbf{x}_i), r_{\pi(\mathbf{x}_i)}) | 1 \leq i < t\}$, which consists of a sequence of triplets. The reward prediction helps the policy π make decisions to increase the total reward.

Assume $y_{k,t}$ is the predicted reward of the arm $a^{(k)}$, which is determined by

$$y_{k,t} = f_{a^{(k)}}(\mathbf{x}_t), \tag{1}$$

where the context $\mathbf{x}_t$ is input and $f_{a^{(k)}}$ is the reward mapping function for arm $a^{(k)}$.

One popular mapping function is defined as the linear combination of the feature vector $\mathbf{x}_t$, which has been successfully used in bandit problems [16][2]. Specifically, $f_{a^{(k)}}(\mathbf{x}_t)$ is formally given as follows:

$$f_{a^{(k)}}(\mathbf{x}_t) = \mathbf{x}_t^\intercal \mathbf{w}_k + \varepsilon_k, \tag{2}$$

where $\mathbf{x}_t^\intercal$ is the transpose of contextual information $\mathbf{x}_t$, $\mathbf{w}_k$ is a d-dimensional coefficient vector, and ε_k is a zero-mean Gaussian noise with variance σ_k^2, i.e., $\varepsilon_k \sim \mathcal{N}(0, \sigma_k^2)$. Accordingly,

$$y_{k,t} \sim \mathcal{N}(\mathbf{x}_t^\intercal \mathbf{w}_k, \sigma_k^2). \tag{3}$$

In this setting, a graphical model representation is provided in Figure 2a. The context $\mathbf{x}_t$ is observed at time t. The predicted reward value $y_{k,t}$ depends on random variable $\mathbf{x}_t$, $\mathbf{w}_k$, and σ_k^2. A conjugate prior distribution for the random variables $\mathbf{w}_k$ and σ_k^2 is assumed and defined as $\mathcal{NIG}$(i.e., Normal Inverse Gamma) distribution with the hyper parameters $\mu_\mathbf{w}$, $\mathbf{\Sigma_w}$, α, and β. The distribution is denoted as $\mathcal{NIG}(\mu_\mathbf{w}, \mathbf{\Sigma_w}, \alpha, \beta)$ and shown below:

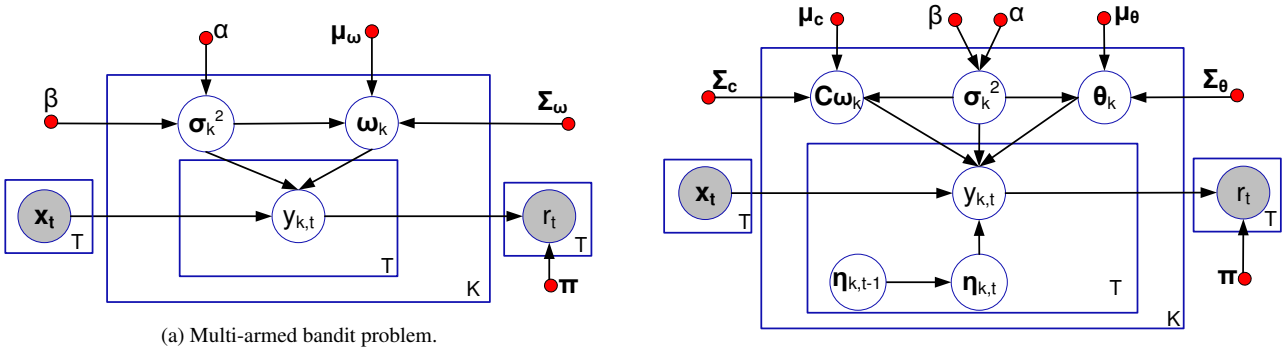
$$\begin{aligned} \mathbf{w}_k | \sigma_k^2 &\sim \mathcal{N}(\mu_\mathbf{w}, \sigma_k^2 \mathbf{\Sigma_w}), \\ \sigma_k^2 &\sim \mathcal{IG}(\alpha, \beta), \end{aligned} \tag{4}$$

where the hyper parameters are predefined.

A policy π selects one arm $a^{(k)}$ to pull according to the reward prediction model. After pulling arm $a^{(k)}$ at time t, a corresponding reward $r_{k,t}$ is observed, while the rewards of other arms are still hidden. A new triplet $(\mathbf{x}_t, \pi(\mathbf{x}_t), r_{\pi(\mathbf{x}_t)})$ is obtained and a new sequence $S_{\pi,t}$ is formed by combining $S_{\pi,t-1}$ with the new triplet. The posterior distribution of $\mathbf{w}_k$ and σ_k^2 given $S_{\pi,t}$ is a $\mathcal{NIG}$ distribution. Denoting the parameters of $\mathcal{NIG}$ distribution at time $t-1$ as $\mu_{\mathbf{w_{t-1}}}$, $\mathbf{\Sigma}_{\mathbf{w_{t-1}}}$, α_{t-1}, and β_{t-1}, the hyper parameters at time t are updated as follows:

$$\begin{aligned} \mathbf{\Sigma_{w_t}} &= (\mathbf{\Sigma_{w_{t-1}}}^{-1} + \mathbf{x}_t \mathbf{x}_t^\intercal)^{-1}, \\ \mu_{\mathbf{w_t}} &= \mathbf{\Sigma_{w_t}}(\mathbf{\Sigma_{w_{t-1}}}^{-1} \mu_{\mathbf{w_{t-1}}} + \mathbf{x}_t r_{\pi(\mathbf{x}_t)}), \\ \alpha_t &= \alpha_{t-1} + \frac{1}{2}, \\ \beta_t &= \beta_{t-1} + \frac{1}{2}[r_{\pi(\mathbf{x}_t)}^2 + \mu_{\mathbf{w_{t-1}}}^\intercal \mathbf{\Sigma_{w_{t-1}}}^{-1} \mu_{\mathbf{w_{t-1}}} - \mu_{\mathbf{w_t}}^\intercal \mathbf{\Sigma_{w_t}}^{-1} \mu_{\mathbf{w_t}}]. \end{aligned} \tag{5}$$

Note that, the posterior distribution of $\mathbf{w}_k$ and σ_k^2 at time t is considered as the prior distribution at time $t+1$. On the basis of the aforementioned inference, a series of algorithms, including Thompson sampling and LinUCB, are proposed to address the contextual multi-armed bandit problem.



(a) Multi-armed bandit problem.

(b) Time varying multi-armed bandit problem.

Figure 2: Graphical model representation for bandit problem. Random variable is denoted as a circle. The circle with gray color filled means the corresponding random variable is observed. Red dot represents a hyper parameter.

Thompson sampling, one of earliest heuristics for the bandit problem [8], belongs to the probability matching family. Its main idea is to randomly allocate the pulling chance according to the probability that an arm gives the largest expected reward given the context. Thompson sampling algorithm for the contextual multi-armed bandit problem involves the following general structure at time t:

1. For each arm $a^{(k)}$, its corresponding σ_k^2 and $\mathbf{w}_k$ are drawn from $\mathcal{NIG}(\mu_{\mathbf{w_{t-1}}}, \mathbf{\Sigma}_{\mathbf{w_{t-1}}}, \alpha_{t-1}, \beta_{t-1})$[1].

