CADET: Context-Conditioned Ads CTR Prediction With a Decoder-Only Transformer

David Pardoe*
Neil Daftary*
Miro Furtado*
Aditya Aiyer*
LinkedIn
Mountain View, California, USA
dpardoe@linkedin.com

Yu Wang
Liuqing Li
Tao Song
Lars Hertel†
LinkedIn
Mountain View, California, USA
ywang11@linkedin.com

Young Jin Yun
Senthil Radhakrishnan
Zhiwei Wang†
Tommy Li
LinkedIn
Mountain View, California, USA
yyun@linkedin.com

Khai Tran
Ananth Nagarajan
Ali Naqvi
Yue Zhang
LinkedIn
Mountain View, California, USA
khtran@linkedin.com

Renpeng Fang
Avi Romascanu
Arjun Kulothungun†
Deepak Kumar
LinkedIn
Mountain View, California, USA
refang@linkedin.com

Praneeth Boda
Fedor Borisyuk
Ruoyan Wang*
LinkedIn
Mountain View, California, USA
rnwang@linkedin.com

Abstract

Click-through rate (CTR) prediction is fundamental to online advertising systems. While Deep Learning Recommendation Models (DLRMs) with explicit feature interactions have long dominated this domain, recent advances in generative recommenders have shown promising results in content recommendation. However, adapting these transformer-based architectures to ads CTR prediction still presents unique challenges, including handling post-scoring contextual signals, maintaining offline-online consistency, and scaling to industrial workloads.

We present CADET (Context-Conditioned Ads Decoder-Only Transformer), an end-to-end decoder-only transformer for ads CTR prediction deployed at LinkedIn. Our approach introduces several key innovations: (1) a context-conditioned decoding architecture with multi-tower prediction heads that explicitly model post-scoring signals such as ad position, resolving the chicken-and-egg problem between predicted CTR and ranking; (2) a self-gated attention mechanism that stabilizes training by adaptively regulating information flow at both representation and interaction levels; (3) a timestamp-based variant of Rotary Position Embedding (RoPE) that captures temporal relationships across timescales from seconds to months; (4) session masking strategies that prevent the model from learning dependencies on unavailable in-session events, addressing train-serve skew; and (5) production engineering techniques including tensor packing and custom FlashAttention kernels that enable efficient training and serving at scale.

In online A/B testing, CADET achieves an 11.04% CTR lift compared to the production LiRank [3] baseline model, a hybrid ensemble of DCNv2 [13] and sequential encoders. The system has been successfully deployed on LinkedIn's advertising platform, serving the main traffic for homefeed sponsored updates.

CCS Concepts

• **Information systems** → **Computational advertising**; Online advertising; • **Computing methodologies** → *Neural networks*; *Deep learning*.

Keywords

Click-Through Rate Prediction, Generative Recommenders, Online Advertising

ACM Reference Format:
David Pardoe, Neil Daftary, Miro Furtado, Aditya Aiyer, Yu Wang, Liuqing Li, Tao Song, Lars Hertel, Young Jin Yun, Senthil Radhakrishnan, Zhiwei Wang, Tommy Li, Khai Tran, Ananth Nagarajan, Ali Naqvi, Yue Zhang, Renpeng Fang, Avi Romascanu, Arjun Kulothungun, Deepak Kumar, Praneeth Boda, Fedor Borisyuk, and Ruoyan Wang. 2026. CADET: Context-Conditioned Ads CTR Prediction With a Decoder-Only Transformer. In *Proceedings of AdKDD 2026 Workshop at KDD (AdKDD '26).* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 Introduction

LinkedIn is the world's largest professional social network, serving over one billion members globally, with a leading B2B-focused advertising platform. Homefeed sponsored updates represent the primary surface for LinkedIn ads, where click-through rate (CTR)

*Authors contributed equally to this research.
†Work done while at LinkedIn.

AdKDD '26, Jeju, South Korea

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

prediction serves as the core model powering ad delivery and determining which ads are displayed to users. The accuracy and efficiency of CTR prediction therefore directly impact both LinkedIn user experience and advertiser outcomes.

Historically, Deep Learning Recommendation Models (DLRMs) such as DCN [12, 13], DIN [18], and DHEN [17], which focus on explicit feature interaction learning, have dominated ads CTR prediction. These models typically operate on individual ad impressions with a large number of engineered features. Transformer-based sequence encoders have been integrated within DLRM architectures and successfully deployed in industrial systems [3, 14, 15]. More recently, HSTU-style generative recommenders [16] reformulated recommendation as a generative paradigm with decoder-only transformers as the sole backbone, and subsequent work such as OneRec [6] and MTGR [8] extends this paradigm to industrial-scale retrieval and ranking. A related challenge across ranking systems is handling post-scoring contextual signals such as ad position, which are unavailable at scoring time but influence user behavior; existing approaches treat position as a debiasing feature, apply counterfactual learning [1], or employ secondary factorized models [2].

