Ad Ranking for Walmart Display Ads System

Fan Yang*, Weijie Yuan, Nahid Anwar, Braden Huffman, Fangping Huang, Lichun Chen, Konstantin Shmakov, Sunil Goda, James Jung, Niranjana Moleyar, Xiaobo Peng, Kuang-chih Lee and Musen Wen

Walmart Inc., USA

Abstract

Walmart's Artemis initiative, which establishes an in-house advertising (ad) platform, holds significant business implications aligned with Walmart's mission to enable customers to save money and live better. By introducing auction-based pricing mechanisms for ad impressions, suppliers seeking to connect customers with pertinent display and video advertisements can specify their desired price per impression across Walmart's owned-and-operated properties as well as third-party sites. A critical challenge in developing this system is to ensure that the ads displayed on our platform are relevant to users. In this paper, we present our first ever endeavor to build an in-house demand-side platform (DSP) ad ranking system within Walmart. We outline our iterative approach to enhancing ad relevance, thereby improving user engagement by catering to user intent and needs, while simultaneously assisting advertisers in achieving their advertising objectives and maximizing return on investment.

Keywords

DSP, XGBoost, Airflow, DAG, click-through-rate, conversion-rate, Deep Ranker

1. Introduction

Demand-side platform (DSP) advertising has become increasingly popular in recent years due to its ability to streamline advertising process for both advertisers and publishers. DSPs enable advertisers to purchase advertising inventory across various platforms, including display, video, mobile, and social media, all through a single interface. Leveraging advanced targeting and optimization algorithms, DSPs facilitate advertisers in reaching their intended audiences with enhanced precision and efficiency, while also generating valuable data-driven insights to inform and refine future advertising campaign strategies.

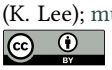
Prior to implementing an in-house demand-side platform system, Walmart relied on third-party platforms, such as Google and TradeDesk, to connect advertisers with their target audiences on the Walmart website. However, these off-the-shelf DSP solutions were neither optimized for Walmart's specific e-commerce requirement nor cost-effective. Consequently, Walmart initiated the development of a customized DSP system designed to better address its distinct advertising needs through the incorporation of tailored data logic. In this paper, we detail the experiments and advancements conducted over the past two years in pursuit of creating this novel system. Although further progress remains necessary, our aim is to offer valuable insights and perspectives derived from our experience in developing this sophisticated, custom-built system.

Specifically, we (1) highlight the key features and data sources incorporated into the system; (2) detail the engineering framework and describe how data science integrates into this process; (3) discuss the machine learning model training and data ingestion pipeline employed to build the core ranking model; and (4) present the results of a series of conducted experiments. Through this paper, we aim to provide insights into the challenges encountered while developing a custom DSP system from scratch,

SIGIR eCom'25: 2025 SIGIR Workshop on eCommerce, July 17, 2025, Padua, Italy

*Corresponding author.

fan.yang0@walmart.com (F. Yang); weijie.yuan@walmart.com (W. Yuan); nahid.anwar@walmart.com (N. Anwar); braden.huffman@walmart.com (B. Huffman); fangping.huang@walmart.com (F. Huang); lichun.chen@walmart.com (L. Chen); konstantin.shmakov@walmart.com (K. Shmakov); sunil.goda@walmart.com (S. Goda); james.jung@walmart.com (J. Jung); niranjana.moleyar@walmart.com (N. Moleyar); xiaobo.peng@walmart.com (X. Peng); kuangchih.lee@walmart.com (K. Lee); musen.wen@walmart.com (M. Wen)



as well as describe the innovative solutions implemented to address these challenges. Furthermore, we outline a road-map for future work, identifying potential next steps aimed at further enhancement and optimization of the proposed system.

2. Background

In advertising technology DSP systems are used by advertisers to bid for ad space [1, 2]. In our context, the platform is the Walmart website, and advertisers are companies involved in selling products ranging from dog food to children's toys. Specifically, if a user is on a page with a particular context, for example searching for orange juice, it would be relevant to suggest associated items based on the user's location and past behavior. Given that, for each ad location, there are hundreds of eligible ads that can be surfaced to the user; the primary challenge for data science is determining the optimal advertisement to display to the user.

Determining how to rank candidate ads to ensure relevance to users to our business success, as it directly influences both user engagement experiences and advertiser business objectives. An ideal ranking methodology should strive to maximize user engagement as well as advertiser's return on investment (ROI). Currently, the ranking formula within Walmart's in-house Artemis system is calculated by multiplying the engagement score by the bid price, as highlighted by the dark green color in Figure 1. As a result, to accurately predict each user's engagement score plays a decisive role for the whole Artemis system. Our team's responsibilities span across the engagement prediction and bid price generation. However, due to the scope limitation of this paper, we will focus exclusively on our efforts related to engagement prediction.