2. The arm a^* is selected to pull, and a reward of a^* is obtained, where $a^* = \arg\max_{1 \le k \le K} \{\mathbf{x}_t^\intercal \mathbf{w}_k\}$.

3. After observing the reward $r_{a^*,t}$, the posterior distribution is updated by Equation 5.

LinUCB, another successful contextual bandit algorithm, is an extension of the UCB algorithm [16]. It pulls the arm with the largest score $LinUCB(\lambda)$, defined as below,

$$LinUCB(\lambda) = \underbrace{\mathbf{x}_t^\intercal \mu_{\mathbf{w_{t-1}}}}_{reward\ expectation} + \lambda \underbrace{\frac{1}{\sigma_k}\sqrt{\mathbf{x}_t^\intercal \mathbf{\Sigma}_{\mathbf{w_{t-1}}}^{-1} \mathbf{x}_t}}_{reward\ deviation}. \quad (6)$$

where λ is a parameter to combine the expectation and standard deviation of reward.

Both LinUCB [16] and Thompson Sampling [8] will be incorporated into our dynamic context drift model. More details will be discussed in Section 4 after modeling the context drift.

3.2 Dynamic Context Drift Modeling

As mentioned in Section 3.1, the reward prediction for arm $a^{(k)}$ is conducted by a linear combination of contextual features $\mathbf{x}_t$, with coefficient vector $\mathbf{w}_k$. Each element in the coefficient vector $\mathbf{w}_k$ indicates the contribution of the corresponding feature for reward prediction. The aforementioned model is based on the assumption that $\mathbf{w}_k$ is unknown but fixed [2], which rarely holds in practice. The real-world problems often involve some underlying processes. These processes often lead to the dynamics in the contribution of each context feature to the reward prediction. To account for the dynamics, our goal is to come up with a model having the capability of capturing the drift of $\mathbf{w}_k$ over time and subsequently obtain a better fitted model for the dynamic reward change. Let $\mathbf{w}_{k,t}$ denote the coefficient vector for arm $a^{(k)}$ at time t. Taking the drift of $\mathbf{w}_k$ into account, $\mathbf{w}_{k,t}$ is formulated as follows:

$$\mathbf{w}_{k,t} = \mathbf{c}_{\mathbf{w}_k} + \delta_{\mathbf{w}_{k,t}}, \quad (7)$$

where $\mathbf{w}_{k,t}$ is decomposed into two components including both the stationary component $\mathbf{c}_{\mathbf{w}_k}$ and the drift component $\delta_{\mathbf{w}_{k,t}}$. Both components are d-dimensional vectors. Similar to modeling $\mathbf{w}_k$ in Figure 2a, the stationary component $\mathbf{c}_{\mathbf{w}_k}$ can be generated with a conjugate prior distribution

$$\mathbf{c}_{\mathbf{w}_k} \sim \mathcal{N}(\mu_{\mathbf{c}}, \sigma_k^2 \mathbf{\Sigma}_{\mathbf{c}}), \quad (8)$$

where $\mu_{\mathbf{c}}$ and $\mathbf{\Sigma}_{\mathbf{c}}$ are predefined hyper parameters as shown in Figure 2b.

However, it is difficult to model the drift component $\delta_{\mathbf{w}_{k,t}}$ with a single function due to the diverse characteristics of the context. For instance, in Figure 1, given the same context, the CTRs of some articles change quickly, while some articles may have relatively stable CTRs. Moreover, the coefficients for different elements in the context feature can change with diverse scales. To simplify the inference, we assume that each element of $\delta_{\mathbf{w}_{k,t}}$ drifts independently. Due to the uncertainty of drifting, we formulate $\delta_{\mathbf{w}_{k,t}}$ with a standard Gaussian random walk $\eta_{\mathbf{k,t}}$ and a scale variable θ_k using the following Equation:

$$\delta_{\mathbf{w}_{k,t}} = \theta_k \odot \eta_{\mathbf{k,t}}, \quad (9)$$

where $\eta_{\mathbf{k,t}} \in \mathcal{R}^d$ is the drift value at time t caused by the standard random walk and $\theta_k \in \mathcal{R}^d$ contains the changing scales for all the elements of $\delta_{\mathbf{w}_{k,t}}$. The operator $\odot$ is used to denote the element-wise product. The standard Gaussian random walk is defined with a Markov process as shown in Equation 10.

$$\eta_{\mathbf{k,t}} = \eta_{\mathbf{k,t-1}} + \mathbf{v}, \quad (10)$$

where $\mathbf{v}$ is a standard Gaussian random variable defined by $\mathbf{v} \sim \mathcal{N}(\mathbf{0}, \mathbf{I}_d)$, and $\mathbf{I}_d$ is a $d \times d$-dimensional identity matrix. It is equivalent that $\eta_{\mathbf{k,t}}$ is drawn from the Gaussian distribution

$$\eta_{\mathbf{k,t}} \sim \mathcal{N}(\eta_{\mathbf{k,t-1}}, \mathcal{I}_d). \quad (11)$$

The scale random variable θ_k is generated with a conjugate prior distribution

$$\theta_k \sim \mathcal{N}(\mu_\theta, \sigma_k^2 \mathbf{\Sigma}_\theta), \quad (12)$$

where μ_θ and $\mathbf{\Sigma}_\theta$ are predefined hyper parameters. σ_k^2 is drawn from the Inverse Gamma (abbr., $\mathcal{IG}$) distribution provided in Equation 4.

[1]Note that most exiting works for Thompson sampling assume σ_k^2 is known and $\mathbf{w}_k$ is drawn from $\mathcal{N}(\mu_{\mathbf{w_{t-1}}}, \mathbf{\Sigma}_{\mathbf{w_{t-1}}})$

Combining Equations 7 and 9, we obtain

$$\mathbf{w}_{k,t} = \mathbf{c}_{\mathbf{w_k}} + \theta_k \odot \eta_{\mathbf{k,t}}. \tag{13}$$

According to Equation 2, $y_{k,t}$ is computed as

$$y_{k,t} = \mathbf{x}_t^\intercal(\mathbf{c}_{\mathbf{w_k}} + \theta_k \odot \eta_{\mathbf{k,t}}) + \epsilon_k. \tag{14}$$

Accordingly, $y_{k,t}$ is modeled to be drawn from the following Gaussian distribution

$$y_{k,t} \sim \mathcal{N}(\mathbf{x}_t^\intercal(\mathbf{c}_{\mathbf{w_k}} + \theta_k \odot \eta_{\mathbf{k,t}}), \sigma_k^2). \tag{15}$$

The new context drift model is presented with a graphical model representation in Figure 2b. Compared with the model in Figure 2a, a standard Gaussian random walk $\eta_{k,t}$ and the corresponding scale θ_k for each arm $a^{(k)}$ are introduced in the new model. The new model explicitly formulates the drift of the coefficients for the reward prediction, considering the dynamic behaviors of the reward in real-world application. From the new model, each element value of $\mathbf{c}_{\mathbf{w_k}}$ indicates the contribution of its corresponding feature in predicting the reward, while the element values of θ_k show the scales of context drifting for the reward prediction. A large element value of θ_k signifies a great context drifting occurring to the corresponding feature over time.

4. METHODOLOGY AND SOLUTION

In this section, we present the methodology for online inference of the context drift model.