However, adapting decoder-only transformers to production-grade ads CTR prediction still presents unique challenges. First, post-scoring contextual signals such as ad position significantly influence CTR but are unavailable at scoring time, creating a chicken-and-egg problem between predicted CTR and final ranking. Second, online recommendation systems require strict offline-online consistency, yet tracking delays and session-level scoring create gaps between training and serving conditions. Third, production environments demand training stability and reproducibility. Fourth, industrial workloads require efficient training and inference over billions of impressions daily.

In this paper, we present CADET (Context-Conditioned Ads Decoder-Only Transformer), an end-to-end decoder-only transformer for ads CTR prediction deployed at LinkedIn. Our approach addresses the above challenges through novel architectural innovations and production engineering techniques.

Our key contributions are as follows:

- **Context-conditioned decoding architecture**: A novel multi-tower decoding block that explicitly models post-scoring contextual signals (e.g., ad position) through K prediction heads, resolving the ranking dependency loop without iterative re-scoring.
- **Self-gated attention mechanism**: An improved self-attention design with representation-level and interaction-level gating that stabilizes training and mitigates pathological attention behaviors, enabling reliable convergence at scale.
- **Temporal encoding with modified RoPE**: A timestamp-based rotary positional encoding that captures time differences ranging from seconds to months, replacing sequence position with event timestamps for ads-specific temporal modeling.
- **Session masking for offline-online consistency**: Customized masking strategies that prevent the model from attending to unavailable in-session events during training, addressing train-serve skew caused by tracking delays.
- **Production engineering at scale**: A comprehensive set of optimizations including tensor packing and custom FlashAttention kernels for multi-item scoring that enable efficient training on large-scale datasets and low-latency serving.

2 Enhancing Decoder-Only Transformer for Ads System

2.1 Overall Architecture

Inspired by the HSTU ranking architecture [16], the full model processes interleaved ad impression and action sequences:

Input Sequence Formulation: The input consists of static user features followed by interleaved impression and contextualized action tokens:

$$\mathbf{x} = [M; I_1, (C_1, A_1); I_2, (C_2, A_2); \ldots; I_{L-1}, (C_{L-1}, A_{L-1}); I_L] \quad (1)$$

where M is user static features (e.g., profile), I_t concatenates ad, request, and other impression features, C_t is post-scoring contextual features unavailable at scoring time (e.g., ad position), and A_t is the action taken on impression t (e.g., click). A decoder-only transformer processes this sequence and predicts the action for every impression in a single pass via teacher forcing.

Context-Conditioned Decoding Block: The decoder-only transformer outputs a sequence of predictions paired with contextual metadata for each impression. For each impression, the decoding block produces K context-conditioned predictions:

$$\{\hat{y}_1, \hat{y}_2, \ldots, \hat{y}_K\} \quad (2)$$

where $\hat{y}_k$ represents the prediction for context bucket k (e.g., different ad positions). Through this context-conditioned decoding block, described in Section 2.3, the model learns to produce predictions conditioned on post-scoring contextual information.

2.2 CADET Transformer Block

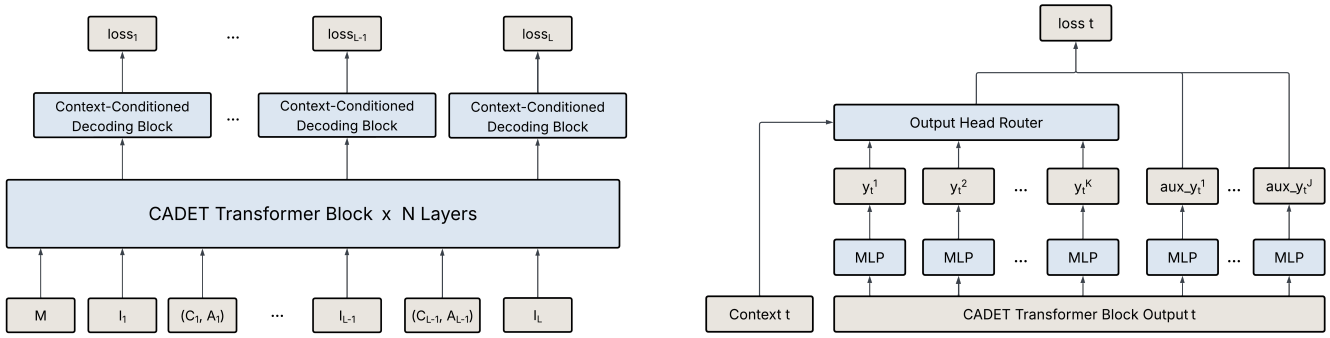
2.2.1 Self-Gated Multi-Head Attention. To improve training stability and mitigate pathological attention behaviors in our proposed model, we introduce self-gated attention [4] that adaptively regulates information flow at both representation and interaction levels; see Figure 2. Unlike output-level gating used in prior work [10, 16], our approach applies gating *before* attention computation on input representations and query-key projections.