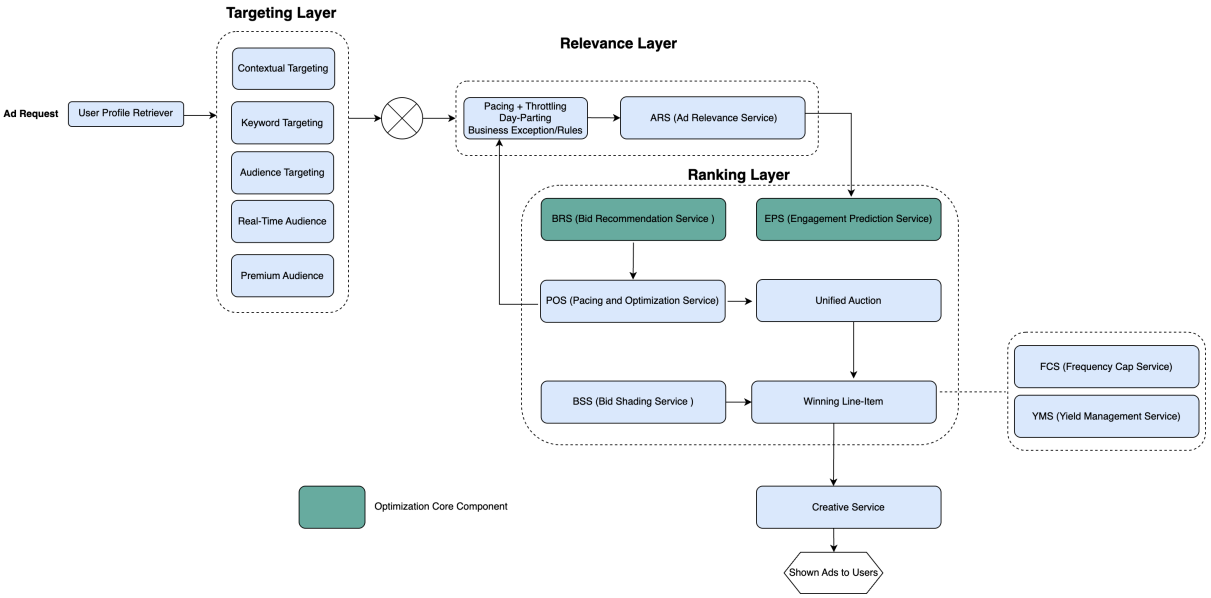


Figure 1: Overview of the Walmart in-house DSP Artemis Architecture

3. Implementation

In this section, we detail our implementation approach from multiple perspectives, including the data and features used in modeling, the modeling framework and techniques employed, and operational excellence considerations.

3.1. Data and Features

Given the highly imbalanced nature of engagement data, significant care was taken to ensure that our model did not become biased towards the majority class [3]. As commonly observed in click-through rate (CTR) prediction tasks, the number of click events is substantially smaller than non-click events. To address this issue, we employed down-sampling to the negative class (non-click events) and assigned greater weight to the minority class (click events) relative to the majority class during the model training process.

We will now discuss the features utilized by our model. Figure 2 illustrates the set of features currently employed in our modeling framework. These features generally fall into the following categories.

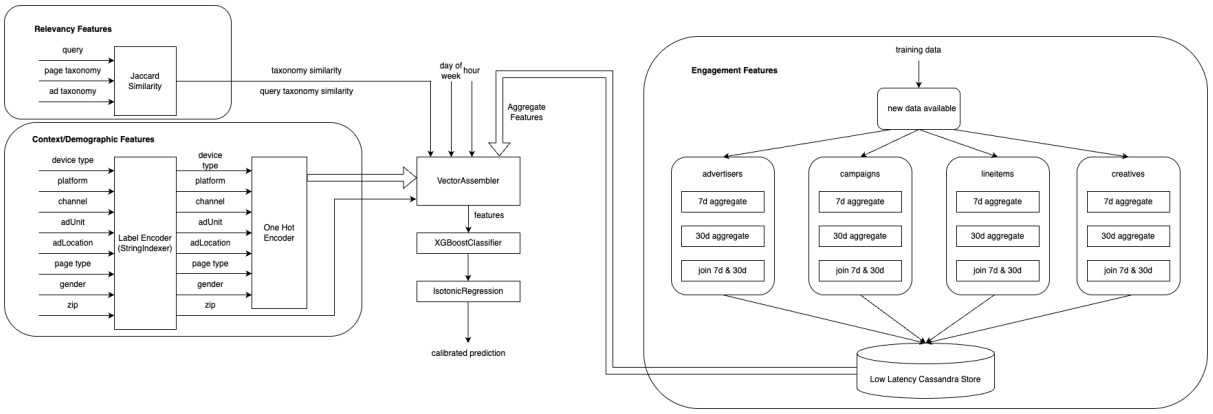


Figure 2: Overview Of Features Used in the Model

The first category consists of supply-side features, like `adLocation`, `deviceType`, `platform` etc. Another category encompasses request-side features like `zipcode`, `time of the day`, `day of week` etc.

Reformatting the Ad Taxonomy path by breaking up its representation into words which enabled the creation of one of our crucial features categories: the `Taxonomy Score`. This score is calculated as the Jaccard similarity [4] between the Page Taxonomy Path (representing the current user's path) and the Ad Taxonomy Path (representing the candidate ad category to be displayed to the user). Upon investigating feature importance, we identified this `Taxonomy Score` as a particularly significant feature, as it can provide essential relevancy measures connecting the user's current page context to candidate advertisements.