The posterior distribution inference involves four random variables, i.e., σ_k^2, $\mathbf{c}_{\mathbf{w_k}}$, θ_k, and $\eta_{\mathbf{k,t}}$. According to the graphical model in Figure 2b, the four random variables are grouped into two categories: parameter random variable and latent state random variable. σ_k^2, $\mathbf{c}_{\mathbf{w_k}}$, θ_k are parameter random variables since they are assumed to be fixed but unknown, and their values do not depend on the time. Instead, $\eta_{\mathbf{k,t}}$ is referred to as a latent state random variable since it is not observable and its value is time dependent according to Equation 10. After pulling the arm $a^{(k)}$ according to the context $\mathbf{x}_t$ at time t, a reward is observed as $r_{k,t}$. Thus, $\mathbf{x}_t$ and $r_{k,t}$ 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. However, since the inference partially depends on the random walk which generates the latent state variable, we use the sequential sampling based inference strategy that are widely used sequential monte carlo sampling [23], particle filtering [10], and particle learning [6] to learn the distribution of both parameter and state random variables.

Since state $\eta_{\mathbf{k,t-1}}$ changes over time with a standard Gaussian random walk, it follows a Gaussian distribution after accumulating $t-1$ standard Gaussian random walks. Assume $\eta_{\mathbf{k,t-1}} \sim \mathcal{N}(\mu_{\eta_k}, \Sigma_{\eta_k})$, a particle is defined as follows.

DEFINITION 1 (PARTICLE). *A particle of an arm $a^{(k)}$ is a container which maintains the current status information of $a^{(k)}$. The status information comprises of random variables such as σ_k^2, $\mathbf{c}_{\mathbf{w_k}}$, θ_k, and $\eta_{\mathbf{k,t}}$, and the parameters of their corresponding distributions such as α and β, μ_c and $\mathbf{\Sigma_c}$, μ_θ and $\mathbf{\Sigma}_\theta$, μ_{η_k} and $\mathbf{\Sigma}_{\eta_k}$.*

4.1 Re-sample Particles with Weights

At time $t-1$, each arm $a^{(k)}$ maintains a fixed-size set of particles. We denote the particle set as $\mathcal{P}_{k,t-1}$ and assume the number of particles in $\mathcal{P}_{k,t-1}$ is p. Let $\mathcal{P}_{k,t-1}^{(i)}$ be the i^{th} particles of arm $a^{(k)}$ at time $t-1$, where $1 \le i \le p$. Each particle $\mathcal{P}_{k,t-1}^{(i)}$ has a weight, denoted as $\rho^{(i)}$, indicating its fitness for the new observed data at time t. Note that $\sum_{i=1}^{p} \rho^{(i)} = 1$. The fitness of each particle $\mathcal{P}_{k,t-1}^{(i)}$ is defined as the likelihood of the observed data $\mathbf{x}_t$ and $r_{k,t}$. Therefore,

$$\rho^{(i)} \propto P(\mathbf{x}_t, r_{k,t} | \mathcal{P}_{k,t-1}^{(i)}). \tag{16}$$

Further, $y_{k,t}$ is the predicted value of $r_{k,t}$. The distribution of $y_{k,t}$, determined by $\mathbf{c}_{\mathbf{w}_k}$, θ_k, σ_k^2 and $\eta_{k,t}$, is given in Equation 15. Therefore, we can compute $\rho^{(i)}$ in proportional to the density value given $y_{k,t} = r_{k,t}$. Thus,

$$\rho^{(i)} \propto \iint_{\eta_{k,t},\eta_{k,t-1}} \{\mathcal{N}(r_{k,t}|\mathbf{x}_t^\intercal(\mathbf{c}_{\mathbf{w}_k} + \theta_k \odot \eta_{k,t}), \sigma_k^2) \mathcal{N}(\eta_{k,t}|\eta_{k,t-1}, \mathcal{I}_d)\mathcal{N}(\eta_{k,t-1}|\mu_{\eta_k}, \mathbf{\Sigma}_{\eta_k})\} d\eta_{k,t}\, d\eta_{k,t-1},$$

where state variables $\eta_{k,t}$ and $\eta_{k,t-1}$ are integrated out due to their change over time, and $\mathbf{c}_{\mathbf{w}_k}$, θ_k, σ_k^2 are from $\mathcal{P}_{k,t-1}^{(i)}$. Then we obtain

$$\rho^{(i)} \propto \mathcal{N}(\mathbf{m}_k, \mathbf{Q}_k), \tag{17}$$

where

$$\begin{aligned} \mathbf{m}_k &= \mathbf{x}_t^\intercal(\mathbf{c}_{\mathbf{w}_k} + \theta_k \odot \mu_{\eta_k}) \\ \mathbf{Q}_k &= \sigma_k^2 + \mathbf{x}_t^\intercal \odot \theta_k(\mathcal{I}_d + \mathbf{\Sigma}_{\eta_\mathbf{k}})\theta_k^\intercal \odot \mathbf{x}_t. \end{aligned} \tag{18}$$

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

4.2 Latent State Inference

At time $t-1$, the sufficient statistics for state $\eta_{k,t-1}$ are the mean (i.e., μ_{η_k}) and the covariance (i.e., $\mathbf{\Sigma}_{\eta_k}$). Provided with the new observation data $\mathbf{x}_t$ and $r_{k,t}$ at time t, the sufficient statistics for state $\eta_{k,t}$ need to be re-computed. We apply the Kalman filtering [13] method to recursively update the sufficient statistics for $\eta_{k,t}$ based on the new observation and the sufficient statistics at time $t-1$. Let μ'_{η_k} and $\mathbf{\Sigma'}_{\eta_k}$ be the new sufficient statistics of state $\eta_{k,t}$ at time t. Then,

$$\begin{aligned} \mu'_{\eta_k} &= \mu_{\eta_k} + \underbrace{\mathbf{G}_k(r_{k,t} - \mathbf{x}_t^\intercal(\mathbf{c}_{\mathbf{w}_k} + \theta_k \odot \eta_{k,t-1}))}_{\text{Correction by Kalman Gain}}, \\ \mathbf{\Sigma'}_{\eta_k} &= \mathbf{\Sigma}_{\eta_k} + \mathcal{I}_d - \underbrace{\mathbf{G}_k\mathbf{Q}_k\mathbf{G}_k^\intercal}_{\text{Correction by Kalman Gain}}, \end{aligned} \tag{19}$$

where $\mathbf{Q}_k$ is defined in Equation 18 and $\mathbf{G}_k$ is Kalman Gain [13] defined as

$$\mathbf{G}_k = (\mathcal{I}_d + \mathbf{\Sigma}_{\eta_k})\theta_k \odot \mathbf{x}_t\mathbf{Q}_k^{-1}.$$

As shown in Equation 19, both μ'_{η_k} and $\mathbf{\Sigma'}_{\eta_k}$ are estimated with a correction using Kalman Gain $\mathbf{G}_k$(i.e., the last term in both two formulas). With the help of the sufficient statistics for the state random variable, $\eta_{k,t}$ can be draw from the Gaussian distribution

$$\eta_{k,t} \sim \mathcal{N}(\mu'_{\eta_k}, \mathbf{\Sigma'}_{\eta_k}). \tag{20}$$

4.3 Parameter Inference

At time $t-1$, the sufficient statistics for the parameter random variables (σ_k^2, $\mathbf{c}_{\mathbf{w_k}}$, θ_k) are (α, β, μ_c, $\mathbf{\Sigma}_c$, μ_θ, $\mathbf{\Sigma}_\theta$).

