A Bayesian-DLM-CF Framework for Real-Time Display Advertising

Michael Els, InMarket, mels@inmarket.com
David Banks, Duke University, banks@stat.duke.edu

ABSTRACT

Click-through rate prediction underpins real-time bidding strategies in display advertising. We propose a unified approach that integrates beta-based Bayesian priors, Dynamic Linear Models, and collaborative filtering to address data sparsity, temporal dynamics, and neighbor relationships. A hierarchical Bayesian structure shares information across campaigns from the same advertiser, improving estimates when per-campaign data are limited. On a real-world dataset, our method outperforms baselines including standard collaborative filtering, random forest, and XGBoost, achieving superior log-loss and mean squared error.

ACM Reference Format:
Michael Els, InMarket, mels@inmarket.com and David Banks, Duke University, banks@stat.duke.edu. 2025. A Bayesian-DLM-CF Framework for Real-Time Display Advertising. In *Proceedings of Proceedings of the ACM SIGKDD Conference (SIGKDD '25).* ACM, New York, NY, USA, 6 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION

Predicting click-through rate (CTR) is a cornerstone of real-time bidding (RTB) systems in online advertising. Small improvements in prediction accuracy can significantly increase the success of ad campaigns, as advertisers and platforms seek to optimize both user engagement and revenue generation [10]. Despite the steady advance of machine learning methods for CTR, several challenges persist. First, human behavior exhibits considerable variation over time, leading to temporal shifts that degrade the performance of static models. Second, the sheer number of possible (person, item) pairs, combined with typically rare click data, creates serious data sparsity issues that can undermine a model's ability to generalize. Finally, real-time environments demand fast and reliable predictions at scale, placing practical constraints on the complexity and latency of CTR models.

Past approaches have tackled these challenges from different angles. Logistic regression and related generalized linear models gave interpretable baselines [10], but struggled with large, high-dimensional feature spaces. Factorization machines and deep learning methods incorporate feature interactions more effectively [4, 9], but require substantial data and hyperparameter tuning to be robust in sparse regimes. Gradient boosting frameworks such as XGBoost [1] can efficiently handle non-linearities and missing data, but they typically lack a built-in mechanism for temporal adaptation. Moreover, collaborative filtering (CF) has proven valuable in recommendation scenarios by leveraging user–item similarities, but the method falters when temporal trends or severe data sparsity are present [6].

In this paper, we propose an integrated solution that unifies Bayesian modeling, Dynamic Linear Models (DLMs), and collaborative filtering for CTR estimation. Our method uses a beta prior to capture uncertainty in sparse data within a DLM framework to model evolving CTR over time, then refines these posterior estimates using item-based collaborative filtering. We also introduce a hierarchical Bayesian component that shares information across related campaigns (e.g., those from the same advertiser), thereby mitigating cold-start problems and increasing accuracy. By systematically merging these elements, our approach reduces the adverse effects of data sparsity, adapts to temporal changes, and captures structural relationships between items and campaigns.

We evaluate our method on a large-scale dataset from a production RTB environment and benchmark it against well-established baselines: vanilla collaborative filtering, random forests, and XGBoost. Experimental results show that our framework achieves state-of-the-art performance in terms of log-loss and mean-squared error, outperforming alternatives by a considerable margin. Further analysis reveals that each component—such as the beta prior, the DLM smoothing, and the collaborative filtering step—contributes distinct benefits to the final model. Crucially, our pipeline remains computationally feasible for practical deployment, as both the DLM and CF steps are highly parallel and the Bayesian updates rely on conjugate priors, enabling efficient runtimes even under real-time requirements and the non-stationary nature of the online advertising ecosystem.

2 RELATED WORK

CTR prediction lies at the core of many recommender systems and RTB platforms, driving advertising revenue and user satisfaction. A rich body of literature has explored methods ranging from classical regression techniques to sophisticated deep learning architectures, each addressing different aspects of sparsity, feature representation, or temporal adaptation. In this section, we review representative advances in CTR modeling, emphasizing both established baselines and recent progress toward integrated, adaptive models.

Generalized Linear and Tree-Based Methods. Early CTR optimization systems often relied on logistic regression (LR) for CTR estimation due to its interpretability and ability to incorporate large numbers of sparse features [10]. Meanwhile, tree-based ensemble methods gained traction by capturing non-linear feature interactions [5]. Approaches like Gradient Boosting Machines (GBMs) and XGBoost [1] streamlined training and inference, demonstrating

SIGKDD '25, August 3-7, 2025, Toronto, Canada

ACM ISBN 978-x-xxxx-xxxx-x/YY/MM
https://doi.org/10.1145/nnnnnnn.nnnnnnn

strong performance on public benchmarks and real-world production logs. Although effective in handling complex feature patterns, these methods commonly treat each observation as temporally independent, relying on feature engineering or retraining for any time-varying behavior.