Historical ad performance provides valuable guidance for predicting future performance. Therefore, we introduced ad engagement features into our model, such as the 30-day average click-through rate (`ctr_30d`) and the 7-day average click-through rate (`ctr_7d`), calculated at multiple granularities including advertiser, campaign, creative, and line-item levels. As these features are unavailable at runtime, we need to pre-process them which is illustrated in the right part of the Figure 2. The pre-computed feature data is then stored in two copies: one copy is saved to a Google Cloud Storage (GCS) [5] bucket for offline training data integration, and the other copy is ingested into an online Cassandra database [6] to facilitate rapid feature retrieval during runtime.

Additionally, we incorporate user-level features such as `gender` and `age`. We are currently emphasizing on adding more of this category of features, as our analyses indicated that incorporating user-specific attributes significantly enhances the targeting capability of our advertisements, allowing us to reach more relevant users. Further details regarding our exploration and utilization of user features are presented in the case study section.

Finally, we introduced additional pairwise features, such as the duration of time spent by each user in various top-level departments, and the frequency with which each user visited pages within level 1 product-type categories on Walmart's website over the past 30 days. Leveraging these cross features between users and advertised items enables us to uncover user preferences more effectively and generate

more personalized ad recommendations based on historical user behavior. Our subsequent analysis of feature importance confirmed that incorporating these pairwise interaction features significantly enhances prediction accuracy and ad targeting effectiveness.

3.2. Model Framework

The primary model architecture employed for our custom DSP is based on XGBoost [7]. This solution was selected due to its speed, reliability, and track record of success. Additionally, XGBoost is already utilized by the Sponsored Products (SP) section of Walmart's AdTech division, making it a natural choice for our application. By leveraging XGBoost, we constructed a powerful and adaptable model capable of delivering valuable insights for optimizing campaign performance.

To simply demonstrate the algorithm of XGBoost. The objective function at iteration t that we need to minimize is the following:

$$\mathscr{L}^{(t)} = \sum_{i=1}^{n} l\left(y_i, \hat{y}_i^{(t-1)} + f_t(x_i)\right) + \Omega(f_t) \tag{1}$$

Where l is a function of CART learners, a sum of the current and previous additive trees. $\Omega(f_t)$ is the regularization at iteration t. Basically, at iteration t, we need to build a learner that achieves the maximum possible reduction of loss. In practice, the steps to build the next learner are:

- Start with a single root node (containing all the training examples).
- Iterate over all features and their respective values, evaluating the loss reduction for each possible split.
- The gain for the best split must be positive and greater than the *min_split_gain* parameter; otherwise, we stop growing the branch.

Furthermore, due to our strategy of down-sampling negative data points, calibration techniques [8] were applied after XGBoost modeling to produce output scores that more accurately represent the true likelihood of events (clicks or conversions, in our case) and are easier to interpret by subsequent optimization systems. Among several widely adopted calibration approaches, we selected isotonic regression, a non-parametric method [9]. Unlike common techniques such as Platt scaling, which assumes a logistic relationship, isotonic regression does not impose any predefined functional form. Additionally, isotonic regression preserves the monotonic relationship between predicted scores and actual probabilities which is a very essential characteristic for ad-ranking models. In simple words, calibration through isotonic regression does not alter the original ranking of impressions provided by the XGBoost model.

To demonstrate the algorithm of isotonic regression, let us assume $(x_1, y_1), \ldots, (x_n, y_n)$ be a simplified representation of a given set of observations, where x_i is the raw probability from XGBoost and y_i is the ground truth label. Isotonic regression seeks a least-squares fit $\hat{y}_i$ for all i, subject to the constraint that $\hat{y}_i \leq \hat{y}_j$ whenever $x_i \leq x_j$. Formally, the isotonic regression objective can be represented as follows:

$$\begin{aligned} &\min \sum_{i=1}^{n} (\hat{y}_i - y_i)^2 \\ &\text{s.t. } \hat{y}_i \leq \hat{y}_j \quad \text{for all } (i,j) \in E, \text{where } E = \{(i,j) \,:\, x_i \leq x_j\} \end{aligned} \tag{2}$$

Models developed by data scientists have traditionally been integrated into engineering systems through manual processes. Specifically, data scientists construct models using Jupyter notebooks hosted on Google Cloud Platform's (GCP) Dataproc environment. These models are rigorously evaluated and validated using offline performance metrics, typically ROC-AUC and PR-AUC scores [10, 11, 12].

Once a model achieves satisfactory performance, a corresponding MLEAP bundle is generated to facilitate integration into production systems [13]. MLEAP is a widely adopted tool in the industry, offering a range of benefits. Using MLEAP, data scientists can construct a comprehensive 'pipeline'

consisting of multiple steps, allowing for complex transformations of the raw features, for example, preprocessing the input features, executing custom transformations, running machine learning models such as XGBoost, and converting the raw probability scores to produce well-calibrated predictions. By providing a standardized, repeatable process for building and deploying models, MLEAP helps streamline the integration of data science models into engineering systems.