Let $\mathbf{z}_t = (\mathbf{x}_t^\intercal, (\mathbf{x}_t \odot \eta_{\mathbf{k,t}})^\intercal)^\intercal$, $\mathbf{\Sigma} = \begin{bmatrix} \mathbf{\Sigma_c} & \mathbf{0} \\ \mathbf{0} & \mathbf{\Sigma}_\theta \end{bmatrix}$, $\mu = (\mu_\mathbf{c}{}^\intercal, \mu_\theta{}^\intercal)^\intercal$, and $\nu_\mathbf{k} = (\mathbf{c}_{\mathbf{w_k}}{}^\intercal, \theta_\mathbf{k}{}^\intercal)^\intercal$ where $\mathbf{z}_t$, μ, and ν are $2d$-dimensional

vector, $\mathbf{\Sigma}$ is a $2d \times 2d$ -dimensional matrix. Therefore, the inference of $\mathbf{c_{w_k}}$ and θ_k is equivalent to infer ν_k with its distribution $\nu_k \sim \mathcal{N}(\mu, \sigma_k^2\mathbf{\Sigma})$. Assume $\mathbf{\Sigma}'$, μ', α', and β' be the sufficient statistics at time t which are updated based on the sufficient statistics at time $t-1$ and the new observation data. The sufficient statistics for parameters are updated as follows:

$$\begin{aligned}
\mathbf{\Sigma}' &= (\mathbf{\Sigma}^{-1} + \mathbf{z}_t\mathbf{z}_t^\intercal)^{-1}, \\
\mu' &= \mathbf{\Sigma}'(\mathbf{z}_t r_{k,t} + \mathbf{\Sigma}\mu), \\
\alpha' &= \alpha + \frac{1}{2}, \\
\beta' &= \beta + \frac{1}{2}(\mu^\intercal\mathbf{\Sigma}^{-1}\mu + r_{k,t}^2 - \mu'^\intercal\mathbf{\Sigma}'^{-1}\mu').
\end{aligned} \tag{21}$$

At time t, the sampling process for σ_k^2 and ν_k is summarized as follows:

$$\begin{aligned}
\sigma_k^2 &\sim \mathcal{IG}(\alpha', \beta'), \\
\nu_k &\sim \mathcal{N}(\mu', \sigma_k^2\mathbf{\Sigma}').
\end{aligned} \tag{22}$$

4.4 Integration with Policies

As discussed in Section 3.1, both LinUCB and Thompson sampling allocate the pulling chance based on the posterior distribution of $\mathbf{w}_k$ and σ_k^2 with the hyper parameters $\mu_\mathbf{w}$, $\mathbf{\Sigma}_\mathbf{w}$, α, and β.

As to the context drifting model, when $\mathbf{x}_t$ arrives at time t, the reward $r_{k,t}$ is unknown since it is not observed until one of arms is pulled. Without observed $r_{k,t}$, the particle re-sampling, latent state inference, and parameter inference for time t can not be conducted. Furthermore, every arm has p independent particles. Within each particle, the posterior distributions of $\mathbf{w}_{k,t-1}$ are not available since $\mathbf{w}_{k,t-1}$ has been decomposed into $\mathbf{c}_{\mathbf{w}_k}$, θ_k, and $\eta_{k,t-1}$ based on Equation 13. We address these issues as follows.

Within a single particle of arm $a^{(k)}$, the distribution of $\mathbf{w}_{k,t-1}$ can be derived by

$$\mathbf{w}_{k,t-1} \sim \mathcal{N}(\mu_{\mathbf{w_k}}, \sigma_k^2\mathbf{\Sigma}_{\mathbf{w_k}}), \tag{23}$$

where

$$\begin{aligned}
\mu_{\mathbf{w_k}} &= \mu_c + (\mathbf{\Sigma}_{\eta_k} + \sigma_k^2\mathbf{\Sigma}_\theta)^{-1}(\mathbf{\Sigma}_{\eta_k}\mu_\theta + \sigma_k^2\mathbf{\Sigma}_\theta\mu_{\eta_k}), \\
\mathbf{\Sigma}_{\mathbf{w_k}} &= \sigma_k^2\mathbf{\Sigma}_c + \sigma_k^2\mathbf{\Sigma}_\theta\mathbf{\Sigma}_{\eta_k}(\mathbf{\Sigma}_{\eta_k} + \sigma_k^2\mathbf{\Sigma}_\theta)^{-1}.
\end{aligned} \tag{24}$$

Let $\mathbf{w^{(i)}}_{k,t-1}$, $\mu^{(\mathbf{i})}_{\mathbf{w_k}}$, $\sigma^{2^{(i)}}_{k}$, and $\mathbf{\Sigma}^{(\mathbf{i})}_{\mathbf{w_k}}$ be the random variables in the $i^{(th)}$ particle. We use the mean of $\mathbf{w}_{k,t-1}$, denoted as $\bar{\mathbf{w}}_{k,t-1}$, to infer the decision in the bandit algorithm. Therefore,

$$\bar{\mathbf{w}}_{k,t-1} \sim \mathcal{N}(\bar{\mu}_{\mathbf{w_k}}, \bar{\mathbf{\Sigma}}_{\mathbf{w_k}}), \tag{25}$$

where

$$\begin{aligned}
\bar{\mu}_{\mathbf{w_k}} &= \frac{1}{p}\sum_{i=1}^{p}\mu^{(\mathbf{i})}_{\mathbf{w_k}}, \\
\bar{\mathbf{\Sigma}}_{\mathbf{w_k}} &= \frac{1}{p^2}\sum_{i=1}^{p}\sigma^{2^{(i)}}_{k}\mathbf{\Sigma}^{(\mathbf{i})}_{\mathbf{w_k}}.
\end{aligned} \tag{26}$$

By virtual of Equation 25, both Thompson sampling and LinUCB can address the bandit problem as mentioned in Section 3.1. Specifically, Thompson sampling draws $\mathbf{w}_{k,t}$ from Equation 25 and then predicts the reward for each arm with $\mathbf{w}_{k,t}$. The arm with maximum predicted reward is selected to pull. While LinUCB selects arm with a maximum score, where the score is defined as a combination of the expectation of $y_{k,t}$ and its standard deviation, i.e.,

$$E(y_{k,t}|\mathbf{x}_t) + \lambda\sqrt{Var(y_{k,t}|\mathbf{x}_t)},$$

where λ is predefined parameter, $E(y_{k,t}|\mathbf{x}_t)$ and $Var(y_{k,t}|\mathbf{x}_t)$ are computed by

$$E(y_{k,t}|\mathbf{x}_t) = \mathbf{x}_t^\intercal\mathbf{w}_{k,t}.$$

$$Var(y_{k,t}|\mathbf{x}_t) = \mathbf{x}_t^\intercal\bar{\mathbf{\Sigma}}^{-\mathbf{1}}_{\mathbf{w_k}}\mathbf{x}_t + \frac{1}{p^2}\sum_{i=1}^{p}\sigma_k^2.$$

4.5 Algorithm

Putting all the aforementioned things together, an algorithm based on the context drifting model is provided below.