Factorization and Deep Learning Approaches. Factorization Machines (FMs) [9] extended linear models by explicitly modeling pairwise feature interactions with linear complexity. Deep learning then pushed this idea further, using embedding layers for sparse categorical features and neural networks for complex, high-order interactions. Models such as DeepFM [4], Wide&Deep [2], and DIN [13] found some success by automatically learning salient feature interactions and user interest representations from massive datasets. Nevertheless, these approaches assume stationarity or rely on periodically refreshed models, leaving them vulnerable to performance degradation when user preferences change.

Collaborative Filtering and Matrix Factorization. Borrowing from recommendation systems, CF posits that users with similar historical engagement patterns exhibit similar future behavior [7]. Matrix factorization techniques decompose the (user, item) ratings matrix into latent factors, effectively handling data sparsity [8]. Follow-up research introduced temporal extensions (TimeSVD++), capturing evolving user and item dynamics [6]. However, conventional CF methods often assume relatively stable preference structures, lack explicit uncertainty quantification and require significant modifications for application in RTB contexts.

Bayesian Modeling and Hierarchical Structures. Bayesian methods handle data sparsity by introducing priors that capture parameter uncertainty. In CTR prediction, hierarchical Bayesian frameworks can pool statistical strength across related entities such as advertisers or product categories [3]. Gamma priors for precision or beta priors for click-rate parameters have proven effective in preventing overfitting in sparse regimes. Despite these advantages, purely Bayesian or hierarchical methods can struggle with large-scale temporal data unless carefully optimized.

Temporal Adaptation via State-Space Models. Time-varying models incorporate sequential structure to adapt to user preference drift. Kalman filtering and Dynamic Linear Models (DLMs) [11] provide a probabilistic mechanism for updating beliefs about latent states over time. In online advertising, they can track CTR fluctuations driven by seasonal or campaign-related factors. However, DLMs alone do not fully exploit cross-campaign or cross-item signals, nor directly address unbalanced data when certain user-item pairs are underrepresented.

Reinforcement Learning and Bandit Perspectives. Recent work interprets ad allocation as a sequential decision-making problem, framing CTR prediction within multi-armed bandit or reinforcement learning (RL) paradigms [12]. These approaches dynamically update bidding strategies to maximize cumulative reward (e.g., clicks or conversions), often modeling the underlying user response stochastically. While RL-based methods can excel at balancing exploration and exploitation, they typically require a reliable reward model and careful sample-efficiency tuning, which is challenging in extremely sparse settings.

Our Contributions in Context. In contrast to purely static ML models, purely Bayesian frameworks, or purely RL-based strategies, our work bridges multiple elements to tackle real-world CTR estimation holistically:

- We adopt a *beta-based prior* to handle item-level or user-level data sparsity, ensuring robust uncertainty estimates.
- We embed *DLM updates* for smoothing temporal fluctuations, reducing the cost of frequent retraining.
- We refine these posterior estimates using *collaborative filtering*, exploiting cross-campaign or cross-item relationships.
- We incorporate a *hierarchical Bayesian model* to share information across related campaigns (e.g., from the same advertiser), mitigating cold-start and rare-event scenarios.

As a result, our approach not only captures evolving CTR dynamics but also addresses data scarcity and latent structure in large-scale RTB environments. Our experimental evaluation demonstrates superior performance against strong baselines, enabling practical use in real-world online advertising systems where speed, scalability, and adaptability are essential.

3 METHODOLOGY

This section describes our unified approach to CTR estimation, integrating beta-based Bayesian priors, a Kalman-filtered DLM, collaborative filtering (CF), and a hierarchical Bayesian framework. We begin with a high-level outline of how these components interact, then detail each step in the subsequent subsections. Algorithm 1 defines the setup and calls Algorithm 2 for the optimization loop.