A crucial feature of MLEAP is its capability to serialize model pipelines into deployable bundles. Once generated, these bundles can be utilized by engineering teams to instantiate Java or Python objects capable of delivering real-time predictions during runtime. PySpark/MLEAP bundles are optimized for computational efficiency, establishing them as a reliable and widely adopted industry standard. This approach facilitates faster prediction serving and enhances the seamless integration of data science models within engineering frameworks. An overview of our XGBoost-based PySpark/MLEAP model-building process is presented in Figure 3.

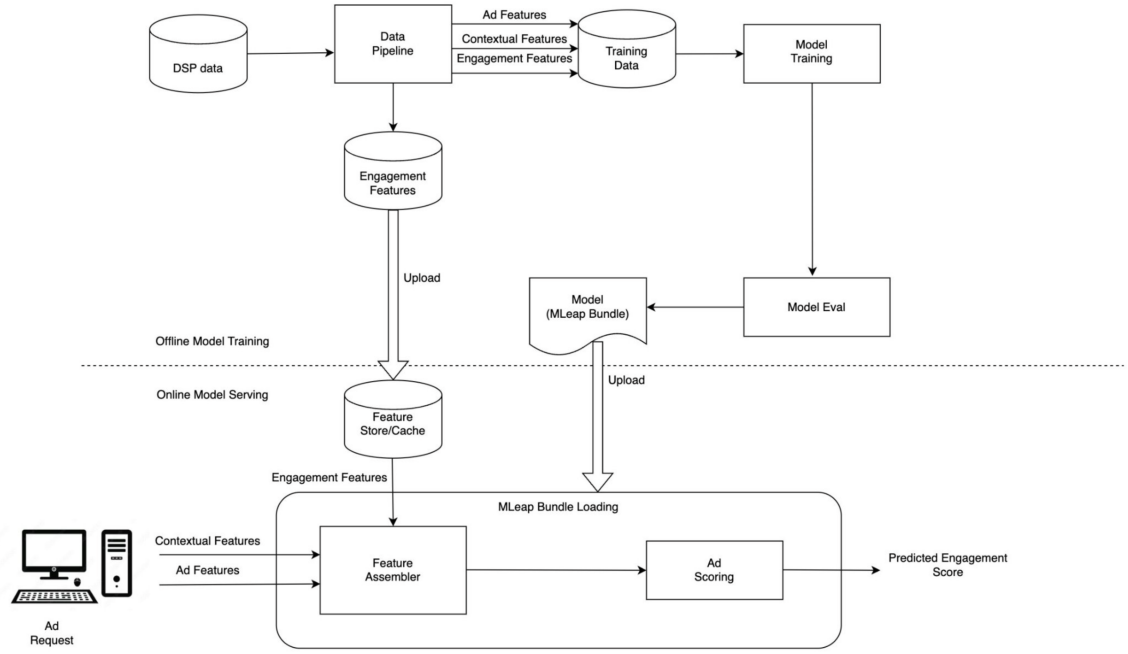


Figure 3: Overview of Model Building Framework

Our current MLEAP-XGBoost model is effective at using categorical contextual features, numeric historical engagement features as well as other low dimensional data; however, we are currently exploring high dimensional representations of both users and advertisements. XGBoost has limitations when working with this high dimensional data. To combat this we've begun exploring different deep learning models. We initially explored a simple multi-layer perceptron, knowing we were planning on moving to a deep learning architecture once we obtained the embedded features. Without embedded features, the MLP saw an AUC-ROC drop of 1.32% and an AUC-PR drop of 0.092% when compared to the current production model.

These results were promising, but not enough to immediately transition. Our team began looking for other models. We tried a Wide and Deep architecture as well as a DeepFMx architecture [14]. Ultimately, these other models didn't solve a second challenge we faced.

We currently support four types of engagement: impressions, views, clicks and conversions. Views, clicks and conversions each require individual models, each packaged as MLEAP bundles, while impressions always have an engagement score of 1. Due to system constraints, each engagement scoring request can only be passed through one of these three aforementioned models due to latency constraints.

In order to utilize embedded features as well as provide three simultaneous engagement scores, we

are developing a multi-task deep learning ranker with an MMoE architecture [15, 16]. Unlike our XGBoost models, we store our deep learning models in model.onnx files. We then serve the loaded deep learning model in a Triton environment [17].

3.3. Model Automation Framework

Although manually building model bundles in notebooks is feasible, this approach may lack scalability. To enhance our processes and improve efficiency, we developed an Airflow Directed Acyclic Graph (DAG) for model training, evaluation, and promotion [18]. This automation ensures consistent and reliable execution of these tasks. Airflow extends traditional scheduling tools like crontab by offering capabilities such as resource allocation, a user-friendly interface for monitoring logs and job execution histories, and task scheduling.

Models are trained on daily basis using the most recent data and subsequently saved to a Google Cloud Storage (GCS) bucket. The engineering team retrieves these updated models and creates the corresponding Java objects used for real-time predictions. The engineering system handles feature extraction from relevant ads and passes these features to the model, generating probability outputs used for ad-ranking decisions.

To enhance our Airflow job monitoring, we integrated our workflows with Hubot and implemented customized logging. These enhancements enable the direct transmission of specific model metrics to a dedicated Slack channel. Airflow also simplifies the aggregation of workflow steps into unified jobs and supports daily hyperparameter tuning during model training.