Algorithm 1 The algorithm for context drift model (`Drift`)

procedure MAIN(p) ▷ main entry
  Initialize arms with p particles.
  for $t \leftarrow 1, T$ **do**
    Get $\mathbf{x}_t$.
    $a^{(k)} = \arg\max_{j=1,K}$ EVAL$(a^{(j)}, \mathbf{x}_t)$
    Receive $r_{k,t}$ by pulling arm $a^{(k)}$.
    UPDATE($\mathbf{x}_t$, $a^{(k)}$, $r_{k,t}$).
  end for
end procedure

procedure EVAL($a^{(k)}$, $\mathbf{x}_t$) ▷ get a score for $a^{(k)}$, given $\mathbf{x}_t$.
  Learn the parameters based on all particles' inferences of $a^{(k)}$ by Equation 25.
  Compute a score based on the parameters learnt.
  return the score.
end procedure

procedure UPDATE($\mathbf{x}_t$, $a^{(k)}$, $r_{k,t}$) ▷ update the inference.
  for $i \leftarrow 1, p$ **do** ▷ Compute weights for each particle.
    Compute weight $\rho^{(i)}$ of particle $\mathcal{P}_k^{(i)}$ by Equation 17.
  end for
  Re-sample $\mathcal{P}'_k$ from $\mathcal{P}$ according to the weights $\rho^{(i)}$s.
  for $i \leftarrow 1, p$ **do** ▷ Update statistics for each particle.
    Update the sufficient statistics for $\eta_{k,t}$ by Equation 19.
    Sample $\eta_{k,t}$ according to Equation 20.
    Update the statistics for σ_k^2, $\mathbf{c}_{\mathbf{w}_k}$, θ_k by Equation 21.
    Sample σ_k^2, $\mathbf{c}_{\mathbf{w}_k}$, θ_k according to Equation 22.
  end for
end procedure

Online inference for contextual multi-armed bandit problem starts with MAIN procedure, as presented in Algorithm 1. As $\mathbf{x}_t$ arrives at time t, the EVAL procedure computes a score for each arm, where the definition of score depends on the specific policy. The arm with the highest score is selected to pull. After receiving a reward by pulling an arm, the new feedback is used to update the contextual drifting model by the UPDATE procedure. Especially in the UPDATE procedure, we use the $resample\text{-}propagate$ strategy in particle learning [6] rather than the $propagate\text{-}resample$ strategy in particle filtering [10]. With the $resample\text{-}propagate$ strategy, the particles are re-sampled by taking $\rho^{(i)}$ as the i^{th} particle's weight, where the $\rho^{(i)}$ indicates the occurring probability of 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.

5. EMPIRICAL STUDY

To demonstrate the efficacy of our proposed algorithm, we conduct our experimental study over two real-world data sets including

the online search advertising data from Track 2 of KDD Cup 2012, and the news recommendation data of Yahoo! Today News. Before diving into the detail of the experiment on each data set, we first outline the general implementation of the baseline algorithms for comparison. Second, we start with a brief description of the data sets and their corresponding evaluation methods. We finally show and discuss the comparative experimental results of both the proposed algorithm and the baseline algorithms.

5.1 Baseline Algorithms

In the experiment, we demonstrate the performance of our method by comparing with the following algorithms. The baseline algorithms include:

1. `Random`: it randomly selects an arm to pull without considering any contextual information.

2. ϵ-`greedy`(ϵ) (or *EPSgreedy*): it randomly selects an arm with probability ϵ and selects the arm of the largest predicted reward with probability $1-\epsilon$, where ϵ is a predefined parameter. When $\epsilon = 0$, it is equivalent to the `Exploit` policy.

3. `GenUCB`(λ): it denotes the general `UCB` algorithm for contextual bandit problems. It can be integrated with linear regression model(e.g.,`LinUCB` [16]) or logistic regression model (e.g., `LogUCB` [19]) for reward prediction. Both `LinUCB` and `LogUCB` take the parameter λ to obtain a score defined as a linear combination of the expectation and the deviation of the reward. When $\lambda = 0$, it becomes the `Exploit` policy that has no exploration.

4. `TS`(q_0): Thompson sampling described in Section 3.1, randomly draws the coefficients from the posterior distribution, and selects the arm of the largest predicted reward. The priori distribution is $\mathcal{N}(\mathbf{0}, q_0^{-1}\mathbf{I})$.

5. `TSNR`(q_0): it is similar to `TS`(q_0), but in the stochastic gradient ascent, there is no regularization by the prior. The priori distribution $\mathcal{N}(\mathbf{0}, q_0^{-1}\mathbf{I})$ is only used in the calculation of the posterior distribution for the parameter sampling, but not in the stochastic gradient ascent. When q_0 is arbitrarily large, the variance approaches 0 and `TSNR` becomes `Exploit`.

6. `Bootstrap`: it is non-Bayesian but an ensemble method for arm selection. Basically, it maintains a set of bootstrap samples for each arm and randomly pick one bootstrap sample for inference [25].

Our methods proposed in this paper include:

1. `TVUCB`(λ): it denotes the time varying `UCB` which integrates our proposed context drift model with `UCB` bandit algorithm. Similar to `LinUCB`, the parameter λ is given.

2. `TVTP`(q_0): it denotes the time varying `Thompson sampling` algorithm which is extended with our proposed context drift model and the algorithm is outlined in Algorithm 1. The parameter q_0, similar to `TS`(q_0), specifies the prior distribution of the coefficients.

5.2 KDD Cup 2012 Online Advertising

5.2.1 Description

Online advertising has become one of the major revenue sources of the Internet industry for many years. In order to maximize the Click-Though Rate (CTR) of displayed advertisements (ads), online advertising systems need to deliver these appropriate ads to individual users. Given the context information, sponsored search which is one type of online advertising will display a recommended ad in the search result page. Practically, an enormous amount of new ads will be continuously brought into the ad pool. These new ads have to be displayed to users, and feedbacks have to be collected for improving the system's CTR prediction. Thereby, the problem of ad recommendation can be regarded as an instance of contextual bandit problem. In this problem, an arm is an ad, a pull is an ad impression for a search activity, the context is the information vector of user profile and search keywords, and the reward is the feedbacks of user's click on ads.

The experimental dataset is collected by a search engine and published by KDD Cup 2012[2]. In this dataset, each instance refers to an interaction between a user and the search engine. It is an ad impression, which consists of the user demographic information (age and gender), query keywords, some ads information returned by the search engine and click count on ads. In our work, the context is represented as a binary feature vector of dimension 1,070,866, including query entry and user's profile information. And each query entry denotes whether a query token is contained in the search query or not. In the experiments, we use 1 million user visit events.

5.2.2 Evaluation Method

We use a simulation method to evaluate the KDD Cup 2012 online ads data, which is applied in [8] as well. The simulation and *replayer* [17] are two of the frequently used methods for the bandit problem evaluation. As discussed in [8] and [25], the simulation method performs better than *replayer* method when the item pool contains a large number of recommending items, especially larger than 50. The large number of recommending items leads to the CTR estimation with a large variance due to the small number of matched visits.