Specifically, let $X \in \mathbb{R}^{B\times L\times d_{\text{model}}}$ denote the token representations at each layer of the decoder-only transformer, where B is batch size, L is sequence length and d_{model} is the model dimension. Queries Q and keys K are obtained through linear projections of X and preserve the same dimensionality, i.e., $Q, K \in \mathbb{R}^{B\times L\times d_{\text{model}}}$

$$Q = XW_Q \quad K = XW_K, \quad (3)$$

where $W_Q, W_K \in \mathbb{R}^{d_{\text{model}}\times d_{\text{model}}}$. We first apply a lightweight self-gating module to obtain gated representations, where the gate is conditioned on the representations themselves. This **representation-level gating** acts as adaptive feature selection, suppressing noisy or low-utility dimensions before attention computation, thereby improving activation scaling and gradient conditioning during training.

$$Gate(X) = \sigma(W_X^{\text{gate}} X), \quad \tilde{X} = X \odot Gate(X) \quad (4)$$



(a) Overall architecture **(b) Context-conditioned decoding block**

Figure 1: Architecture of the proposed model. (a) Overall architecture showing the interleaved impression-action sequence processing through the decoder-only transformer. (b) Detailed view of the context-conditioned decoding block with multiple prediction heads.

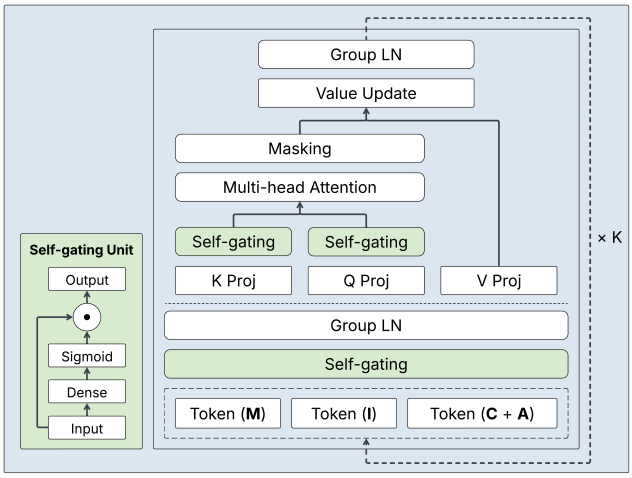


Figure 2: Detailed description of self-gated attention module

where $\sigma(\cdot)$ is the sigmoid function, $\odot$ denotes element-wise multiplication, and $W_X^{\text{gate}} \in \mathbb{R}^{d_{\text{model}} \times d_{\text{model}}}$.

In addition, we incorporate self-gating directly into the attention mechanism by modulating the projected queries Q and keys K. This **interaction-level gating** explicitly regulates query-key interactions by constraining dot-product magnitudes, preventing dominant tokens from monopolizing attention and effectively mitigating attention sink phenomena.

$$\begin{aligned} Gate(Q) &= \sigma(W_Q^{\text{gate}} Q), \quad \tilde{Q} = Q \odot Gate(Q) \\ Gate(K) &= \sigma(W_K^{\text{gate}} K), \quad \tilde{K} = K \odot Gate(K) \end{aligned} \tag{5}$$

where $W_Q^{\text{gate}}, W_K^{\text{gate}} \in \mathbb{R}^{d_{\text{model}} \times d_{\text{model}}}$.

2.2.2 Temporal Encoding. We encode temporal information using a modified form of Rotary Positional Embeddings (RoPE) [11]. In the original RoPE formulation, a query or key vector $\mathbf{x} \in \mathbb{R}^d$ is decomposed into $d/2$ two-dimensional components, and the i-th component is rotated by an angle

$$\alpha_i(p) = p \cdot \theta_i, \qquad \theta_i = 10000^{-\frac{2i}{d}},$$

where p denotes the item's sequence position. The dot product of rotated queries and keys then depends on their angular difference, letting the model learn relative position.

For ads modeling, we care more about differences in time than sequence position, so we replace position p with a Unix timestamp t. To handle time differences ranging from seconds to months, we define the rotation angle for the i-th component as

$$\alpha_i(t) = t \cdot \theta_i, \qquad \theta_i = \frac{\phi_{min}}{\Delta t_{\max}} \cdot base^{\frac{2i}{d}},$$

where $\Delta t_{\max}$ is the lookback window used in constructing sequences and thus the largest possible difference in timestamps. ϕ_{min} is a tunable parameter that lets us control what the smallest relative rotation (difference in α_0) would be for items with timestamps differing by $\Delta t_{\max}$, and $base$ is a tunable parameter that lets us control the highest frequency rotations that are useful for representing short-term information.