3.1 Overview

We aim to estimate the click-through rate (CTR) for a given campaign, represented by user u, and an ad opportunity, represented by item i, at time t. Here, our units of analysis are campaigns and ad opportunities, but we adopt the conventional user-item notation prevalent in collaborative filtering literature. Our procedure operates offline in batches, and the final CTR matrix is retrieved in real-time bidding. Concretely:

(1) **Beta Hierarchical Model:** Each user–item pair has beta-based parameters, with hyperpriors across advertisers (or other groupings) to stabilize estimates in sparse scenarios.
(2) **DLM (Kalman Filter) Fit:** A Gaussian DLM is run separately to produce a posterior CTR for each (u, i) at each time t, capturing temporal changes day by day.
(3) **CF on the DLM Posterior:** We treat the DLM-updated CTR matrix as input to item-based CF. Neighbor similarities (scaled by impressions; i.e., the number of times the ad has been shown) refine each posterior estimate, partially blending in the collaborative signals.
(4) **Global Mixture with Beta:** Finally, a single global mixing parameter combines the beta posterior mean for each pair (u, i) with the CF-refined DLM estimate. This ensures that beta priors remain influential for pairs with less data, while the DLM and CF steps drive estimates where observational evidence is stronger.

Although each step can be updated efficiently with local or gradient-based methods, the presence of collaborative filtering and

Algorithm 1 Beta+DLM+CF Pipeline: Initialization and Deployment

Require: *Historical logs* $(c_{u,i,t}, n_{u,i,t})$ over time; *initial global priors* (α_0, β_0) from system logs; *hierarchical structure* (advertiser-level).

Ensure: *CTR estimates* $\widetilde{x}_{u,i,t}$ for real-time bidding.

(A) Calibrate Process Noise:
  Learn optimal Q that minimizes $\mathcal{L}_{\mathrm{KF}}$.
(B) Offline Iterative Updates Until Convergence (Algorithm 2):
  Run the repeated steps (Beta priors, DLM, CF, hierarchical updates, calibration) in Algorithm 2, monitoring changes in $(w_\alpha, w_\beta, \phi)$ and overall loss.
  Return Final CTR estimates $\widetilde{x}_{u,i,t}$ for each (u, i, t) at convergence.
(C) Offline-to-Online Deployment:
  (a) Once converged, store final CTR estimates $\widetilde{x}_{u,i,t}$ in a fast lookup table.
  (b) At bid time, retrieve $\widetilde{x}_{u,i,t}$ for the relevant (u, i) pair to score ad opportunities.

global mixing means the overall model is not strictly conjugate. Attaining a fully conjugate solution would require handling global item–item dependencies in closed-form, which quickly becomes intractable for large-scale data. In practice, however, the partial conjugacy in beta–binomial updates and Gaussian DLM provides strong computational advantages, allowing us to handle real-world volumes without MCMC sampling or variational EM.

3.2 Beta Priors: $(\alpha_{u,i}, \beta_{u,i})$

To model the prior distribution on $\mathrm{CTR}_{u,i}$ (the click-through rate for user u on item i), we adopt a $\mathrm{Beta}(\alpha_{u,i}, \beta_{u,i})$ form. This beta prior incorporates both the observed clicks/impressions and a *global* system-level prior captured by (α_0, β_0). The values of α_0 and β_0 are derived from historical log data *at the start of the process*, providing a consistent global reference point for all user–item pairs. Concretely,

$$\alpha_{u,i} = c_{u,i} \;+\; w_\alpha\,\alpha_0, \qquad \beta_{u,i} = \big(n_{u,i} - c_{u,i}\big) \;+\; w_\beta\,\beta_0,$$

where $c_{u,i}$ is the number of clicks and $n_{u,i}$ is the number of impressions for campaign u on item i. The learnable weights (w_α, w_β) ensure that the global prior does not dominate the data, especially in sparse regimes, by minimizing the negative log-likelihood of the prior.

Negative Log-Likelihood of the Beta Prior. For a single pair (u, i), the negative log-likelihood is

$$\ell_{\mathrm{prior}}^{(u,i)} = -\,c_{u,i}\,\ln\!\Big(\tfrac{\alpha_{u,i}}{\alpha_{u,i}+\beta_{u,i}}\Big) \;-\; \big(n_{u,i} - c_{u,i}\big)\,\ln\!\Big(\tfrac{\beta_{u,i}}{\alpha_{u,i}+\beta_{u,i}}\Big).$$

Rewriting,

$$\ell_{\mathrm{prior}}^{(u,i)} = -\,c_{u,i}\big[\ln(\alpha_{u,i}) - \ln(\alpha_{u,i}+\beta_{u,i})\big] \\ -\;\big(n_{u,i} - c_{u,i}\big)\big[\ln(\beta_{u,i}) - \ln(\alpha_{u,i}+\beta_{u,i})\big].$$

The partial derivatives with respect to w_α and w_β involve the chain rule applied to $\alpha_{u,i}$ and $\beta_{u,i}$. Specifically, for the hierarchical weights w_α and w_β, we have:

$$\frac{\partial \mathcal{L}_{\mathrm{prior}}}{\partial w_\alpha} = \sum_{(u,i)} c_{u,i} \cdot \frac{1}{\alpha_{u,i}} \cdot \frac{\partial \alpha_{u,i}}{\partial w_\alpha} - n_{u,i} \cdot \frac{1}{\alpha_{u,i}+\beta_{u,i}} \cdot \frac{\partial(\alpha_{u,i}+\beta_{u,i})}{\partial w_\alpha},$$

and similarly for w_β.

