Product-Aware Architectures for Conversion Rate Prediction in E-Commerce Advertising

Aditya Chichani[1], Ameya Raul[1], Amey Porobo Dharwadker[1] and Saurabh Gupta[1]

[1]*Meta Platforms, Inc., USA*

Abstract

In e-commerce advertising, dynamic product ads display multiple products from an advertiser's catalog, and a ranking model must predict which product a given user is most likely to convert on. This is typically modeled as a conversion rate (CVR) prediction problem in which a deep neural network ingests a rich set of user, ad, and product features through a shared backbone architecture.

While user and ad features are strong predictors of the base conversion rate, they are constant across all candidate products within a ranking session. Product features are the only signals that vary across candidates and therefore the only signals that can discriminate between products. However, because shared backbone model architectures over-index on the stronger user and ad signals, the weaker but critically important product-level features are suppressed. Furthermore, due to the nature of ad conversions, the vast majority of training examples contain only a single product per user-ad session, which further limits the model's ability to learn product-specific patterns in the absence of contrastive examples.

We propose and evaluate three architectural interventions that elevate product signals in CVR prediction: (1) a *dedicated product tower* that processes product features through an isolated MLP with feature masking from the shared backbone; (2) an *additive logit combination* (LogitSumArch) that fuses product tower and ensemble predictions in logit space, preserving probability calibration; and (3) *pre-trained product embeddings* that provide learned product representations through the shared backbone.

We conduct extensive offline evaluation using normalized entropy and ranking metrics (pairwise accuracy, NDCG, MRR) on a large-scale industrial e-commerce advertising platform. Our combined approach (LogitSumArch with product embeddings) achieves the best overall performance, improving both pointwise and ranking metrics over a real world baseline, with feature importance analysis confirming that the product tower successfully elevates product-level signals to become top-ranked contributors.

Keywords

e-commerce advertising, conversion rate prediction, product ranking, multi-task learning, dynamic product ads

1. Introduction

E-commerce advertising platforms serve dynamic product ads (DPAs) that display multiple products from an advertiser's catalog to users based on predicted relevance. An advertiser uploads a product catalog, potentially containing millions of items, and the platform automatically selects and ranks a small subset for each user. These ads are typically rendered as horizontally scrollable carousels where each card displays a product with its image, title, price, and a call-to-action.

An important component is the *product ranking model*, which predicts the probability of conversion (e.g., purchase, add-to-cart) for each candidate product given a user and ad context. In industry, such models are typically deep neural networks trained on user interaction logs [1], with input features spanning three categories: *user features* (eg: engagement history), *ad features* (eg: advertiser metadata), and *product features* (eg: price, catalog attributes).

All features are typically processed through a shared backbone, and the model is trained with pointwise cross-entropy loss.

This design introduces a structural tension. User and ad features are strong predictors of the overall conversion rate, but they are constant across all candidate products within a single ranking session; only product features vary. Importantly, user and ad features cannot simply be removed: user features capture

ECOM'26: SIGIR Workshop on eCommerce, Jul 24, 2026, Melbourne, Australia

achichani@meta.com (A. Chichani); araul@meta.com (A. Raul); ameydhar@meta.com (A. P. Dharwadker); saurabg@meta.com (S. Gupta)

individual preferences (e.g., which product categories a user tends to purchase), and ad features capture contextual signals (e.g., whether certain ad formats or creative styles are better suited for particular products). These cross-feature interactions remain useful for product ranking, not just calibration. The challenge is therefore not that user and ad features are unhelpful, but that the shared architecture allows them to dominate at the expense of product-level signals. A model trained with pointwise loss can achieve low training error by primarily learning from user/ad signals, at the cost of product-level discrimination. We term this failure mode *product signal suppression* and observe it through three manifestations:

1. **Training data homogeneity.** Due to the sparsity of ad interactions, the vast majority of training examples contain only a single product per user-ad pair, providing no contrastive signal for product discrimination.
2. **Prediction collapse.** A significant fraction of ranking sessions produce near-identical predicted scores across products. The model assigns similar conversion probabilities to all products in the carousel, rendering within-ad product ranking arbitrary.
3. **Low feature importance.** Product features consistently rank low in feature importance, despite being the sole discriminative signal for within-ad ranking.