We do not add explicit information about sequence position, as causal transformers can learn this implicitly [9]. Crucially, this does not discard ordering: the causal mask still enforces sequence order, so within a dense burst of near-simultaneous events the model retains order while RoPE correctly contributes near-zero relative rotation.

2.2.3 Customized Masking. We employ customized time-based masking strategies during training and inference to ensure offline-online consistency and efficient candidate scoring. Here, "in-session" denotes not a hard session boundary but events within an impression's delay window Δ_{delay} ($t_i - \Delta_{\text{delay}} < t_j \le t_i$), which may not yet be observable at serving time; the masking below depends only on this threshold.

Inference-time Masking. At scoring time, the ranking model needs to rank hundreds of candidates simultaneously. Similar to previous work on generative recommenders [16], instead of scoring candidates individually, we append all candidates to the end of the

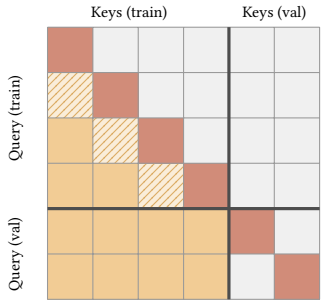


Figure 3: Session-aware masking strategies (depicted before item/action interleaving). Solid yellow cells are always unmasked. Striped cells fall within the Δ_{delay} threshold, masked during training, but available at inference. Gray cells are masked and dark red cells denote the preserved diagonal.

sequence and employ a special mask as shown in Figure 3 so that all candidates are scored in a single forward pass while remaining causally isolated from each other. This significantly improves evaluation and online inference speed.

Training Masking. Maintaining offline-online consistency is a critical challenge. In online ranking, recent activity may be unavailable due to system delay, simultaneous ranking across ad slots, or delayed interactions (e.g., a user clicking an ad seen earlier in the session). Attending to such same-session tokens would inflate offline metrics and create reliance on information unavailable at inference time.

To prevent the model from learning to depend on interactions that are unavailable at inference-time, we enforce session-aware masking during training. We apply a train-time mask M that prevents tokens from attending to same-session information:

$$M_{i,j} = \begin{cases} -\infty & \text{if } t_j > t_i - \Delta_{\text{delay}} \\ 0 & \text{otherwise} \end{cases} \tag{6}$$

where Δ_{delay} encodes the minimum latency we guarantee between an event and its observability online. We materialize this mask at runtime as shown in Figure 3. The final attention computation is:

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^\top}{\sqrt{d_k}} + M\right)V \tag{7}$$

2.3 Context-Conditioned Decoding Block

The decoding block addresses a fundamental challenge in ads recommendation: the gap between training-time observability and inference-time constraints. Critical contextual signals, such as final ad position or specific UI treatments, are often unknown during initial scoring. We resolve this by employing a multi-tower architecture that generates predictions conditioned on potential rendering contexts in a single forward pass.

2.3.1 Context-Conditioned Heads. To model behavior influenced by post-scoring context, we partition the context space into K discrete buckets based on historical click-through rates (e.g., $k = 1$ for positions 1–4 and $k = 2$ for positions 5+, the configuration we deploy). For each bucket $k \in \{1, \ldots, K\}$, we define a dedicated prediction head branching from the shared transformer output h_t:

$$\hat{y}_k = \text{MLP}_k(h_t) \tag{8}$$

where h_t is the transformer representation for impression t and $\hat{y}_k$ denotes the logit produced by the k-th head.

Training: During training, for each impression t, we observe its ground-truth context bucket $k_t \in \{1, \ldots, K\}$. Only the head corresponding to the realized context contributes to the loss:

$$\mathcal{L}_{\text{ctx}} = \sum_{t=1}^{L} \text{CE}\big(\hat{y}_{k_t},\, y_t\big) \tag{9}$$

where $\hat{y}_{k_t}$ denotes the prediction from the k_t-th head, and y_t is the ground-truth label. This selective routing ensures each head learns CTR for its specific context while eliminating train-serve mismatch by using position only for loss routing rather than as an input feature.

Serving: At inference time, all K heads produce predictions in parallel, yielding $\{\hat{y}_1, \ldots, \hat{y}_K\}$ for each candidate ad. During the online auction, when the serving system renders ads for a specific context, it selects the prediction from the corresponding bucket—e.g., $\hat{y}_1$ when filling positions 1–4, or $\hat{y}_2$ when filling positions 5+. This single-pass approach provides policy-ready signals that resolve the ranking dependency loop without the latency cost of iterative re-ranking or multi-pass feature toggling.