Summing this cost over *all* (u, i) yields the overall prior loss,

$$\mathcal{L}_{\mathrm{prior}} \;=\; \sum_{(u,i)} \ell_{\mathrm{prior}}^{(u,i)}.$$

We update (w_α, w_β) via gradient-based methods until convergence, ensuring that the final priors respect the data while retaining the global baseline context from (α_0, β_0).

Algorithm 2 Beta+DLM+CF Iterative Steps (Repeated Until Convergence)

Beta Prior Initialization:
  For each (u, i), set

$$\alpha_{u,i} = c_{u,i} \;+\; w_\alpha\,\alpha_0, \qquad \beta_{u,i} = \big(n_{u,i} - c_{u,i}\big) \;+\; w_\beta\,\beta_0.$$

  Solve $\min \mathcal{L}_{\mathrm{prior}}$ to update w_α, w_β.
Kalman Filter (DLM) Step:
  (a) Predict:

$$\widehat{x}_{u,i,t|t-1} = \widehat{x}_{u,i,t-1}, \quad P_{u,i,t|t-1} = P_{u,i,t-1} + Q_{u,i}.$$

  (b) Compute Kalman Gain:

$$K_{u,i,t} = \frac{P_{u,i,t|t-1}}{P_{u,i,t|t-1} + R_{u,i,t}}.$$

  (c) Update:

$$\widehat{x}_{u,i,t} = \widehat{x}_{u,i,t|t-1} \;+\; K_{u,i,t}\big(z_{u,i,t} - \widehat{x}_{u,i,t|t-1}\big),$$
$$P_{u,i,t} = \big(1 - K_{u,i,t}\big)\,P_{u,i,t|t-1}.$$

Collaborative Filtering (Inverse Beta Variance):
  For each (u, i), compute

$$w_{u,i} = \frac{\big(\alpha_{u,i}+\beta_{u,i}\big)^2\,\big(\alpha_{u,i}+\beta_{u,i}+1\big)}{\alpha_{u,i}\,\beta_{u,i}},$$

  the reciprocal of the beta variance.
  Calculate item–item similarities $\mathrm{sim}(i, j)$ using $w_{u,i}\,w_{u,j}$ and $\widehat{x}_{u,i,t}$ from the DLM.
  Update each CTR:

$$\widetilde{x}_{u,i,t} = \frac{\sum_j \mathrm{sim}(i,j)\,\widehat{x}_{u,j,t}}{\sum_j \big|\mathrm{sim}(i,j)\big|}.$$

Hierarchical Bayesian Updates:
  Incorporate hierarchical priors on (w_α, w_β) and (α_0, β_0).
  Update $x_{\mathrm{final}}(u, i, t) \;=\; \phi\; p_{\mathrm{beta}}(u, i) \;+\; (1 - \phi)\;\widetilde{x}_{u,i,t}$.
  Solve $\min\Big[\mathcal{L}_{\mathrm{prior}} + \mathcal{L}_{\mathrm{CF}} + \mathcal{L}_{\mathrm{KF}} + \lambda_p\big(w_\alpha^2 + w_\beta^2 + \phi^2\big)\Big]$.
Check Convergence:
  If changes in $(w_\alpha, w_\beta, \phi)$ or the total loss are below threshold, end. Else, repeat.

3.3 Kalman Filtering for Temporal Dynamics

To handle temporal variation in campaign–item CTRs, we adopt a Dynamic Linear Model (DLM). At time t, the observed CTR $z_{u,i,t}$ is treated as a noisy measurement of a latent state $x_{u,i,t}$, which evolves according to

$$x_{u,i,t} = x_{u,i,t-1} + \omega_{u,i,t}, \quad z_{u,i,t} = x_{u,i,t} + \nu_{u,i,t},$$

where $\omega_{u,i,t} \sim \mathcal{N}(0, Q_{u,i})$ is the process noise, and $\nu_{u,i,t} \sim \mathcal{N}(0, R_{u,i,t})$ is the measurement noise. The standard Kalman update recursions yield a posterior estimate $\widehat{x}_{u,i,t}$ of the latent CTR. In practice, the measurement variance $R_{u,i,t}$ approximates a binomial proportion:

$$R_{u,i,t} \approx \frac{\widehat{x}_{u,i,t|t-1}\left(1 - \widehat{x}_{u,i,t|t-1}\right)}{n_{u,i,t}},$$

where $n_{u,i,t}$ is the number of impressions at time t.

