# Isotonic Layer: A Universal Framework for Generic Recommendation Debiasing

Hailing Cheng
haicheng@linkedin.com
Linkedin Inc
Mountain View, California, USA

Yafang Yang
yafyang@linkedin.com
Linkedin Inc
Mountain View, California, USA

Hemeng Tao
hetao@linkedin.com
Linkedin Inc
Mountain View, California, USA

Fengyu Zhang
fezhang@linkedin.com
Linkedin Inc
Mountain View, California, USA

## Abstract

Model calibration and debiasing are fundamental to ensuring the reliability and fairness of large-scale recommendation systems. While traditional methods like **Platt Scaling** [15] offer functional smoothness and **Isotonic Regression** [6] provides non-parametric flexibility, both paradigms struggle to integrate into modern end-to-end deep learning pipelines: the former is often too restrictive for complex distributions, while the latter lacks the differentiability required for gradient-based optimization.

In this paper, we introduce the **Isotonic Layer**[1], a novel, differentiable framework that integrates piecewise linear fitting directly into neural architectures. By partitioning the feature space into discrete segments and optimizing non-negative slopes via a constrained dot-product mechanism, we instantiate a global monotonic inductive bias. Unlike standard neural layers, which lack inter-segment coordination, our approach enforces strict order properties, ensuring that model outputs remain logically consistent with critical features such as latent relevance, recency, or quality scores.

We further generalize this architecture by parameterizing segment-wise slopes as learnable embeddings. This enables the model to adaptively capture nuanced, context-specific distortions—such as position-based click-through rate (CTR) bias[4]—through specialized isotonic profiles. At its core, our approach utilizes a dual-task formulation that decouples the recommendation objective into (1) **latent relevance estimation** and (2) **bias-aware calibration**. By reframing debiasing as a differentiable score-calibration challenge, the Isotonic Layer serves as a functional bridge mapping biased observations back to their true underlying utility.

A major contribution of this work is the framework's ability to perform highly granular, customized calibration. By learning embeddings for diverse context features—including display positions, device types, advertiser IDs, and objective types—the Isotonic Layer can achieve tailored calibration for arbitrary combinations of context features. This level of fine-grained control (e.g., per-advertiser calibration) is virtually impossible with traditional non-parametric methods. To address the inherent heterogeneity of modern systems, we extend this framework to **Multi-Task Learning (MTL)** environments with dedicated embeddings for distinct objectives. Extensive empirical evaluations on real-world datasets and large-scale production A/B tests demonstrate that the Isotonic Layer effectively mitigates systematic bias and enhances calibration fidelity without the risk of underfitting. Our approach significantly outperforms state-of-the-art baselines in both predictive accuracy and ranking consistency.

## CCS Concepts

• **Information systems → Recommender systems**.

## Keywords

Multitask Learning, Isotonic Layer, Model Debiasing

**ACM Reference Format:**
Hailing Cheng, Yafang Yang, Hemeng Tao, and Fengyu Zhang. 2026. Isotonic Layer: A Universal Framework for Generic Recommendation Debiasing. In *Proceedings of Make sure to enter the correct conference title from your rights confirmation email (Conference acronym 'XX)*. ACM, New York, NY, USA, 10 pages. https://doi.org/XXXXXXX.XXXXXXX

## 1 Introduction

In modern large-scale recommendation systems, the predicted probabilities generated by **Deep Neural Networks (DNNs)** often diverge from pure latent user preferences. These signals are typically confounded by systemic factors, most notably position bias, presentation bias, and selection bias. While the industry has historically addressed these challenges through post-hoc calibration and debiasing, a fundamental architectural tension persists.

Traditional non-parametric methods, such as **Isotonic Regression** [6], provide indispensable monotonic guarantees—ensuring that higher predicted relevance corresponds to a higher event probability—but are notoriously difficult to integrate into end-to-end, gradient-based learning. Conversely, standard deep learning layers offer high representational flexibility but lack the global constraints necessary to enforce logically ordered relationships between inputs and outputs.

[1]https://github.com/hailingc/Isotonic-Layer


*Conference acronym 'XX, Woodstock, NY*

ACM ISBN 978-1-4503-XXXX-X/2026/06
https://doi.org/XXXXXXX.XXXXXXX

Previous researchers have explored integrating monotonic properties within deep learning frameworks. For instance, Shen et al. [16] introduced *Isotonic Embedding* to represent promotional activities monotonically. However, such approaches primarily utilize the monotonic property of the embeddings without exploring their broader fitting capabilities or utilizing them as universal calibration tools.