Historical Context Information: As described in Section 2.1, contextual signals C_t are paired with actions A_t rather than included in impression features I_t. This design prevents information leakage during training while enabling the model to learn historical behavior patterns conditioned on context.

2.3.2 Auxiliary Action Heads. To improve representation learning, we introduce auxiliary prediction tasks for fine-grained user actions, including click type classification, landing-page dwell time, and impression duration regression. Each auxiliary head $\hat{y}_j^{\text{aux}} = \text{MLP}_j(h_t)$ is a shared prediction module used solely for training-time regularization and does not participate in inference.

2.3.3 Loss Construction. The total loss combines the context-conditioned and auxiliary heads:

$$\mathcal{L} = \lambda_{\text{ctx}}\mathcal{L}_{\text{ctx}} + \sum_{j=1}^{J} \lambda_j \mathcal{L}_j^{\text{aux}} \tag{10}$$

where J is the number of auxiliary tasks, $\mathcal{L}_j^{\text{aux}}$ denotes the auxiliary loss for task j, and $\{\lambda_{\text{ctx}}, \lambda_j\}$ are hyperparameters balancing the terms.

3 Productionization and Scaling

3.1 Training System

We train on NVIDIA H200 GPUs using PyTorch Hybrid Sharded Data Parallel (HSDP) and FlexAttention [7] to support customized masking.

The autoregressive paradigm let us densify training data. We migrated from a legacy pointwise format, where each impression was an independent record with over 100 features, to a user-centric

sequence format; aggregating impressions and actions into unified sequences and pruning to high-utility signals substantially reduces the data footprint while preserving accuracy (see Section 4.2).

3.1.1 Packing. A common approach to training sequence models materializes a dense (B, L, d_{model}) tensor, but since attention is the only operation requiring explicit sequence structure, the majority of activation memory and MLP compute is wasted on padding tokens.

To densify training, we make the packed representation the canonical form throughout training and validation. The dataloader compacts every batch into a contiguous token buffer of shape (`total_tokens`, d_{model}) together with prefix-sum offsets and sequence-length metadata. Each worker aggregates user sequences until the packed length approaches a configured budget, pads the final few tokens to hit the exact target, and emits the batch—fixing `total_tokens` per batch is critical for stability and `torch.compile`.

3.2 Serving System

Each inference request scores multiple candidate ads against a shared user context (Section 2.1) in a single forward pass—the multi-item scoring pattern described in Section 2.2.3—within a p99 latency budget of 50 ms. Because this pattern is simpler than the session masking used during training, we develop a custom FlashAttention [5] kernel that exploits its sparsity, integrating the masking logic directly into the tiled attention computation and avoiding mask materialization. The shared-context attention shape is $L^2/2+LN$ versus a standard $(L+N)^2$—linear rather than quadratic in the number of candidates.

4 Experiments and Results

Here we present the results of offline experiments designed to illustrate the contributions of various components of the CADET model, as well as online A/B test results. All experiments are conducted on LinkedIn's sponsored updates ad click dataset.

4.1 Ablation Studies on CADET Components

Table 1 shows the results of ablation studies on various model components, measured in relative ROC AUC change compared to the full CADET model. All experiments are performed on one year of data with the final day used for validation and a Δ_{delay} of one hour. We use $n_{\text{layers}} = 8$, $n_{\text{heads}} = 4$, $d_{\text{model}} = 352$, and sequence length $L = 4096$.

4.1.1 Self-Gated Multi-Head Attention. We study the effect of self-gating on training dynamics by comparing a baseline without gating, representation-level gating only, and the combined representation- and interaction-level configuration. Across multiple runs, the baseline exhibits clear instability with sharp early-training AUC drops; representation-level gating smooths the trajectory by reducing the frequency and magnitude of these drops; and the combined configuration yields the most stable convergence.

4.1.2 Context-Conditioned Modeling. We use LinkedIn feed position as the context signal in experiments. We set $K = 2$ (positions 1–4, and 5+) to simplify integration with the downstream auction system while capturing the primary position-dependent CTR variation. Removing context-conditioned modeling significantly hurts performance, reducing AUC by 0.61%.

4.1.3 Temporal RoPE. We used $\Delta t_{\max} = 1$ year in milliseconds, $\phi_{min} = 10^{-4}$, and $base = 6 \times 10^5$. Removing RoPE applied to impression timestamps reduces AUC by 0.13%, showing that the model is able to learn the importance of time between impressions.

4.1.4 Session Masking. In this experiment, we retain the Δ_{delay} of one hour when creating attention masks for validation tokens, but set Δ_{delay} to zero otherwise, meaning that during training we allow tokens to attend to any previous token. The drop of 0.31% in AUC shows that the model is indeed harmed by allowing it access to information at training time that would not be available during inference due to latency in updating sequences.