Process Noise Calibration. Although one could define $Q_{u,i} = \exp(\theta_{u,i})$ and optimize a distinct parameter $\theta_{u,i}$ for each user–item pair, we instead selects a *single global* σ^2 for the sake of computational tractability. We perform a grid search for Q over a range of candidate values and choose the one minimizing a Kalman-filter SSE. Formally,

$$\min_{Q} \mathcal{L}_{\mathrm{KF}} = \sum_{t,i \in \text{top-}k} \left[\frac{\left(z_{i,t} - \widehat{x}_{i,t|t-1}\right)^2}{P_{i,t|t-1} + R_{i,t}} + \ln\left(P_{i,t|t-1} + R_{i,t}\right) \right],$$

where $P_{i,t|t-1} = P_{i,t-1} + Q$ is the prior state uncertainty for item i, and $z_{i,t}$ is the observed CTR. After identifying the Q value that best fits these top-k items, we apply it globally to every campaign–item pair in the analytic pipeline. This strategy balances numerical tractability with robust smoothing, obviating the need for item-specific process noise estimates and preventing extreme Q values in sparse regions of the dataset.

3.4 Collaborative Filtering with Inverse-Variance Weights

Given $\widehat{x}_{u,i,t}$, the DLM posterior $x_{u,i,t}$ mean for each campaign u and item i at time t, we refine those estimates via *item-based* collaborative filtering (CF) on *centered* data. Each pair (u, i) receives a beta-based weight:

$$w_{u,i} = \frac{\left(\alpha_{u,i} + \beta_{u,i}\right)^2 \left(\alpha_{u,i} + \beta_{u,i} + 1\right)}{\alpha_{u,i}\, \beta_{u,i}},$$

which is the reciprocal of the variance of the beta distribution. Items with sparse or volatile data thus have smaller $w_{u,i}$ and exert less influence on other items.

Weighted Cosine Similarity on Centered Vectors. To compare items i and j, we first *center* each item's DLM posteriors across campaigns:

$$x'_{u,i,t} = \hat{x}_{u,i,t} - \bar{\hat{x}}_{i,t},$$

where $\bar{x}_{i,t}$ is item i's mean DLM posterior CTR across users at time t. Then, we incorporate the beta weights in a *weighted cosine similarity*:

$$\mathrm{sim}_w(i,j) = \frac{\sum_u w_{u,i}\, w_{u,j} \left(x'_{u,i,t}\right)\left(x'_{u,j,t}\right)}{\sqrt{\sum_u \left(w_{u,i}\, x'_{u,i,t}\right)^2}\, \sqrt{\sum_u \left(w_{u,j}\, x'_{u,j,t}\right)^2}}.$$

Hence, each item's column is effectively scaled by $w_{u,i}$ before taking the dot products. Once we obtain $\mathrm{sim}_w(i,j)$ for all pairs, we update the CTR estimate for user u on item i via:

$$\widetilde{x}_{u,i,t} = \frac{\sum_j \mathrm{sim}_w(i,j)\, x_{u,j,t}}{\sum_j \left|\mathrm{sim}_w(i,j)\right|}.$$

The denominator normalizes by the sum of absolute similarities so that high-similarity neighbors have more impact. One may also impose a top-k truncation on neighbors to reduce computational overhead. This CF step thus blends time-aware DLM updates with beta-based inverse-variance weighting, providing robust, neighbor-driven refinements in sparse settings.

3.5 Hierarchical Bayesian Modeling

Campaigns from the same advertiser often share thematic designs or targeting constraints. To leverage these similarities, we incorporate hierarchical gamma priors on the beta weights (w_α, w_β) and on (α_0, β_0) for each advertiser by pooling data across multiple campaigns:

$$w_\alpha \sim \mathrm{Gamma}(a_\alpha, b_\alpha), \qquad w_\beta \sim \mathrm{Gamma}(a_\beta, b_\beta).$$

These hyperpriors reduce variance and mitigate cold-start problems for newly added campaigns. They may be fitted via gradient-based methods.