To the best of our knowledge, this work is the first to systematically integrate isotonic regression into the deep learning architecture by formulating it as a universal **Isotonic Calibration Layer**. By treating debiasing as an end-to-end calibration task, we present a generic framework that maintains structural monotonicity while leveraging the full representational power of deep ranking models.

## 1.1 The Necessity of Monotonicity

Many recommendation scenarios rely on domain-specific priors regarding the relationship between features and outcomes. For instance, in "feed quality" modeling, an increase in the quality score should, *ceterisparibus*, result in a strictly higher model output. However, in standard deep learning architectures, parameters are updated locally via backpropagation without regard for global order. This lack of architectural coordination often results in "inversion errors"—where Quality Level 5 is scored lower than Quality Level 4 due to data noise—leading to inconsistent rankings and poor generalization in production environments.

## 1.2 The Challenge of Task Heterogeneity

The complexity of debiasing in industrial recommendation engines is compounded by Multi-Task Learning (MTL). Systems must simultaneously optimize for diverse user behaviors with heterogeneous bias profiles; for instance, a "click" is far more sensitive to position and visual prominence than a "purchase." Current frameworks often fail by treating model outputs as a single, homogeneous score, ignoring these task-specific distortions. Specifically:

- **Rigid Parametric Methods:** Techniques like Platt Scaling lack the capacity to model varying distortion profiles across tasks.
- **Standard Neural Layers:** These lack the global inductive bias necessary to prevent cross-task "inversion" noise.

## 1.3 Proposed Solution: The Isotonic Layer

To address these limitations, we introduce the Isotonic Layer, a differentiable, "plug-and-play" architectural component that bridges the gap between flexible deep learning and rigorous monotonic constraints. Our framework is built upon two core pillars:

- **Differentiable Piecewise Fitting:** We partition the input feature space into discrete segments, each governed by a local fitting weight $w_i$. By enforcing a non-negativity constraint ($w_i \geq 0$) through an activation function (e.g., ReLU or Softplus), we instantiate a global inductive bias that guarantees the output is monotonically non-decreasing.
- **Debiasing as Calibration:** We reframe debiasing as a monotonic transformation problem. By parameterizing segment weights as learnable, context-aware embeddings, the model "learns the distortion" for specific features like rank or task type. This allows the Isotonic Layer to act as a universal debiasing agent, reshaping distorted scores back into calibrated utility scores while preserving the underlying model's ranking integrity.

## 1.4 Contributions

The key innovations of this work are summarized as follows:

(1) **Differentiable Isotonic Layer:** We introduce a deep-learning-native Isotonic Layer that enforces global monotonicity within a neural framework. By conditioning the layer on high-dimensional context features (e.g., advertiser ID, device type) via learnable embeddings, we enable **fine-grained, context-aware calibration** that transcends the limitations of traditional non-parametric methods.
(2) **Efficient Piecewise Architecture:** We implement the layer using a dot-product between a non-negative weight vector and a transformed input activation vector. This formulation leverages optimized BLAS operations, ensuring that the layer is computationally efficient for high-throughput production systems.
(3) **Handling Task Heterogeneity:** To address the inherent heterogeneity of user behaviors, we scale the Isotonic Layer to **Multi-Task Learning (MTL)** environments through the introduction of task-specific isotonic embeddings. This architecture enables the model to adaptively 'stretch' or 'compress' predicted score distributions by learning specialized monotonic profiles for distinct objectives. By doing so, the framework dynamically accounts for the unique distortion intensities associated with different engagement types — such as the high position-sensitivity of click-through rates (CTR) versus the intent-driven nature of conversion rates (CVR) — ensuring seamless and robust calibration across a unified MTL pipeline..
(4) **Dual-Task Formulation:** We decouple recommendation into a latent relevance estimation task and a bias-aware calibration task, using the Isotonic Layer as a differentiable functional bridge and extended this formulation to a generic debiasing framework.

Mathematically, for an input $x$ and fixed knots $k_0, \ldots, k_n$, the layer computes:

$$y = w_0(k_1 - k_0) + w_1(k_2 - k_1) + \cdots + w_i(x - k_i)$$

This piecewise linear approach provides sufficient capacity to prevent underfitting while maintaining the structural properties required for robust, unbiased recommendations in heterogeneous environments.