Model Variant	ΔAUC
full CADET	—
- context-conditioned	−0.61%
- temporal RoPE	−0.13%
- session masking	−0.31%

Table 1: Ablation studies on CADET components.

We additionally compare against HSTU-style baselines (Table 2). *CADET-HSTU* replaces the Transformer backbone with HSTU while keeping the rest of CADET's design (context-conditioned scoring, self-gating, timestamp RoPE); *CADET-Transformer* is our default backbone. Both variants are reported without session masking because session masking is not readily supported by Meta's released HSTU Triton kernel. The two backbones perform within 0.0001 AUC of each other, indicating CADET's gains come from its modeling components rather than the backbone choice; full CADET (with session masking) reaches 0.8554.

Model Variant	AUC
HSTU-Base	0.8497
HSTU-Large	0.8505
CADET-HSTU (w/o session masking)	0.8535
CADET-Transformer (w/o session masking)	0.8534
Full CADET	**0.8554**

Table 2: Comparison with HSTU baselines and backbone-swapped CADET variants.

4.2 Scaling and Efficiency

Migrating from the legacy pointwise format to user-centric sequences with 12 high-utility signals reduced our data footprint by 98% while preserving accuracy through prioritization of temporal behavioral signals. Packing eliminates padding tokens, increasing effective batch size by approximately 4× given the right-skewed user-sequence distribution. Our custom FlashAttention kernel achieves a 3× speedup over fused multi-head attention (FMHA), reaching 330 requests per second per GPU at p99 < 50 ms.

4.3 Online Results

We evaluated CADET through large-scale online A/B testing on LinkedIn's homefeed sponsored updates against the production **LiRank** [3] baseline—a hybrid ensemble of **DCNv2** [13] for feature

interactions and **TransAct** [14] for sequential history modeling, which we replaced entirely with the unified CADET model. CADET achieved a statistically significant **+11.04%** lift in CTR over the LiRank production baseline, demonstrating that a unified generative approach can outperform a highly optimized multi-component hybrid system. Because LinkedIn advertisers typically spend their full budgets under auto-bidding, revenue remained approximately neutral. These relevance gains translated into stronger advertiser ROI, with cost-per-click (CPC) falling 10.9%.

5 Conclusion

We presented CADET, an end-to-end decoder-only transformer for large-scale ads CTR prediction at LinkedIn. This work addressed key challenges in applying autoregressive transformers to advertising: self-gated attention stabilizes optimization, timestamp-based RoPE captures temporal relationships across timescales, session-aware masking ensures offline-online consistency, and context-conditioned decoding resolves the chicken-and-egg problem between CTR prediction and post-scoring signals.

Beyond modeling innovations, productionization techniques including tensor packing and custom FlashAttention kernels enable efficient training and low-latency serving. Empirically, CADET achieves 11.04% CTR lift over our LiRank baseline [3], showing that a unified decoder-only transformer, combined with targeted modeling and production optimizations, can effectively outperform DLRM-style ensemble models in industrial ads ranking.

We leave to future work the extension to multiple concurrent post-scoring signals beyond position and evaluation on public benchmarks that include such context.

Acknowledgments

We thank Oren Sar Shalom, Yuan Gao and Franklin Graves for their valuable discussions and paper reviews. We also thank Guandong Zhu, Rutvij Mehta, Serhat Leloglu, Zeyu Jin, Shuzhe Xiao, Oz Ozkan, Hina Arora, Shao Tang, Alex Berlingeri, Ankur Agrawal, Emily Ki, Hongzhou Li, Joe Cho, Qian Li, Siva Popuri, Tiffany Zhou, Chen Zhu, Haoyue Tang, Jaideep Ray, Yang Pei, Yujie Ai, Sumit Rangwala, and Xiaoling Zhai for their collaboration and contributions to this work. Finally, we thank Sen Zhou, Xianxing Zhang, Julie Choi, Manas Apte, Yitong Zhou, and Sriram Sankar for their guidance and leadership support.

References

[1] Aman Agarwal, Kenta Takatsu, Ivan Zaitsev, and Thorsten Joachims. 2019. A General Framework for Counterfactual Learning-to-Rank. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*. 5–14. https://doi.org/10.1145/3331184.3331202

[2] Rohan Anil, Sandra Gadanho, Da Huang, Nijith Jacob, Zhuoshu Li, Dong Lin, Todd Phillips, Cristina Pop, Kevin Regan, Gil I. Shamir, Rakesh Shivanna, and Qiqi Yan. 2022. On the Factory Floor: ML Engineering for Industrial-Scale Ads Recommendation Models. In *ORSUM@RecSys 2022*. https://ceur-ws.org/Vol-3303/paper6.pdf