Recently, collaboration with our machine learning infrastructure team led to the Airflow DAG development process becoming more efficient. The machine learning infrastructure team integrates most of the features mentioned above into one single unified platform which is called model automation framework (MAF). MAF facilitates streamlined on-boarding of new models, training old models with updated configurations, triggering new run to refresh model artifacts, inspecting model metrics, and assessing feature coverage, significantly improving the efficiency of our model deployment and promotion processes.

4. Case Study

Our efforts to boost system performance have focused on two primary areas: model innovation and feature innovation. For the purpose of this paper, we focus on our advancements related to feature innovation in this section.

4.1. Click Model User Feature

Previously, our models incorporated ad-side engagement features, which measured how different users interacted with a particular ad, for instance, metrics such as the total number of impressions or the ad's average click-through rate (CTR). These ad-centric features contributed significantly to model performance and ranked prominently in feature importance analyses. Extending this approach, we subsequently examined another critical dimension influencing ad engagement: user past behavior.

Figure 4 illustrates the calculation methodology for these user features. Specifically, we aggregated each user's interactions with Walmart's display ads over the previous 7-day and 30-day periods, generating summary metrics such as total impressions, views, clicks, and conversions, along with derived ratios including view rate, click-through rate (CTR), and conversion rate (CVR). These newly created user-level features were subsequently combined with existing production features for offline evaluation. In our offline simulation, we observed a relative lift of 3% in ROC-AUC performance, as shown in Table 2. Motivated by these promising results, we proceeded to deploy these features in production.

The primary challenge in deploying this feature to production lies in efficiently managing the high volume of user-level features at scale for both offline and online environments. Since, Walmart E-commerce observes more than 30 million unique users daily; therefore, efficiently generating hundreds

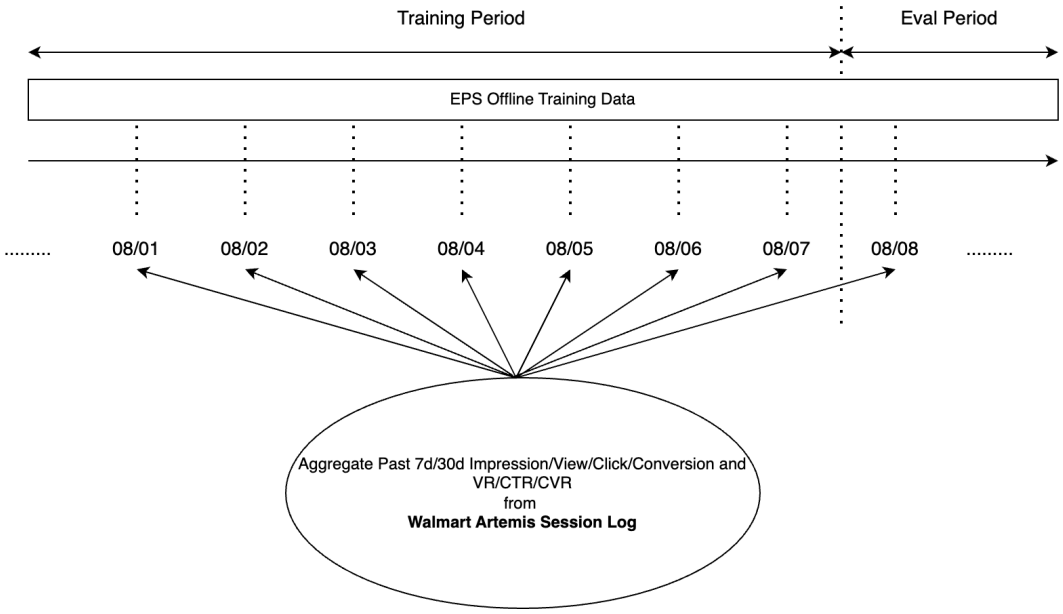


Figure 4: User Engagement Generation and Offline Evaluation Process

Table 1
User Engagement Feature Offline Evaluation Results

Metric	Baseline (Current Production Features)	Baseline + Past 7d Features	Baseline + Past 7d/30d Features	Relative Lift
ROC-AUC	0.9000	0.9140	0.9272	3.02%
PR-AUC	0.10000	0.11004	0.16116	61.16%

Note: Metric values have been shifted by additive deltas for anonymization. Relative improvements (Lift) remain unchanged.

of millions of offline training data records, along with handling online serving queries within tens of milliseconds, necessitates sophisticated system architecture and best engineering practices. Through multiple iterations and explorations, we developed a scalable user-feature architecture by leveraging existing infrastructure components capable of addressing these demanding requirements.

At first, data scientists provide prototype user-feature aggregation queries to the Feature Management Service (FMS) team. FMS refines these queries into production-quality code and subsequently stores the aggregated features in both offline and online feature stores. Offline features are integrated with current production features to form updated offline training datasets. For online feature serving, we utilize the User Profile Service (UPS), a newly developed dedicated system at Walmart designed explicitly for handling high-throughput, low-latency queries. Further details regarding feature development and serving are depicted in Figure 5.