To address these challenges, we propose a product-aware ranking architecture with three complementary interventions:

- We identify and quantify the product signal suppression problem in industry-scale e-commerce CVR models, characterizing it through training-data homogeneity, prediction collapse, and feature importance.
- We propose three architectural interventions: a dedicated product tower with feature masking, additive logit combination, and pre-trained product embeddings enriching the shared backbone.
- We evaluate with both pointwise metrics (normalized entropy, calibration) and ranking metrics (pairwise accuracy, NDCG@k, MRR), demonstrating consistent improvements.
- We present feature importance analysis confirming that product features are elevated to top-ranked signals.

2. Related Work

Dynamic Product Ads and Product Ranking

Dynamic product ads present a unique ranking challenge: given a catalog of potentially millions of products, the system must select and rank a subset for each user in real time. Abutbul et al. [2] study audience prospecting for DPAs, focusing on predicting conversion rates and expanding advertiser reach. Their work addresses the *who to show* problem (audience selection), whereas we address the *what to show* problem (product ranking within an ad). Lobo et al. [3] study user behavior in carousel recommendation systems, finding that position bias significantly affects click patterns, which highlights the importance of accurate product ranking.

Multi-Task Learning for Recommendation

Industrial recommendation systems commonly employ multi-task learning (MTL) to jointly predict multiple conversion objectives over a shared backbone [4]. A long line of work studies how to share parameters across tasks without one task degrading another, from mixture-of-experts gating [5] to progressively separated expert layers [6], as well as mutual learning across task towers [7]. All of these govern how tasks share parameters. Our contribution is orthogonal and complementary: it concerns how a single feature group (product features) is routed and given dedicated capacity within whatever backbone is used.

Conversion Rate Prediction

CVR prediction is complicated by sparse and delayed conversion labels, since conversions are observed only for the small clicked subset of impressions. A substantial body of work mitigates the resulting selection bias by jointly modeling the click-to-conversion funnel: the entire-space multi-task model [8] estimates post-click CVR over the full impression space, and later variants add counterfactual objectives [9] and handle cascade delayed feedback [10]. This line of work largely targets cross-ad CVR estimation, predicting whether a user will convert on a given ad relative to other competing ads. Our problem arises at a later stage in the funnel: once an ad has already won the auction for a given user, we must decide which products from the advertiser's catalog to show within that ad. At this stage, user and ad context are held constant, and only product features vary, making the problem fundamentally one of within-ad product discrimination rather than cross-ad ranking.

Feature Interaction Architectures

Much industrial work improves how features interact: DCN-V2 [11] learns bounded-degree feature crosses at scale, and ensemble backbones combine several interaction modules in one network [12]. These methods enrich feature combination but do not address the structural imbalance we identify, in which strong user/ad signals dominate weak product signals. A complementary, more recent direction instead routes distinct feature groups through dedicated subnetworks so that dominant signals do not suppress weaker ones; DERM [13] does this for entity representations in ads ranking. Our product tower adopts this routing principle but adds two elements specific to within-ad product ranking: explicit masking, so the tower sees only product features, and a calibration-preserving additive logit combination rather than an unconstrained fusion.

Product Representations

Pre-trained product embeddings decompose the recommendation problem into representation learning and downstream modeling. Lake et al. [14] demonstrate large-scale collaborative filtering with product embeddings at Amazon. Gong et al. [15] present ItemSage at Pinterest, learning product embeddings capturing both behavioral and content-based signals. Our approach incorporates pre-trained product embeddings that capture both behavioral and content-based signals, enriching the shared backbone with learned product representations rather than routing them through a separate sparse module.

3. Problem Analysis

We analyze product signal suppression in a real world CVR model serving dynamic product ads at scale.

3.1. System Context

Given a user u, ad context a, and candidate product p, the model predicts $P(\text{conversion} \mid u, a, p)$. The baseline model is a multi-task deep neural network with hundreds of features. All features flow through a shared backbone (deep cross networks combined with shallow neural networks) with no dedicated capacity for product-level signal. Furthermore, the limited latency budgets for product ranking during overall ad serving disfavor heavyweight architectures such as cross-attention between products or listwise re-ranking passes. Any architectural intervention must add minimal inference cost per product, which motivates our choice of a lightweight MLP-based product tower and additive logit combination over more expressive but computationally expensive alternatives.