Blending Beta Posterior and DLM State. In addition to the hierarchical structure on (w_α, w_β) and (α_0, β_0), we define a mixture to combine the beta-based posterior mean with the DLM-updated CTR. Specifically, for each campaign–item pair (u, i), let

$$p_{\mathrm{beta}}(u,i) = \frac{\alpha_{u,i}}{\alpha_{u,i} + \beta_{u,i}},$$

represent the beta posterior mean derived from $(\alpha_{u,i}, \beta_{u,i})$. Then, we introduce a global mixture parameter $\phi \in [0,1]$. The final CTR prediction is:

$$x_{\mathrm{final}}(u,i,t) = \phi \left[p_{\mathrm{beta}}(u,i)\right] + \left(1 - \phi\right) \widetilde{x}_{u,i,t},$$

where $\widetilde{x}_{u,i,t}$ is the CF-based state. Hence, the hierarchical Bayesian framework supplies campaign-level priors for (w_α, w_β) and baseline hyperparameters (α_0, β_0), while the DLM smoothly adapts CTR estimates over time and the CF step incorporates item exploration. The mixture factor ϕ balances these two components, finding the optimal mixture between the total accrued information through the beta posterior and the time-varying DLM+CF process.

3.6 Offline Training and Real-Time Scoring

All steps run in a batch pipeline, collecting daily (or sub-daily) logs to update $(w_\alpha, w_\beta, \phi)$, Q, CF similarities, and hierarchical parameters. The final CTR estimates $x_{\mathrm{final}}(u,i,t)$ are then cached for real-time retrieval, providing swift inferences for high-frequency bidding requests.

4 EXPERIMENTS AND RESULTS

We conduct an extensive set of experiments on a large-scale *mobile display advertising* dataset sampled from InMarket's Demand-Side Platform (DSP). Our primary goal is to investigate whether

combining beta-based priors, Dynamic Linear Models (DLMs), and collaborative filtering (CF) with inverse variance weighting can surpass established baselines in click-through rate (CTR) prediction accuracy. This section outlines the data properties, experimental protocols, evaluation criteria, and the final performance results on the test set.

4.1 Dataset and Experimental Protocol

Data Collection and Summary. We use impression-and-click logs from InMarket's DSP, covering campaigns (users) and ad opportunities (items) over **July 1, 2024** to **September 30, 2024**. Each record comprises:

- **Campaign ID** (u),
- **Item/App ID** (i),
- **Impressions** $(n_{u,i,t})$,
- **Clicks** $(c_{u,i,t})$,
- **Day-level timestamp** (t).

The dataset is highly sparse and includes 1,438 unique campaigns, 18,653 items, and 36 advertisers. In total, there are 588,457,330 impressions, 2,655,840 clicks, and an overall average CTR of 0.00451.

Time-Based Partitioning. We split the data chronologically into:

- **Training**: First 56 days (60%) of the log period,
- **Validation**: Next 18 days (20%),
- **Test**: Final 18 days (20%).

Hence, the training set covers early July to late August; the validation set spans the subsequent 18 days; and the test set covers the remainder of September. We optimize hyperparameters on the validation set, then assess final model performance on the test set.

Competing Methods. We evaluate the following models:

- **Beta+DLM+CF** (*ours*): Integrates beta-based priors, temporal smoothing via a Kalman-filtered DLM, and CF weighted by the inverse beta variance.
- **Beta+CF**: An ablated version of our approach that uses beta priors in CF but omits the DLM time-series component.
- **DLM+CF**: Another ablated version that applies a DLM for temporal updates plus CF, but without beta priors.
- **Vanilla CF**: A standard item-based collaborative filter without beta-based uncertainty or temporal modeling.
- **Random Forest**: A tree-ensemble classifier using campaign/item IDs as categorical features, ignoring explicit temporal adaptation.
- **XGBoost**: A boosted-tree classifier recognized for strong performance in many CTR tasks, similarly lacking an explicit temporal mechanism.

4.2 Evaluation Metrics

We report both MSE and log loss metrics. The log loss provides a stronger penalization for rare events than most other measures:

Log-Loss (Binary Cross-Entropy).

$$\text{LogLoss} = -\frac{\sum_{(u,i,t)} \left[c_{u,i,t} \, \ln(\widetilde{x}_{u,i,t}) + \left(n_{u,i,t} - c_{u,i,t}\right) \, \ln\left(1 - \widetilde{x}_{u,i,t}\right) \right]}{\sum_{(u,i,t)} n_{u,i,t}}.$$

Log-loss heavily penalizes poor probability estimates.

Mean Squared Error (MSE)..

$$\text{MSE} = \sum_{(u,i,t)} \left(\widetilde{x}_{u,i,t} - \frac{c_{u,i,t}}{n_{u,i,t}} \right)^2 \Big/ \sum_{(u,i,t)} 1,$$

which provides a simple global measure of prediction accuracy.

4.3 Test Set Results

Table 1 summarizes each method's performance (log-loss and MSE) on the test set. Notably, we incorporate two partial variants of our final pipeline—*Beta+CF* and *DLM+CF*—to illustrate the incremental gains from combining beta priors and dynamic modeling.