Our subsequent A/B tests confirmed significant improvements in terms of AUC after incorporating these user engagement features. These results will be discussed further in the next section.

4.2. Conversion Model Prediction Spread Improvement with User Feature

In the initial iterations of the conversion model, user-specific features were not incorporated, resulting in challenges related to the concentration of predicted probability scores. Specifically, the model produced nearly identical conversion probabilities for certain line items, despite variations across dimensions such as zip code, time of day, and ad location.

Upon deeper analysis of feature distributions and XGBoost feature importance, we identified the root cause of this issue: the primary predictive engagement features, calculated at various granularities,

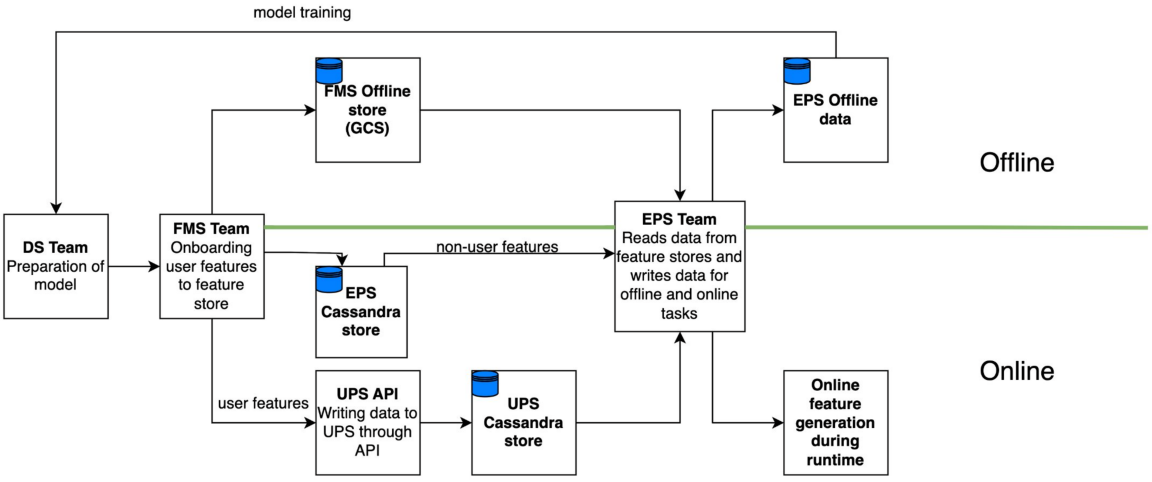


Figure 5: Offline/Online Serving orchestration for User Feature in Artemis System

remained constant throughout each day due to their daily update schedule. Additionally, another significant category of features—relevancy scores—often exhibited values at or near zero for these problematic line items. This feature distribution consequently caused predicted scores to cluster around a single point or within a small range.

To resolve this issue, incorporating user-level features emerged as a natural solution, aligning closely with the ranking system's core objective: selecting ads tailored to user preferences while maximizing platform profitability at the same time. By integrating user-specific features such as the amount of time spent in various departments and user visit frequency within different product categories, the model became more adept at identifying ads most likely to attract user interest and drive clicks or conversions.

Following the successful integration and offline evaluation of these user-level time-spent and visit-frequency features, as detailed previously in section 4.1, we observed a significant mitigation in prediction score concentration, resulting in a more varied and discriminative prediction distribution. Additionally, recent enhancements, including the integration of further user engagement and brand-category features, have further improved model performance, aligning with the strategies discussed earlier in section 4.1.

Table 2
User Engagement Feature and Brand Category Affinity Feature Offline Evaluation Result

Metric	Baseline (Current Production Features)	Baseline + User Engagement Features + Brand Category Affinity Features	Relative Lift
ROC-AUC	0.90000	0.90216	0.24%
PR-AUC	0.15000	0.16133	7.55%

Note: Metric values have been shifted by additive deltas for anonymization. Relative improvements (Lift) remain unchanged.

It is important to note that enhancing the performance of the conversion model, particularly with respect to evaluation metrics, is inherently more challenging. One contributing factor is that the existing production conversion model incorporates a broader set of features compared to the click model, such as ad conversion engagement metrics and user visit frequency. Given that the current conversion model has already achieved a high AUC, achieving further incremental improvements becomes increasingly

difficult.

Furthermore, producing prediction scores with greater variance provides significant benefits to downstream optimization systems by facilitating clearer distinctions among candidate ads, thus enabling more effective decision-making when selecting advertisements with the highest potential value.

5. EXPERIMENTATION

This section presents the outcomes of our online A/B experiments designed to evaluate the effectiveness of newly introduced user-level features in both click and conversion models. We report performance improvements across key metrics, including click-through rate (CTR), conversion rate (CVR), and return on ad spend (ROAS), based on production-scale testing on Walmart's internal platform.

5.1. Result of Click Model User Feature

The A/B test was conducted using Walmart internal testing platform, known as 'EXPO'. The experiment was conducted on the ad request session level. During ad serving, each user was randomly assigned to either a control or a variant group based on user ID hashing. Additionally, the Ads Serving team recorded a session-specific `SPA ID` to facilitate traffic identification during experiment analysis.