[3] Fedor Borisyuk, Mingzhou Zhou, Qingquan Song, Siyu Zhu, Birjodh Tiwana, Ganesh Parameswaran, Siddharth Dangi, Lars Hertel, Qiang Charles Xiao, Xiaochen Hou, Yunbo Ouyang, Aman Gupta, Sheallika Singh, Dan Liu, Hailing Cheng, Lei Le, Jonathan Hung, Sathiya Keerthi, Ruoyan Wang, Fengyu Zhang, Mohit Kothari, Chen Zhu, Daqi Sun, Yun Dai, Xun Luan, Sirou Zhu, Zhiwei Wang, Neil Daftary, Qianqi Shen, Chengming Jiang, Haichao Wei, Maneesh Varshney, Amol Ghoting, and Souvik Ghosh. 2024. LiRank: Industrial Large Scale Ranking Models at LinkedIn. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD '24)*. 4804–4815. https://doi.org/10.1145/3637528.3671561

[4] Yekun Chai, Shuo Jin, and Xinwen Hou. 2020. Highway Transformer: Self-Gating Enhanced Self-Attentive Networks. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, Dan Jurafsky, Joyce Chai, Natalie Schluter, and Joel Tetreault (Eds.). Association for Computational Linguistics, Online, 6887–6900. https://doi.org/10.18653/v1/2020.acl-main.616

[5] Tri Dao. 2024. FlashAttention-3: Fast and Accurate Attention with Asynchrony and Low-precision. *arXiv preprint arXiv:2407.08608* (2024).

[6] Jiaxin Deng, Shiyao Wang, Kuo Cai, Lejian Ren, Qigen Hu, Weifeng Ding, Qiang Luo, and Guorui Zhou. 2025. OneRec: Unifying Retrieve and Rank with Generative Recommender and Iterative Preference Alignment. arXiv:2502.18965 [cs.IR] https://arxiv.org/abs/2502.18965

[7] Juechu Dong, Boyuan Feng, Driss Guessous, Yanbo Liang, and Horace He. 2024. Flex Attention: A Programming Model for Generating Optimized Attention Kernels. arXiv:2412.05496 [cs.LG] https://arxiv.org/abs/2412.05496

[8] Ruidong Han, Bin Yin, Shangyu Chen, He Jiang, Fei Jiang, Xiang Li, Chi Ma, Mincong Huang, Xiaoguang Li, Chunzhen Jing, Yueming Han, MengLei Zhou, Lei Yu, Chuan Liu, and Wei Lin. 2025. MTGR: Industrial-Scale Generative Recommendation Framework in Meituan. In *Proceedings of the 34th ACM International Conference on Information and Knowledge Management (CIKM '25)*. 5731–5738. https://doi.org/10.1145/3746252.3761565

[9] Amirhossein Kazemnejad, Inkit Padhi, Karthikeyan Natesan Ramamurthy, Payel Das, and Siva Reddy. 2023. The Impact of Positional Encoding on Length Generalization in Transformers. In *Proceedings of the 37th International Conference on Neural Information Processing Systems (NIPS '23)*. Article 1082.

[10] Zihan Qiu, Zekun Wang, Bo Zheng, Zeyu Huang, Kaiyue Wen, Songlin Yang, Rui Men, Le Yu, Fei Huang, Suozhi Huang, Dayiheng Liu, Jingren Zhou, and Junyang Lin. 2025. Gated Attention for Large Language Models: Non-linearity, Sparsity, and Attention-Sink-Free. In *Proceedings of the 39th Annual Conference on Neural Information Processing Systems (NeurIPS '25)*.

[11] Jianlin Su, Murtadha Ahmed, Yu Lu, Shengfeng Pan, Wen Bo, and Yunfeng Liu. 2024. RoFormer: Enhanced Transformer with Rotary Position Embedding. *Neurocomputing* 568 (2024), 127063. https://doi.org/10.1016/j.neucom.2023.127063

[12] Ruoxi Wang, Bin Fu, Gang Fu, and Mingliang Wang. 2017. Deep & Cross Network for Ad Click Predictions. In *Proceedings of the ADKDD'17 (ADKDD'17)*. Article 12, 7 pages. https://doi.org/10.1145/3124749.3124754

[13] Ruoxi Wang, Rakesh Shivanna, Derek Cheng, Sagar Jain, Dong Lin, Lichan Hong, and Ed Chi. 2021. DCN V2: Improved Deep & Cross Network and Practical Lessons for Web-scale Learning to Rank Systems. In *Proceedings of the Web Conference 2021 (WWW '21)*. 1785–1797. https://doi.org/10.1145/3442381.3450078

[14] Xue Xia, Pong Eksombatchai, Nikil Pancha, Dhruvil Deven Badani, Po-Wei Wang, Neng Gu, Saurabh Vishwas Joshi, Nazanin Farahpour, Zhiyuan Zhang, and Andrew Zhai. 2023. TransAct: Transformer-based Realtime User Action Model for Recommendation at Pinterest. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD '23)*. 5249–5259. https://doi.org/10.1145/3580305.3599918