In this data set, we build our ads pool by randomly selecting $K = 100$ ads from the entire set of ads. There is no explicit time stamp associated with each ad impression, and we assume the ad impression arrives in chronological order with a single time unit interval between two adjacent impressions. The context information of these ads are real and obtained from the given data set. However, the reward of the k^{th} ad is simulated with a coefficient vector $\mathbf{w}_{k,t}$, which dynamically changes over time. Let ϱ be the change probability, where each coefficient keeps unchanged with probability $1-\varrho$ and varies dynamically with probability ϱ. We model the dynamical change as a Gaussian random walk by $\mathbf{w}_{k,t} = \mathbf{w}_{k,t} + \Delta_w$ where Δ_w follows the standard Gaussian distribution, i.e., $\Delta_w \sim \mathcal{N}(\mathbf{0}, \mathcal{I}_d)$. Given a context vector $\mathbf{x}_t$ at time t, the click of the k^{th} ad is generated with a probability $(1+\exp(-\mathbf{w}_{k,t}^T\mathbf{x}_t))^{-1}$. For each user visit and each arm, the initial weight vector $\mathbf{w}_{k,0}$ is drawn from a fixed normal distribution that is randomly generated before the evaluation.

5.2.3 Context Change Tracking

With the help of the simulation method, we get a chance to know the ground truth of the coefficients. Therefore, we first explore the fitness of our model with respect to the true coefficient values over time. Then we conduct our experiment over the whole online ads data set containing 1 million impressions by using the CTR as the evaluation metric.

We simulate the dynamical change of coefficients in multiple different ways including the random walk over a small segment of data set shown in Figure 3. Sampling a segment of data containing $120k$ impressions from the whole data set, we assume a dynamical change occurring on only one dimension of the coefficient vector,

[2]http:/www. kddcup2012.org/c/kddcup2012-track2

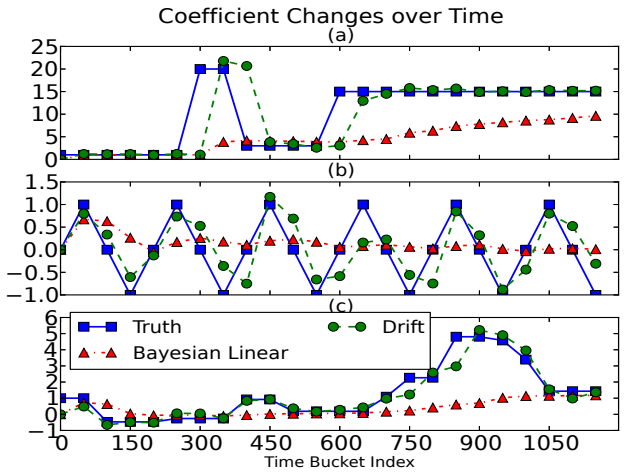


Figure 3: A segment of data originated from the whole data set is provided. The reward is simulated by choosing one dimension of the coefficient vector, which is assumed to vary over time in three different ways. Each time bucket contains 100 time units.

keeping other dimensions constant. In (a), we divide the whole segment of data into four intervals, where each has a different coefficient value. In (b), we assume the coefficient value of the dimension changes periodically. In (c), a random walk mentioned above is assumed, where $\varrho = 0.0001$. We compare our algorithm `Drift` with the bandit algorithm such as `LinUCB` with Bayesian linear regression for reward prediction. We set `Drift` with 5 particles. It shows that our algorithm can fit the coefficients better than Bayesian linear regression and can adaptively capture the dynamical change instantly. The reason is that, `Drift` has a random walk for each particle at each time and estimates the coefficient by re-sampling these particles according to their goodness of fitting.

5.2.4 CTR Optimization for Online ADS

In this section, we evaluate our algorithm over the online ads data in terms of CTR. The performance of each baseline algorithm listed in Section 5.1 depends on the underlying reward prediction model (e.g., logistic regression, linear regression) and its corresponding parameters. Therefore, we first conduct the performance comparison for each algorithm with different reward prediction models and diverse parameter settings. Then the one with best performance is selected to compare with our proposed algorithm. The experimental result is presented in Figure 4. The algorithm `LogBoostrap(10)` achieves better performance than `LinBootstrap(10)` since our simulation method is based on the *Logit* function.

Although our algorithms `TVTP(1)` and `TVUCB(1)` are based on linear regression model, they can still achieve high CTRs and their performance is comparable to those algorithms based on logistic regression method such as, `LogTS(0.001)`,`LogTSnr(10)`. The reason is that both `TVTP` and `TVUCB` are capable of capturing the non-linear reward mapping function by explicitly considering the context drift. The algorithm `LogEpsGreedy(0.5)` does not perform well. The reason is that the value of parameter ϵ is large, incurring lots of exploration.

5.3 Yahoo! Today News

5.3.1 Description

The core task of personalized news recommendation is to display

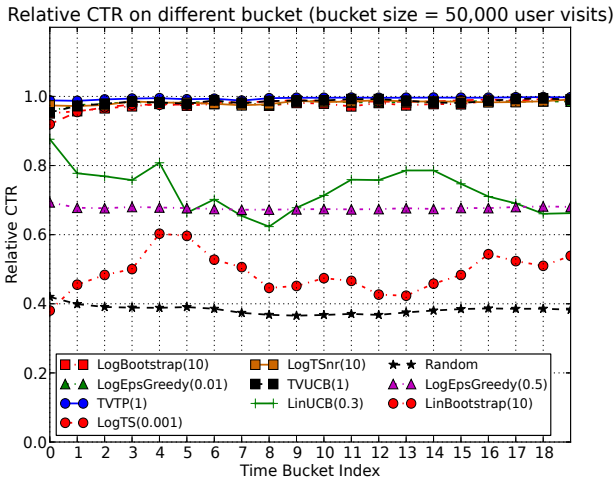


Figure 4: The CTR of KDD CUP 2012 online ads data is given for each time bucket. `LogBooststrap`, `LogTS`, `LogTSnr`, and `LogEpsGreedy` are bandit algorithms with logistic regression model. `LinUCB`, `LinBoostrap`, `TVTP`, and `TVUCB` are based on linear regression model.

appropriate news articles on the web page for the users according to the potential interests of individuals. However, it is difficult to track the dynamical interests of users only based on the content. Therefore, the recommender system often takes the instant feedbacks from users into account to improve the prediction of the potential interests of individuals, where the user feedbacks are about whether the users click the recommended article or not. Additionally, every news article does not receive any feedbacks unless the news article is displayed to the user. Accordingly, we formulate the personalized news recommendation problem as an instance of contextual multi-arm bandit problem, where each arm corresponds to a news article and the contextual information including both content and user information.

The experimental data set is a collection based on a sample of anonymized user interaction on the news feeds, collected by Yahoo! Today module and published by Yahoo! research lab[3]. The dataset contains 28,041,015 visit events of user-news item interaction data, collected by the Today Module from October 2nd, 2011 to October 16th, 2011 on Yahoo! Front Page. In addition to the interaction data, user's information, e.g., demographic information (age and gender), behavior targeting features, etc., is provided for each visit event, and represented as a binary feature vector of dimension 136. Besides, the interaction data is also stamped with the user's local time, which is suitable for contextual recommendation and temporal data mining. This data set has been used for evaluating contextual bandit algorithms in[16][8][17]. In our experiments, 2.5 million user visit events are used.