3.2. Conversion Data in Advertising

E-commerce advertising presents unique challenges for conversion modeling compared to search and organic recommendation settings.

In search and organic settings, the full user interaction funnel (impressions, clicks, and conversions) is typically captured on first-party platforms, providing rich behavioral signals at every stage. A unified model can jointly learn from click and conversion events, leveraging the abundance of click data to improve product discrimination even when conversions are sparse.

In advertising, however, this approach is not feasible. Conversions often occur off-platform (e.g., on an advertiser's website or app), relying on third-party tracking mechanisms. These signals are inherently noisier, subject to attribution windows, and incomplete due to tracking limitations and data restrictions. As a result, click-through rate (CTR) and conversion rate (CVR) are typically modeled separately: CTR models benefit from abundant first-party click data, while CVR models must contend with sparser, noisier conversion labels.

3.3. Training Data Sparsity

The separation of CTR and CVR modeling compounds the product signal suppression problem. The CVR model trains on post-click data, where each training example represents a product that was clicked within an ad. However, most users who click on an ad interact with only a single product, so the vast majority of training examples contain only one product per user-ad pair. The model rarely encounters contrastive examples where the same user-ad context is paired with different products having different conversion outcomes. Without such contrastive signal, the model learns to predict conversion primarily from user and ad features, treating product features as noise.

One might expect that incorporating earlier funnel stages (impressions, product views) would alleviate this sparsity by providing more product-level variation per session. However, impression-level data introduces its own challenges: it is orders of magnitude larger, heavily skewed toward non-engagement, and does not directly provide conversion labels. While we explore impression logging as a future direction (Section 8), the architectural interventions proposed in this paper operate within the existing post-click training paradigm.

3.4. Empirical Evidence

We observe two clear symptoms of product signal suppression in our baseline:

Prediction Homogeneity

The variance of predicted conversion scores across products within a ranking session is negligible for the majority of sessions, with a meaningful fraction producing completely identical predictions across all candidate products. This confirms that the model fails to discriminate between products, despite product features being the only varying input.

Feature Importance Gap

Feature importance analysis reveals that product features consistently rank low in importance, despite being the primary discriminative signal for within-ad ranking.

4. Proposed Approach

We propose three complementary architectural interventions. Figure 1 illustrates the overall architecture.

4.1. Product Tower with Feature Masking

We introduce a dedicated product tower: a multi-layer perceptron (MLP) that processes only product-level features, such as aggregate engagement statistics, price, and catalog metadata, producing a scalar product logit z_{prod}.

Feature masking ensures that non-product features (user and ad signals) are excluded from the product tower input. This forces the product tower to rely solely on product-level signals, preventing it from