[15] Zhichen Zeng, Xiaolong Liu, Mengyue Hang, Xiaoyi Liu, Qinghai Zhou, Chaofei Yang, Yiqun Liu, Yichen Ruan, Laming Chen, Yuxin Chen, Yujia Hao, Jiaqi Xu, Jade Nie, Xi Liu, Buyun Zhang, Wei Wen, Siyang Yuan, Hang Yin, Xin Zhang, Kai Wang, Wen-Yen Chen, Yiping Han, Huayu Li, Chunzhi Yang, Bo Long, Philip S. Yu, Hanghang Tong, and Jiyan Yang. 2025. InterFormer: Effective Heterogeneous Interaction Learning for Click-Through Rate Prediction. In *Proceedings of the 34th ACM International Conference on Information and Knowledge Management (CIKM '25)*. 6225–6233. https://doi.org/10.1145/3746252.3761527

[16] Jiaqi Zhai, Lucy Liao, Xing Liu, Yueming Wang, Rui Li, Xuan Cao, Leon Gao, Zhaojie Gong, Fangda Gu, Jiayuan He, Yinghai Lu, and Yu Shi. 2024. Actions Speak Louder than Words: Trillion-Parameter Sequential Transducers for Generative Recommendations. In *Proceedings of the 41st International Conference on Machine Learning (ICML '24)*. PMLR, 58484–58509. https://proceedings.mlr.press/v235/zhai24a.html

[17] Buyun Zhang, Liang Luo, Xi Liu, Jay Li, Zeliang Chen, Weilin Zhang, Xiaohan Wei, Yuchen Hao, Michael Tsang, Wenjun Wang, Yang Liu, Huayu Li, Yasmine Badr, Jongsoo Park, Jiyan Yang, Dheevatsa Mudigere, and Ellie Wen. 2022. DHEN: A Deep and Hierarchical Ensemble Network for Large-Scale Click-Through Rate Prediction. arXiv:2203.11014 [cs.IR] https://arxiv.org/abs/2203.11014

[18] Guorui Zhou, Xiaoqiang Zhu, Chenru Song, Ying Fan, Han Zhu, Xiao Ma, Yanghui Yan, Junqi Jin, Han Li, and Kun Gai. 2018. Deep Interest Network for Click-Through Rate Prediction. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining (KDD '18)*. 1059–1068. https://doi.org/10.1145/3219819.3219823

APPENDIX

A Experimental Setup

This appendix consolidates the model, data, training, and evaluation configuration used throughout the paper.

Data. All experiments use LinkedIn's homefeed sponsored-updates ad-click dataset. Unless otherwise noted, we train on approximately one year of interactions and hold out the final day for validation, using a delay threshold Δ_{delay} of one hour. Each user is encoded as static features M followed by an interleaved sequence of impression and contextualized-action tokens; we retain 12 high-utility signals per token, down from over 100 features in the legacy pointwise format.

Model. CADET is a decoder-only transformer with $n_{\text{layers}} = 8$, $n_{\text{heads}} = 4$, model dimension $d_{\text{model}} = 352$, and maximum sequence length $L = 4096$. Every layer applies self-gated attention with both representation- and interaction-level gating. Temporal information uses the timestamp-based RoPE with $\Delta t_{\max} = 1$ year (in milliseconds), $\phi_{min} = 10^{-4}$, and $base = 6 \times 10^5$. The context-conditioned decoding block uses $K = 2$ position buckets (positions 1–4 and 5+). Auxiliary heads predict click type, landing-page dwell time, and impression duration, and are used only at training time. The training objective is $\mathcal{L} = \lambda_{\text{ctx}}\mathcal{L}_{\text{ctx}} + \sum_j \lambda_j \mathcal{L}_j^{\text{aux}}$.

Training. Models are trained on NVIDIA H200 GPUs with PyTorch Hybrid Sharded Data Parallel (HSDP) and FlexAttention [7] to support session-aware masking. Each batch uses the packed $(\texttt{total\_tokens}, d_{\text{model}})$ representation with a fixed token budget, which is required for stable `torch.compile`.

Serving and evaluation. At inference, each request scores multiple candidate ads against a shared user context in a single forward pass, served by a custom FlashAttention kernel within a p99 latency budget of 50 ms (~330 requests per second per GPU). This kernel is purely a serving-latency optimization: all training and offline evaluation use stock FlexAttention, and none of the accuracy results in the paper depend on it, so CADET's modeling contributions reproduce without any bespoke attention code. Offline quality is reported as relative ROC AUC, and ablations as relative ΔAUC against the full CADET model.