5.3.2 Evaluation Method

We apply the *replayer* method to evaluate our proposal method on the news data collection since the number of articles in the pool is not larger than 50. The *replayer* method is first introduced in [17], which provides an unbiased offline evaluation via the historical logs. The main idea of *replayer* is to replay each user visit to the algorithm under evaluation. If the recommended article by the testing algorithm is identical to the one in the historical log, this visit is considered as an impression of this article to the user. The ratio

[3]http://webscope.sandbox.yahoo.com/catalog.php

Table 2: Relative CTR on Yahoo! News Data.

Algorithm	Logistic Regression				Linear Regression			
	mean	std	min	max	mean	std	min	max
ϵ-greedy(0.01)	**0.0644**	0.00246	0.0601	0.0685	0.0554	0.00658	0.0374	0.0614
ϵ-greedy(0.1)	0.0633	0.00175	0.0614	0.0665	0.0626	0.00127	0.0599	0.0643
ϵ-greedy(0.3)	0.0563	0.00129	0.0543	0.0588	0.0583	0.00096	0.0564	0.0595
ϵ-greedy(0.5)	0.0491	0.00118	0.0471	0.0512	0.0522	0.00057	0.0514	0.0533
Bootstrap(1)	0.0605	0.00427	0.0518	0.0683	0.0389	0.01283	0.0194	0.0583
Bootstrap(5)	0.0615	0.00290	0.0578	0.0670	0.0400	0.01089	0.0194	0.0543
Bootstrap(10)	**0.0646**	0.00169	0.0611	0.0670	0.0448	0.00975	0.0216	0.0571
Bootstrap(30)	0.0644	0.00161	0.0612	0.0667	0.0429	0.01036	0.0226	0.0599
LinUCB(0.01)	0.0597	0.00184	0.0572	0.0633	0.0423	0.00912	0.0325	0.0608
LinUCB(0.1)	0.0444	0.00054	0.0434	0.0454	0.0612	0.00205	0.0561	0.0630
LinUCB(0.3)	0.0419	0.00047	0.0413	0.0429	0.0701	0.00132	0.0669	0.0712
LinUCB(0.5)	0.0410	0.00044	0.0402	0.0416	**0.0702**	0.00041	0.0693	0.0707
LinUCB(1.0)	0.0402	0.00055	0.0392	0.0411	0.0668	0.00035	0.0661	0.0673
TS(0.001)	0.0453	0.00050	0.0445	0.0463	0.0431	0.00373	0.0401	0.0536
TS(0.01)	0.0431	0.00074	0.0420	0.0448	0.0526	0.00188	0.0489	0.0548
TS(0.1)	0.0416	0.00081	0.0401	0.0433	0.0594	0.00155	0.0551	0.0606
TS(1.0)	0.0397	0.00040	0.0391	0.0404	**0.0597**	0.00070	0.0585	0.0607
TS(10.0)	0.0325	0.00833	0.0180	0.0432	0.0592	0.00071	0.0577	0.0603
TSNR(0.01)	0.0445	0.00052	0.0433	0.0454	0.0596	0.00040	0.0591	0.0605
TSNR(0.1)	0.0449	0.00066	0.0441	0.0463	0.0592	0.00084	0.0577	0.0605
TSNR(1.0)	0.0468	0.00071	0.0456	0.0479	0.0596	0.00069	0.0585	0.0606
TSNR(10.0)	0.0594	0.00168	0.0573	0.0619	0.0605	0.00053	0.0594	0.0614
TSNR(100.0)	**0.0643**	0.00293	0.0592	0.0679	0.0586	0.00201	0.0555	0.0614
TSNR(1000.0)	0.0641	0.00222	0.0609	0.0690	0.0535	0.00345	0.0482	0.0606
Parameter	**TVUCB**				**Parameter**	**TVTP**		
$\lambda = 0.01$	0.0427	0.0122	0.0278	0.0623	$q_0 = 0.001$	0.0614	0.00139	0.0592	0.0644
$\lambda = 0.1$	0.0606	0.0038	0.0520	0.0651	$q_0 = 0.01$	0.0648	0.00135	0.0611	0.0661
$\lambda = 0.3$	0.0643	0.0023	0.0585	0.0676	$q_0 = 0.1$	0.0652	0.00091	0.06339	0.06655
$\lambda = 0.5$	**0.0705***	0.0017	0.0689	0.0715	$q_0 = 1.0$	**0.0656**	0.0012	0.0638	0.0669
$\lambda = 1.0$	0.06824	0.0024	0.0655	0.0714	$q_0 = 10$	0.0624	0.0016	0.05938	0.0643

between the number of user clicks and the number of impressions is referred as CTR. The work in [17] shows that the CTR estimated by the *replayer* method approaches the real CTR of the deployed online system if the items in historical user visits are randomly recommended.

5.3.3 CTR Optimization for News Recommendation

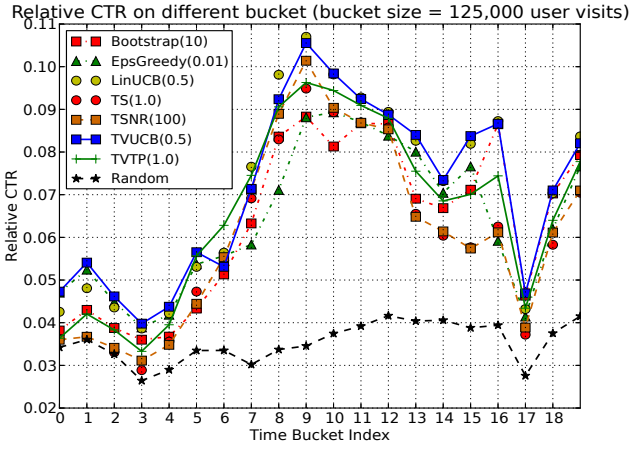


Figure 5: The CTR of Yahoo! News data is given for each time bucket. Those baseline algorithms are configured with their best parameters settings.

Similar to the CTR optimization for online ads data in Section 5.2.4, we first conduct the performance evaluation for each algorithm with different regression models and parameter settings. The experimental result is displayed in Table 2. The setting of each algorithm with the highest reward is highlighted in bold. It can be observed that our algorithm TVUCB(0.5) achieves the best performance among all algorithms. In four of all five parameter λ settings, the performances of TVUCB consistently exceed the ones of LinUCB.

All baseline algorithms are configured with their best parameter settings provided by Table 2. We conduct the performance comparison on different time buckets in Figure 5. The algorithm TVUCB(0.5) and EpsGreedy(0.01) outperforms others among the first four buckets, known as cold-start phrase when the algorithms are not trained with sufficient observations. After the fourth bucket, the performance of both TVUCB(0.5) and LinUCB(0.5) constantly exceeds the ones of other algorithms. In general, TVTP(1.0) performs better than TS(1.0) and TSNR(100), where all the three algorithms are based on the Thompson sampling. Overall, TVUCB(0.5) consistently achieves the best performance.

5.4 Time Cost

The time cost for TVUCB and TVTP on both two data sets are displayed in Figure 6. It shows that the time costs are increased linearly with the number of particles. In general, TVUCB is faster than TVTP since TVTP highly depends on the sampling process.