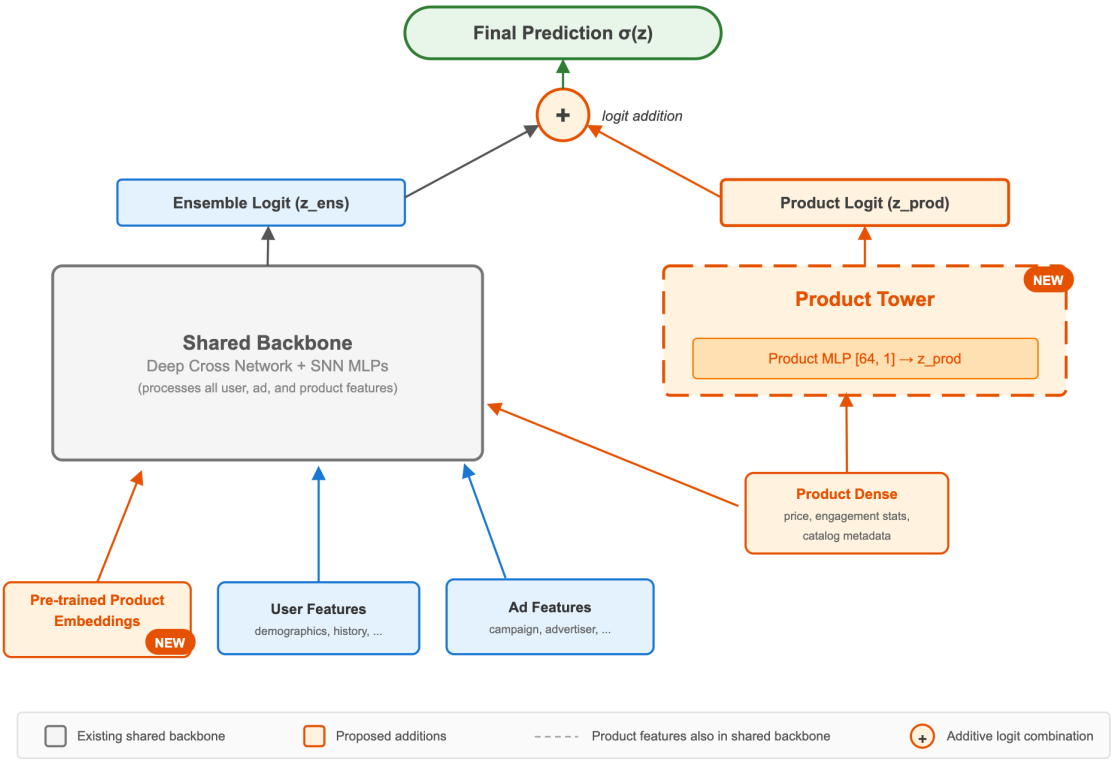

Figure 1: Proposed product-aware architecture. The shared backbone (gray) processes all features through deep cross networks and MLPs. The product tower (orange) processes only product dense features through a dedicated MLP, producing a product logit. Product embeddings (orange) flow through the shared backbone, enriching the common representation with learned product semantics. The final prediction combines the ensemble logit and product logit additively, preserving calibration.

learning shortcuts through user/ad features that would undermine its ability to discriminate between products.

4.2. Additive Logit Combination (LogitSumArch)

The product tower logit is combined with the ensemble logit from the shared backbone via addition:

$$z_{\text{final}} = z_{\text{ensemble}} + z_{\text{prod}} \tag{1}$$

where z_{ensemble} is the logit from the shared architecture and z_{prod} is the product tower output. The final prediction is $\hat{y} = \sigma(z_{\text{final}})$.

Unlike multiplicative or gating-based combination, logit addition preserves the probability scale and provides a natural residual connection. The product tower learns a product-specific *correction* to the base prediction:

$$P(\text{conv} \mid u, a, p) = \sigma\big(\underbrace{\log \tfrac{P_{\text{base}}}{1-P_{\text{base}}}}_{\text{ensemble logit}} + \underbrace{z_{\text{prod}}}_{\text{product correction}}\big) \tag{2}$$

This formulation allows the product tower to focus solely on product-level variation without re-learning user/ad patterns.

We also evaluate a multiplicative variant (MulArch) that combines the two predictions in probability space rather than in logit space:

$$\hat{y}_{\text{mul}} = \sigma(z_{\text{ensemble}}) \cdot \sigma(z_{\text{prod}}) \tag{3}$$

which is equivalent to gating the base prediction by a product-conditioned factor $\sigma(z_{\text{prod}}) \in (0, 1)$. Because this factor is strictly less than one, the multiplicative form can only deflate the base prediction; as we show in Section 6, this systematically distorts the probability scale and collapses calibration. We include it as a direct comparison to the additive formulation.

4.3. Product Embeddings

We augment the shared backbone with pre-trained product embeddings that capture both semantic and behavioral signals. These embeddings are produced by adapting a pretrained multimodal large language model to the e-commerce domain: the model jointly encodes each product's image and text, and is fine-tuned on large-scale user engagement (co-click) signals with a contrastive objective, so that products users engage with interchangeably are placed close together in the embedding space. This yields compact dense product vectors that bridge multimodal semantic understanding with behavioral shopping intent, along with derived discrete features (cluster and hierarchical semantic IDs) at multiple granularity levels. Unlike the product tower, which processes features through an isolated pathway, these embeddings flow through the shared backbone, enriching the common representation available to all task heads. The two are complementary: the product tower captures product-level engagement patterns from aggregate statistics, while the embeddings provide combined semantic and behavioral understanding of product identity and substitutability.

5. Experimental Setup

5.1. Dataset

We use real-world data from a large-scale e-commerce advertising platform. The data consists of user interaction logs including product impressions, clicks, and conversions for ads showing dynamic product catalogs. The evaluation is conducted on 100M+ scale dataset.