Our core success metric for this experiment was the empirical click-through rate (eCTR). We also monitored various guardrail metrics such as cost-per-click (CPC), cost-per-thousand impressions (CPM), total ad spend, and fill rate to ensure overall system health.

The experiment initially began with traffic allocations of 1% each for the control and variant buckets, subsequently increasing to 10%, 33.3%, and ultimately 50%. At each stage, statistically significant lifts in eCTR were observed. 10% relative eCTR lift was observed at full test traffic (50% control group vs. 50% experiment group) at 99% confident interval. Encouraged by these positive outcomes, the feature was launched into production. Post-launch monitoring has consistently demonstrated an upward trend in system-wide eCTR, achieving over 5% relative lift.

5.2. Result of Conversion Model

Conversion model A/B testing follows the same logic and procedure as described in section 5.1, with the control variant representing a nearly random assignment of auction winners. Upon deployment, the conversion optimization model demonstrated substantial performance improvements across key metrics, including up to a 7-times increase in the number of conversions, a 6-times increase in sales revenue, a 2-4 times improvement in empirical conversion rate (eCVR), and a 2-times enhancement in return on ad spend (ROAS) for 13 of the 17 tested campaign lines.

For the remaining 4 out of 17 under performing low-conversion campaign, the model either suffers from score concentration issue or inherently low conversion propensity. The former issue was addressed using the strategies outlined previously in section 4.2. For campaigns inherently resistant to conversion optimization, we implemented a conversion threshold criterion, requiring lines to accumulate a sufficient number of conversions before qualifying for conversion-based optimization. This threshold ensures that conversion optimization is applied effectively. We acknowledge that campaigns with limited historical conversions may not benefit from optimization regardless of model sophistication.

6. Future Direction

The development of Walmart's new in-house DSP model has provided a robust foundation upon which several promising areas for future exploration have emerged, particularly centered around advanced model architectures and further automation of pipelines.

A key area of interest moving forward is the integration and rigorous evaluation of deep learning-based ranking models (also known as Deep Rankers). Given the complex, sparse, and nonlinear nature of user-ad interactions, deep learning architectures, such as DeepFM, Wide & Deep, and xDeepFM

[14, 19, 20], could offer substantial promise. These models generally excel at capturing intricate patterns, latent interactions, and higher-order feature relationships, potentially delivering significant predictive accuracy improvements. Future work will involve deploying and thoroughly evaluating these deep architectures via extensive online A/B testing to quantify their incremental benefits and assess their computational feasibility in production environments.

Additionally, we aim to expand our DSP framework to include video advertisements alongside existing display ads. Video ads present unique challenges, such as modeling sequential viewer behavior, attention span dynamics, and engagement patterns, which deep learning methods are particularly well-suited to address, given their capability to handle complex, sequential, and contextual data signals.

Moreover, we plan to enhance our automated modeling framework by implementing continuous hyperparameter optimization through Airflow DAGs, improving logging and monitoring capabilities to rapidly diagnose performance issues, and achieving faster model deployment cycles by leveraging advanced serving frameworks like the PyTorch-based Deep Java Library (DJL) [21].

Finally, our long-term vision includes investigating sophisticated optimization strategies, such as real-time model personalization, online learning methods, and hybrid approaches combining traditional machine learning with deep learning paradigms. These advancements will ensure Walmart's DSP continues evolving in alignment with the cutting-edge developments in advertising technology.

7. Conclusion

In this paper, we presented Walmart's pioneering effort to build its first in-house Demand-Side Platform for ad ranking. We detailed the challenges encountered, the rationale behind critical implementation choices, and key innovations throughout the system development, emphasizing our effective utilization of XGBoost models integrated through robust engineering pipelines using MLEAP serialization and Airflow automation.

Our initial experimental results have demonstrated both the feasibility and effectiveness of this customized DSP solution, achieving measurable improvements in click-through rate and validating our approach. Additionally, initial assessments suggest promising opportunities to explore deep learning models, highlighting significant potential for future adoption of advanced deep ranking architectures to further enhance model performance.

Developing a DSP system from scratch is both challenging and rewarding. Through continuous innovation and rigorous experimentation, we anticipate that our DSP will substantially elevate ad relevance, improve customer engagement, and enhance advertising efficiency, ultimately aligning with Walmart's core mission of enabling customers to save money and live better. We believe the insights and methodologies presented in this paper will benefit both practitioners and researchers, marking a significant step forward in the advancement of advertising technology.

References

[1] P. Grigas, A. Lobos, Z. Wen, K. chih Lee, Profit maximization for online advertising demand-side platforms, 2017. URL: https://arxiv.org/abs/1706.01614. arXiv:1706.01614.

[2] D. Moriwaki, Y. Hayakawa, A. Matsui, Y. Saito, I. Munemasa, M. Shibata, A real-world implementation of unbiased lift-based bidding system, in: 2021 IEEE International Conference on Big Data (Big Data), IEEE, 2021, p. 1877–1888. URL: http://dx.doi.org/10.1109/BigData52589.2021.9671800. doi:10.1109/bigdata52589.2021.9671800.