Table 1: Test Set Performance: Log-Loss and MSE (lower is better). Our full Beta+DLM+CF approach achieves the best accuracy, demonstrating the combined value of beta priors and DLM smoothing.

Method	Log-Loss	MSE
Beta+DLM+CF	**0.025234**	**0.000144**
Beta+CF	0.029487	0.000185
DLM+CF	0.030523	0.000523
Vanilla CF	0.035877	0.000654
Random Forest	0.042545	0.000204
XGBoost	0.043481	0.000204

4.4 Discussion of Improvements

Impact of Beta Priors vs. DLM.. Comparing **DLM+CF** (log-loss 0.030523) with **Beta+CF** (log-loss 0.029487) reveals that beta priors alone yield a larger improvement than DLM alone for these data. This highlights how uncertainty modeling via beta distributions can substantially mitigate sparse or cold-start conditions by preventing overconfident estimates.

Combination of Beta and DLM.. Our full model, **Beta+DLM+CF**, outperforms both partial variants (*Beta+CF* and *DLM+CF*), underscoring the combined benefits of dynamic smoothing on top of beta-based priors. In particular, the log-loss drops from 0.029487 to 0.025234—a relative improvement of over 14%—once we include time-series adaptation in addition to beta's uncertainty mechanism.

Comparison to Tree Methods. Random forest and XGBoost yield higher log-loss (above 0.042) and slightly higher MSE, aligning with previous observations that purely feature-based classifiers often lack a built-in way to handle time-varying CTR or confidence weighting for sparse user-item pairs.

Vanilla CF vs. Weighted Approaches. Vanilla CF performs worse (0.035877 log-loss) than any version incorporating beta or DLM. This gap illustrates the importance of weighting edges by confidence (beta variance) and capturing time-series trends (DLM). The naive CF approach fails to handle heterogeneity across campaigns, leading to less reliable neighbor estimates.

4.5 Summary of Experimental Findings

The experimental results confirm that **Beta+DLM+CF** provides the most robust CTR predictions, with both beta priors and temporal smoothing offering clear advantages over partial or alternative methods. Each ablated variant (Beta+CF, DLM+CF) highlights a piece of our final pipeline's contribution. Overall, these findings motivate the integration of uncertainty modeling, dynamic adaptation, and collaborative filtering in next-generation programmatic advertising systems.

4.6 Runtime and Scalability

Another key advantage of our framework is its computational efficiency. Both the DLM updates and the item-based CF steps are embarrassingly parallel, allowing us to batch daily logs or distribute user–item pairs across multiple threads. Moreover, the Bayesian components (beta prior updates, hierarchical hyperparameters) use conjugate forms that yield closed-form updates.

On a 13th Gen Intel® Core™ i9-13900H laptop, the entire 90-day experiment (training + daily DLM + CF steps and convergence) finished in **1 hour 47 minutes**. Once the model is fitted, generating CTR outputs for a single new day took only **0.077 seconds** per campaign, making this approach well-suited for daily or sub-daily updates in real-time bidding settings.

5 CONCLUSION

We have presented a new CTR prediction framework, *Beta+DLM+CF*, which unifies beta-based priors, a Dynamic Linear Model (DLM) for temporal smoothing, and collaborative filtering (CF) driven by inverse beta variance weighting. This combination addresses three key challenges in real-time bidding environments: uncertainty due to sparse user–item interactions, non-stationary user behavior over time, and the heterogeneous importance of different campaigns and items. By learning priors that adapt to data availability, accounting for temporal shifts with the Kalman filter, and emphasizing reliable (low-variance) neighbors in CF, the proposed system significantly improves log-loss and MSE relative to standard baselines.

Empirically, we demonstrated that our approach achieves state-of-the-art performance on a large-scale dataset, outperforming methods such as vanilla CF, item-average baselines, and strong tree-based classifiers (Random Forest, XGBoost). An ablation analysis further confirmed that removing any of our framework's components (e.g., beta priors, DLM smoothing, or variance-based CF) causes measurable deterioration in prediction accuracy. Moreover, the hierarchical Bayesian extension empowers the system to mitigate cold-start conditions by pooling statistical strength across related campaigns.

From a practical standpoint, the offline training plus real-time inference design ensures compatibility with high-frequency programmatic auctions, where computational speed is paramount. Our calibration procedures for process noise (Q) and hyperpriors (α_0, β_0) balance robustness with responsiveness to changes in user behavior.