5.2. Model Variants

We evaluate the following variants, all trained on the same data with the same hyperparameters:

1. **Baseline:** Real-world model with all features processed through a shared backbone.
2. **MulArch:** Product tower combined with the ensemble via multiplicative combination of probabilities (Equation 3).
3. **LogitSumArch:** Product tower combined with the ensemble via additive logit combination (Equation 1).
4. **LogitSumArch + Product Embeddings:** LogitSumArch product tower combined with pre-trained product embeddings flowing through the shared backbone.

5.3. Evaluation Metrics

We evaluate using both pointwise and ranking metrics.

Pointwise Metrics

- **Normalized Entropy (NE)** measures cross-entropy loss normalized by the entropy of the marginal conversion rate [16], so that $\text{NE} = 1$ corresponds to a naive model predicting the average rate:

$$\text{NE} = \frac{-\frac{1}{N}\sum_{i=1}^{N}\left[y_i \log(p_i) + (1 - y_i)\log(1 - p_i)\right]}{-\left[\bar{p}\log\bar{p} + (1 - \bar{p})\log(1 - \bar{p})\right]} \tag{4}$$

  where p_i is the predicted probability, $y_i \in \{0, 1\}$ is the label, and $\bar{p} = \frac{1}{N}\sum_i y_i$ is the observed positive rate. Lower is better.

- **Calibration-free NE (CF-NE)** is computed identically to NE but after recalibrating predictions to match the observed positive rate, isolating discrimination ability from calibration.
- **Calibration** is the ratio of mean predicted probability to mean observed label:

$$\text{Cal} = \frac{\sum_{i=1}^{N} p_i}{\sum_{i=1}^{N} y_i} \tag{5}$$

  Perfect calibration yields $\text{Cal} = 1.0$.

Ranking Metrics

All ranking metrics are computed per multi-product session (sessions with ≥ 2 products) and averaged across sessions.

- **Pairwise Accuracy (PA)** is the fraction of concordant product pairs:

$$\text{PA} = \frac{\sum_{(i,j):y_i>y_j} \mathbf{1}[p_i > p_j]}{\sum_{(i,j):y_i>y_j} 1} \tag{6}$$

- **NDCG@k** ($k \in \{1, 3, 5\}$) measures ranking quality with position discounting:

$$\text{NDCG@}k = \frac{\text{DCG@}k}{\text{IDCG@}k}, \quad \text{DCG@}k = \sum_{i=1}^{k} \frac{2^{r_i} - 1}{\log_2(i+1)} \tag{7}$$

  where $r_i \in \{0, 1\}$ is the binary relevance (converted or not) at rank i, and $\text{IDCG@}k$ is the ideal DCG with items sorted by relevance.
- **MRR** is the mean reciprocal rank of the first converted product:

$$\text{MRR} = \frac{1}{|Q|} \sum_{q \in Q} \frac{1}{\text{rank}_q} \tag{8}$$

  where rank_q is the position of the first relevant item in session q.
- **Softmax CE Over Odds (SCE)** computes cross-entropy over within-session softmax-normalized odds, measuring the model's ability to concentrate probability on the correct product:

$$\text{SCE} = -\frac{1}{|Q|} \sum_{q \in Q} \sum_{j \in S_q} y_j \log \left(\frac{\exp(z_j)}{\sum_{k \in S_q} \exp(z_k)} \right) \tag{9}$$

  where $z_j = \log \frac{p_j}{1-p_j}$ are the predicted log-odds for product j in session S_q. Lower is better.

6. Results

6.1. Overall Performance

Table 1 presents relative improvements over the real-world baseline across all variants.[1]

6.2. Ablation: Combination Strategy

We compare multiplicative and additive combination strategies. The multiplicative variant (MulArch) causes severe calibration collapse: calibration degrades by 45%, and NE increases by over 50%. This occurs because multiplying probabilities systematically pushes predictions lower, distorting the probability scale.

LogitSumArch preserves calibration (within 0.3% of baseline) while achieving strong ranking improvements: pairwise accuracy improves by 0.09%, NDCG@1 by 0.15%, and MRR by 0.08%. The additive formulation's residual learning property, where the product tower learns a correction to the base prediction, leads to stable training dynamics.