[3] H. He, E. A. Garcia, Learning from imbalanced data, IEEE Trans. on Knowl. and Data Eng. 21 (2009) 1263–1284. URL: https://doi.org/10.1109/TKDE.2008.239. doi:10.1109/TKDE.2008.239.

[4] C. D. Manning, P. Raghavan, H. Schütze, Introduction to Information Retrieval, Cambridge University Press, 2008.

[5] Google Cloud, Cloud storage documentation, https://cloud.google.com/storage/docs, 2025. Accessed: 2025-04-16.

[6] A. Lakshman, P. Malik, Cassandra: a decentralized structured storage system, SIGOPS Oper. Syst. Rev. 44 (2010) 35–40. URL: https://doi.org/10.1145/1773912.1773922. doi:10.1145/1773912.1773922.

[7] T. Chen, C. Guestrin, Xgboost: A scalable tree boosting system, in: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '16, ACM, 2016, p. 785–794. URL: http://dx.doi.org/10.1145/2939672.2939785. doi:10.1145/2939672.2939785.

[8] F. M. Ojeda, M. L. Jansen, A. Thiéry, S. Blankenberg, C. Weimar, M. Schmid, A. Ziegler, Calibrating machine learning approaches for probability estimation: A comprehensive comparison, Statistics in Medicine 42 (2023) 5451–5478. URL: https://onlinelibrary.wiley.com/doi/abs/10.1002/sim.9921. doi:https://doi.org/10.1002/sim.9921. arXiv:https://onlinelibrary.wiley.com/doi/pdf/10.1002/sim.9921.

[9] A. Niculescu-Mizil, R. Caruana, Predicting good probabilities with supervised learning, in: Proceedings of the 22nd International Conference on Machine Learning, ICML '05, Association for Computing Machinery, New York, NY, USA, 2005, p. 625–632. URL: https://doi.org/10.1145/1102351.1102430. doi:10.1145/1102351.1102430.

[10] M. B. A. McDermott, H. Zhang, L. H. Hansen, G. Angelotti, J. Gallifant, A closer look at auroc and auprc under class imbalance, 2025. URL: https://arxiv.org/abs/2401.06091. arXiv:2401.06091.

[11] J. Davis, M. Goadrich, The relationship between precision-recall and roc curves, in: Proceedings of the 23rd International Conference on Machine Learning, ICML '06, Association for Computing Machinery, New York, NY, USA, 2006, p. 233–240. URL: https://doi.org/10.1145/1143844.1143874. doi:10.1145/1143844.1143874.

[12] A. P. Bradley, The use of the area under the roc curve in the evaluation of machine learning algorithms, Pattern Recognit. 30 (1997) 1145–1159. URL: https://api.semanticscholar.org/CorpusID:13806304.

[13] Combust.ml Team, Mleap documentation, https://combust.github.io/mleap-docs/, 2025. Accessed: 2025-04-16.

[14] H. Guo, R. Tang, Y. Ye, Z. Li, X. He, Deepfm: A factorization-machine based neural network for ctr prediction, 2017. URL: https://arxiv.org/abs/1703.04247. arXiv:1703.04247.

[15] J. Ma, Z. Zhao, X. Yi, J. Chen, L. Hong, E. H. Chi, Modeling task relationships in multi-task learning with multi-gate mixture-of-experts, in: Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, KDD '18, Association for Computing Machinery, New York, NY, USA, 2018, p. 1930–1939. URL: https://doi.org/10.1145/3219819.3220007. doi:10.1145/3219819.3220007.

[16] F. Wang, H. Gu, D. Li, T. Lu, P. Zhang, N. Gu, Towards deeper, lighter and interpretable cross network for ctr prediction, in: Proceedings of the 32nd ACM International Conference on Information and Knowledge Management, CIKM '23, Association for Computing Machinery, New York, NY, USA, 2023, p. 2523–2533. URL: https://doi.org/10.1145/3583780.3615089. doi:10.1145/3583780.3615089.

[17] C. Savard, N. Manganelli, B. Holzman, L. Gray, A. Perloff, K. Pedro, K. Stenson, K. Ulmer, Optimizing high throughput inference on graph neural networks at shared computing facilities with the nvidia triton inference server, 2023. URL: https://arxiv.org/abs/2312.06838. arXiv:2312.06838.

[18] Apache Software Foundation, Apache airflow documentation, https://airflow.apache.org/docs/, 2025. Accessed: 2025-04-16.

[19] H.-T. Cheng, L. Koc, J. Harmsen, T. Shaked, T. Chandra, H. Aradhye, G. Anderson, G. Corrado, W. Chai, M. Ispir, R. Anil, Z. Haque, L. Hong, V. Jain, X. Liu, H. Shah, Wide deep learning for recommender systems, 2016. URL: https://arxiv.org/abs/1606.07792. arXiv:1606.07792.

[20] R. Wang, B. Fu, G. Fu, M. Wang, Deep cross network for ad click predictions, 2017. URL: https://arxiv.org/abs/1708.05123. arXiv:1708.05123.

[21] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Köpf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, S. Chintala, Pytorch: An imperative style, high-performance deep learning library, 2019. URL: https://arxiv.org/abs/1912.01703. arXiv:1912.01703.