Several promising directions remain for future exploration. First, we intend to integrate more advanced embeddings and contextual signals (e.g., user demographics or device types) into the CF step. Second, adapting reinforcement learning or bandit methods on top of Beta+DLM+CF could further optimize bidding policies, leveraging strong CTR predictions as a baseline while exploring novel ad placements. Lastly, scaling hierarchical priors to thousands of advertisers raises compelling research questions on distributed inference and approximate Bayesian techniques. We anticipate that continued refinement of these ideas will advance the field of CTR prediction, offering increasingly accurate and adaptive solutions for real-time advertising systems.

REFERENCES

[1] Tianqi Chen and Carlos Guestrin. XGBoost: A scalable tree boosting system. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, pages 785–794. ACM, 2016. doi: 10.1145/2939672.2939785. URL https://doi.org/10.1145/2939672.2939785.

[2] Heng-Tze Cheng, Levent Koc, Jeremiah Harmsen, Tal Shaked, Tushar Chandra, Hrishi Aradhye, Glen Anderson, Greg Corrado, Wei Chai, Mustafa Ispir, Rohan Anil, Zakaria Haque, Lichan Hong, Vihan Jain, Xiaobing Liu, and Hemal Shah. Wide & deep learning for recommender systems. In *Proceedings of the 1st Workshop on Deep Learning for Recommender Systems*, DLRS 2016, page 7–10, New York, NY, USA, 2016. Association for Computing Machinery. ISBN 9781450347952. doi: 10.1145/2988450.2988454. URL https://doi.org/10.1145/2988450.2988454.

[3] Andrew Gelman, John B. Carlin, Hal S. Stern, David B. Dunson, Aki Vehtari, and Donald B. Rubin. *Bayesian Data Analysis*. Chapman and Hall/CRC, second edition edition, 2006. ISBN 978-1584883883.

[4] Huifeng Guo, Ruiming Tang, Yunming Ye, Zhenguo Li, and Xiuqiang He. Deepfm: A factorization-machine based neural network for ctr prediction. In *Proceedings of the 26th International Joint Conference on Artificial Intelligence*, pages 1725–1731, 2017.

[5] Xinran He, Junfeng Pan, Ou Jin, Tianbing Xu, Bo Liu, Tao Xu, Yanxin Shi, Antoine Atallah, Ralf Herbrich, Stuart Bowers, and Joaquin Quiñonero Candela. Practical lessons from predicting clicks on ads at facebook. In *International Workshop on Data Mining for Online Advertising*, 2014. URL https://api.semanticscholar.org/CorpusID:2999385.

[6] Yehuda Koren. Collaborative filtering with temporal dynamics. In *Communications of the ACM*, volume 53, pages 89–97, 2010.

[7] Yehuda Koren, Robert Bell, and Chris Volinsky. Matrix factorization techniques for recommender systems. *Computer*, 42(8):30–37, 2009. doi: 10.1109/MC.2009.263. URL https://doi.org/10.1109/MC.2009.263.

[8] Linhui Fu Mingyan Li Zhengyu Tang Patrick M. LeBlanc, David Banks and Qiuyi Wu. Recommender systems: A review. *Journal of the American Statistical Association*, 119(545):773–785, 2024. doi: 10.1080/01621459.2023.2279695. URL https://doi.org/10.1080/01621459.2023.2279695.

[9] Steffen Rendle. Factorization machines. *2010 IEEE International Conference on Data Mining*, pages 995–1000, 2010. URL https://api.semanticscholar.org/CorpusID:17265929.

[10] Matthew Richardson, Ewa Dominowska, and Robert J. Ragno. Predicting clicks: estimating the click-through rate for new ads. In *The Web Conference*, 2007. URL https://api.semanticscholar.org/CorpusID:14669618.

[11] Mike West and Jeff Harrison. *Bayesian forecasting and dynamic models*. Springer, 1997.

[12] Yang Zhao, Chang Zhou, Jin Cao, Yi Zhao, Shaobo Liu, Chiyu Cheng, and Xingchen Li. Multi-scenario combination based on multi-agent reinforcement learning to optimize the advertising recommendation system. In *2024 5th International Conference on Artificial Intelligence and Electromechanical Automation IEEE*, 2024. doi: 10.48550/2407.02759. URL https://arxiv.org/abs/2407.02759.

[13] Guorui Zhou, Na Mou, Ying Fan, Qi Pi, Weijie Bian, Chang Zhou, Xiaoqiang Zhu, and Kun Gai. Deep interest evolution network for click-through rate prediction. *Proceedings of the AAAI Conference on Artificial Intelligence*, 33:5941–5948, 07 2019. doi: 10.1609/aaai.v33i01.33015941.