[1] All NE improvements reported in this paper are statistically significant.

Table 1
Relative change (%) over baseline. ↓ indicates lower is better; ↑ indicates higher is better. Bold indicates best result per metric.

Variant	NE (↓)	Cali-Free NE (↓)	Calibration (↑)	Pairwise Acc. (↑)	NDCG@1 (↑)	NDCG@3 (↑)	NDCG@5 (↑)	MRR (↑)	Softmax CE (↓)
MulArch	+51.2	+94.4	−45.0	−1.28	−0.71	−0.45	−0.37	−0.36	+4.23
LogitSumArch	−0.04	+0.05	−0.27	**+0.09**	**+0.15**	**+0.08**	**+0.05**	**+0.08**	−1.78
LogitSum+ProdEmb	**−0.19**	**−0.30**	**−0.03**	**+0.09**	+0.13	+0.06	**+0.05**	**+0.08**	**−2.07**

6.3. Product Embeddings Contribution

The full proposed architecture configuration (LogitSumArch + Product Embeddings) combines the product tower with pre-trained product embeddings flowing through the shared backbone. This achieves the best NE (−0.19%), best calibration-free NE (−0.30%), and best softmax CE (−2.07%), while maintaining the ranking improvements from LogitSumArch. The product embeddings provide complementary value: the product tower captures engagement patterns through an isolated MLP, while product embeddings enrich the shared backbone with semantic product understanding. The combination achieves improvements across both pointwise and ranking metrics, confirming that feature isolation (product tower) and feature enrichment (product embeddings) are complementary strategies.

6.4. Feature Importance Analysis

To validate that our architecture successfully elevates product signals, we analyze feature importance across variants.

In the baseline, product features rank low in overall feature importance. With LogitSumArch, product feature share increases by approximately 5 percentage points, and the highest-ranked product feature rises significantly in the importance ordering. LogitSumArch + Product Embeddings achieves similar gains, confirming that the additive architecture provides genuine, balanced uplift in product signal utilization. In contrast, the multiplicative variant shows artificially inflated product importance due to mechanistic distortion in how it combines logits, rather than genuine signal utilization.

7. Discussion

Why Additive Combination?

Multiplicative combination can capture interaction effects but amplifies calibration errors. Additive logit combination provides a residual learning framework where the product tower learns a correction term. Because both terms lie on the same log-odds scale, no explicit normalization is needed to keep one from overwhelming the other: the ensemble logit carries the base rate and the product tower learns only a bounded product-specific correction on top of it. Our results confirm that additive combination achieves better ranking metrics while maintaining calibration.

Feature Masking Trade-offs

Feature masking excludes user and ad features from the product tower, ensuring it learns solely from product-level signals. Without masking, the product tower could learn to rely on user/ad features as shortcuts, undermining its ability to discriminate between products. Our results confirm that masking produces better ranking metrics by forcing the product tower to develop genuine product understanding rather than recapitulating the shared backbone's user/ad-driven predictions.

Scalability

The product tower adds minimal computational overhead (a single small MLP per product). Product embeddings leverage pre-trained representations and add negligible inference cost.

Limitations

Our evaluation is based on offline metrics. While offline ranking metrics correlate with online product ranking quality, ultimate validation requires online A/B testing, which we plan as future work. Additionally, our approach addresses the architectural aspect of product signal suppression; complementary data-side approaches (e.g., contrastive training data, ranking losses) remain as future directions.

8. Conclusion and Future Work

We presented a product-aware architecture for improving product-level CVR prediction in e-commerce advertising. By introducing a dedicated product tower with feature masking, an additive logit combination (LogitSumArch), and pre-trained product embeddings, we provide the model with dedicated capacity for product-level learning. Offline evaluation on a large-scale industrial e-commerce advertising platform demonstrates improvements in both pointwise and ranking metrics, and feature importance analysis confirms that product signals are elevated from bottom-ranked to top-ranked features.

Future directions include:

1. **Product impression logging:** Expanding training data to include all product impressions within clicked ads, providing contrastive examples for product discrimination.
2. **Product-centric ranking losses:** Introducing pairwise or listwise ranking losses that directly optimize within-session product ordering.
3. **Online evaluation:** A/B testing the combined approach to validate offline gains.