6. CONCLUSIONS

In this paper, we take the dynamic behavior of reward into account and explicitly model the context drift as a random walk. We

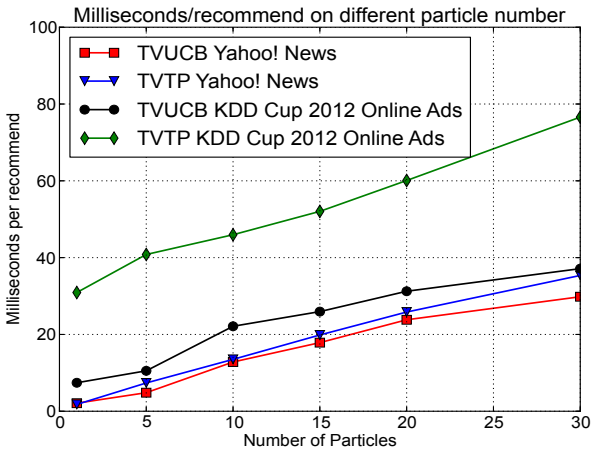


Figure 6: The time costs on different numbers of particles are given for both two data collections.

propose a method based on the particle learning to efficiently infer both parameters and latent drift of the context. Integrated with existing bandit algorithms, our model is capable of tracking the contextual dynamics and consequently improve the performance of personalized recommendation in terms of CTRs, which is verified in two real applications,i.e., online advertising and news recommendation.

The recommend items, e.g., advertisements or news articles, may have some underlying relations with each other. For example, two advertisements may belong to the same categories, or come from business competitors, or have other same features. In the future, we plan to consider the potential correlations among different items, or say, arms. It is interesting to model these correlations as constraints, and incorporate them into the contextual bandit modeling process. Moreover, the dynamically changing behaviors of two correlated arms tend to be correlated with a time lag, where the change correlation can be interpreted as an event temporal pattern [27]. Therefore, another possible research direction is to extend our time varying model considering the correlated change behaviors with the time lag.

7. ACKNOWLEDGMENTS

The work was supported in part by the National Science Foundation under grants CNS-1126619, IIS-1213026, and CNS-1461926, the U.S. Department of Homeland Securitys VACCINE Center under Award Number 2009-ST-061-CI0001, and a gift award from Huawei Technologies Co. Ltd.

8. REFERENCES

[1] D. Agarwal, B.-C. Chen, and P. Ela. Explore/exploit schemes for web content optimization. In *Data Mining, 2009. ICDM'09. Ninth IEEE International Conference on*, pages 1–10. IEEE, 2009.

[2] S. Agrawal and N. Goyal. Thompson sampling for contextual bandits with linear payoffs. In *Proceedings of The 30th International Conference on Machine Learning*, pages 127–135, 2013.

[3] P. Auer, N. Cesa-Bianchi, Y. Freund, and R. E. Schapire. The nonstochastic multiarmed bandit problem. *SIAM Journal on Computing*, 32(1):48–77, 2002.

[4] B. Awerbuch and R. Kleinberg. Online linear optimization and adaptive routing. *Journal of Computer and System Sciences*, 74(1):97–114, 2008.

[5] D. Bouneffouf, A. Bouzeghoub, and A. L. Gançarski. A contextual-bandit algorithm for mobile context-aware recommender system. In *Neural Information Processing*, pages 324–331. Springer, 2012.

[6] C. Carvalho, M. S. Johannes, H. F. Lopes, and N. Polson. Particle learning and smoothing. *Statistical Science*, 25(1):88–106, 2010.

[7] S. Chang, J. Zhou, P. Chubak, J. Hu, and T. S. Huang. A space alignment method for cold-start tv show recommendations. In *Proceedings of the 24th International Conference on Artificial Intelligence*, pages 3373–3379. AAAI Press, 2015.

[8] O. Chapelle and L. Li. An empirical evaluation of thompson sampling. In *NIPS*, pages 2249–2257, 2011.

[9] W. Chu and S.-T. Park. Personalized recommendation on dynamic content using predictive bilinear models. In *Proceedings of the 18th international conference on World wide web*, pages 691–700. ACM, 2009.

[10] P. M. Djurić, J. H. Kotecha, J. Zhang, Y. Huang, T. Ghirmai, M. F. Bugallo, and J. Miguez. Particle filtering. *Signal Processing Magazine, IEEE*, 20(5):19–38, 2003.

[11] A. Doucet, S. Godsill, and C. Andrieu. On sequential monte carlo sampling methods for bayesian filtering. *Statistics and computing*, 10(3):197–208, 2000.

[12] J. H. Halton. Sequential monte carlo. In *Mathematical Proceedings of the Cambridge Philosophical Society*, volume 58, pages 57–78. Cambridge Univ Press, 1962.

[13] A. C. Harvey. *Forecasting, structural time series models and the Kalman filter*. Cambridge university press, 1990.

[14] R. Kleinberg, A. Slivkins, and E. Upfal. Multi-armed bandits in metric spaces. In *Proceedings of the fortieth annual ACM symposium on Theory of computing*, pages 681–690. ACM, 2008.

[15] J. Langford and T. Zhang. The epoch-greedy algorithm for multi-armed bandits with side information. In *NIPS*, 2007.

[16] L. Li, W. Chu, J. Langford, and R. E. Schapire. A contextual-bandit approach to personalized news article recommendation. In *Proceedings of the 19th international conference on World wide web*, pages 661–670. ACM, 2010.

[17] L. Li, W. Chu, J. Langford, and X. Wang. Unbiased offline evaluation of contextual-bandit-based news article recommendation algorithms. In *Proceedings of the fourth ACM international conference on Web search and data mining*, pages 297–306. ACM, 2011.

[18] L. Li, L. Zheng, F. Yang, and T. Li. Modeling and broadening temporal user interest in personalized news recommendation. *Expert Systems with Applications*, 41(7):3168–3177, 2014.

[19] D. K. Mahajan, R. Rastogi, C. Tiwari, and A. Mitra. Logucb: an explore-exploit algorithm for comments recommendation. In *Proceedings of the 21st ACM international conference on Information and knowledge management*, pages 6–15. ACM, 2012.

[20] S. Pandey and C. Olston. Handling advertisements of unknown quality in search advertising. In *Advances in neural information processing systems*, pages 1065–1072, 2006.

[21] F. Radlinski, R. Kleinberg, and T. Joachims. Learning diverse rankings with multi-armed bandits. In *Proceedings of the 25th international conference on Machine learning*, pages 784–791. ACM, 2008.

[22] H. Robbins. Some aspects of the sequential design of experiments. In *Herbert Robbins Selected Papers*, pages 169–177. Springer, 1985.

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

[24] L. Tang, Y. Jiang, L. Li, and T. Li. Ensemble contextual bandits for personalized recommendation. In *Proceedings of the 8th ACM Conference on Recommender Systems*, pages 73–80. ACM, 2014.

[25] L. Tang, Y. Jiang, L. Li, C. Zeng, and T. Li. Personalized recommendation via parameter-free contextual bandits. In *Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 323–332. ACM, 2015.

[26] M. Tokic. Adaptive ε-greedy exploration in reinforcement learning based on value differences. In *KI 2010: Advances in Artificial Intelligence*, pages 203–210. Springer, 2010.

[27] C. Zeng, L. Tang, T. Li, L. Shwartz, and G. Y. Grabarnik. Mining temporal lag from fluctuating events for correlation and root cause analysis. In *Network and Service Management (CNSM), 2014 10th International Conference on*, pages 19–27. IEEE, 2014.