Declaration on Generative AI

During the preparation of this work, the author(s) used Claude (Anthropic) to improve writing style and drafting content. After using these tool(s)/service(s), the author(s) reviewed and edited the content as needed and take(s) full responsibility for the publication's content.

References

[1] G. Zhou, X. Zhu, C. Song, Y. Fan, H. Zhu, X. Ma, Y. Yan, J. Jin, H. Li, K. Gai, Deep interest network for click-through rate prediction, in: Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery & data mining, 2018, pp. 1059–1068.

[2] E. Abutbul, Y. Kaplan, N. Krasne, O. Somekh, O. David, O. Duvdevany, E. Segal, Audience prospecting for dynamic-product-ads in native advertising, in: arXiv preprint arXiv:2312.07160, 2023.

[3] S. de Leon-Martinez, Understanding user behavior in carousel recommendation systems for click modeling and learning to rank, in: Proceedings of the 17th ACM International Conference on Web Search and Data Mining, 2024, pp. 1139–1141.

[4] Y. Wang, H. T. Lam, Y. Wong, Z. Liu, X. Zhao, Y. Wang, B. Chen, H. Guo, R. Tang, Multi-task deep recommender systems: A survey, arXiv preprint arXiv:2302.03525 (2023).

[5] 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, 2018, pp. 1930–1939.

[6] H. Tang, J. Liu, M. Zhao, X. Gong, Progressive layered extraction (ple): A novel multi-task learning (mtl) model for personalized recommendations, in: Proceedings of the 14th ACM conference on recommender systems, 2020, pp. 269–278.

[7] Y. Ren, Y. Du, B. Wang, S. Zhang, Deep mutual learning across task towers for effective multi-task recommender learning, arXiv preprint arXiv:2309.10357 (2023).

[8] X. Ma, L. Zhao, G. Huang, Z. Wang, Z. Hu, X. Zhu, K. Gai, Entire space multi-task model: An effective approach for estimating post-click conversion rate, in: The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval, 2018, pp. 1137–1140.

[9] H. Wang, T.-W. Chang, T. Liu, J. Huang, Z. Chen, C. Yu, R. Li, W. Chu, Escm2: Entire space counterfactual multi-task model for post-click conversion rate estimation, in: Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval, 2022, pp. 363–372.

[10] Y. Zhao, X. Yan, X. Gui, S. Han, X.-R. Sheng, G. Yu, J. Chen, Z. Xu, B. Zheng, Entire space cascade delayed feedback modeling for effective conversion rate prediction, in: Proceedings of the 32nd ACM International Conference on Information and Knowledge Management, 2023, pp. 4981–4987.

[11] R. Wang, R. Shivanna, D. Cheng, S. Jain, D. Lin, L. Hong, E. Chi, Dcn v2: Improved deep & cross network and practical lessons for web-scale learning to rank systems, in: Proceedings of the web conference 2021, 2021, pp. 1785–1797.

[12] B. Zhang, L. Luo, X. Liu, J. Li, Z. Chen, W. Zhang, X. Wei, Y. Hao, M. Tsang, W. Wang, et al., Dhen: A deep and hierarchical ensemble network for large-scale click-through rate prediction, arXiv preprint arXiv:2203.11014 (2022).

[13] J. Liu, Y. Li, J. Sun, K. Li, H. Sun, S. Wang, H. Wu, S. Gao, P. Soares, N. Li, et al., Decoupled entity representation learning for pinterest ads ranking, in: Proceedings of the Nineteenth ACM Conference on Recommender Systems, 2025, pp. 940–944.

[14] T. Lake, S. A. Williamson, A. T. Hawk, C. C. Johnson, B. P. Wing, Large-scale collaborative filtering with product embeddings, arXiv preprint arXiv:1901.04321 (2019).

[15] P. Baltescu, H. Chen, N. Pancha, A. Zhai, J. Leskovec, C. Rosenberg, Itemsage: Learning product embeddings for shopping recommendations at pinterest, in: Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, 2022, pp. 2703–2711.

[16] X. He, J. Pan, O. Jin, T. Xu, B. Liu, T. Xu, Y. Shi, A. Atallah, R. Herbrich, S. Bowers, et al., Practical lessons from predicting clicks on ads at facebook, in: Proceedings of the eighth international workshop on data mining for online advertising, 2014, pp. 1–9.