# 2 Related Work

Model calibration and debiasing are central challenges in recommender systems, where observed signals are systematically distorted by exposure, selection, and presentation effects. Prior work largely falls into three categories: post-hoc calibration, differentiable monotonic modeling, and causal or counterfactual debiasing.

### 2.1 Classical Calibration

Post-hoc calibration methods such as **Platt Scaling** [15] apply a parametric logistic transformation to predicted scores. While efficient, these methods are too restrictive for the complex, non-linear distortions observed in modern recommender systems. **Isotonic Regression** [17] offers a non-parametric monotonic alternative, but relies on non-differentiable projection (e.g., PAVA), making it difficult to integrate into end-to-end training pipelines. In addition, isotonic regression often suffers from overfitting under data sparsity and produces staircase-shaped mappings that degrade ranking resolution.

### 2.2 Differentiable Monotonic Calibration

Recent work has explored embedding monotonic calibration into neural networks. Our previous paper, **LiRank** [2] introduces a differentiable isotonic calibration layer by bucketizing logits and learning non-negative weights via backpropagation. Similarly, **Deep Isotonic Promotion Network (DIPN)** [13] models monotonic incentive–response relationships using isotonic embeddings with additional smoothness regularization.

While effective, these approaches typically apply a *single global monotonic function* and focus on single-task settings. More recent neural monotonic calibrators based on unconstrained monotonic networks [1] further improve expressiveness, but remain limited to score-level calibration and do not address task heterogeneity or context-specific bias.

### 2.3 Debiasing via Causal and Counterfactual Learning

A parallel line of work addresses debiasing through causal inference and counterfactual learning. Classical approaches include **Position-as-Feature (PAF)** modeling and **Inverse Propensity Scoring (IPS)** [10], which reweight training samples to simulate randomized exposure. However, PaF suffers from inference-time bias shift, while IPS-based methods are high-variance and sensitive to propensity estimation errors.

Recent deep learning approaches improve robustness by modeling bias explicitly through representation learning or meta optimization. These include disentangled or variational models that separate preference from bias factors [7], adversarial or propensity-learning frameworks [18], and general debiasing meta-frameworks such as **AutoDebias** [4], which jointly optimize reweighting, imputation, and prediction. While powerful, these methods primarily operate on the *data distribution* or *latent representations* and do not impose explicit structural constraints on the score transformation itself.

Beyond monotonic calibration and counterfactual debiasing, recent work has explored multifactorial bias correction in recommendation, where both popularity and positivity biases are jointly modeled to improve fairness and robustness [8, 14]. Contrastive representation learning has been applied to mitigate dual biases such as popularity and conformity [9], while invariant learning frameworks aim to disentangle invariant user preferences from biased interactions using imputation and knowledge distillation [5]. Additionally, thorough analyses and surveys on popularity bias shed light on the broader impacts of algorithmic bias in recommender systems and their mitigation strategies [3, 11].

### 2.4 Multi-Task Learning and Bias Heterogeneity

Modern industrial recommenders are inherently multi-task, optimizing signals with vastly different bias profiles (e.g., clicks vs. conversions). Prior work in multi-task debiasing focuses on mitigating negative transfer via task decoupling or causal regularization [12]. However, most approaches implicitly assume a shared calibration or bias correction mechanism across tasks, despite large differences in exposure sensitivity.

### 2.5 Summary

In contrast to prior work, our approach treats debiasing as a *differentiable monotonic calibration problem* and embeds it directly into multi-task neural architectures. By learning context- and task-conditioned isotonic embeddings, the proposed Isotonic Layer enables fine-grained, per-objective calibration within a unified multi-task framework—addressing bias heterogeneity that global monotonic or causal reweighting methods do not explicitly handle.

## 3 Isotonic Layer Architecture Overview

We propose an *Isotonic Layer*, a differentiable neural network module that enforces monotonically non-decreasing outputs with respect to a scalar input, illustrated in **Figure 1(b)**. Unlike classical isotonic regression methods that rely on exact projection operators, the proposed layer adopts a bucket-based cumulative construction that is fully compatible with end-to-end gradient-based optimization.

The layer discretizes the input domain into fixed-width buckets and associates each bucket with a learnable non-negative weight. Monotonicity is guaranteed by (i) cumulative activation of buckets as the input increases and (ii) explicit non-negativity constraints on bucket weights enforced via a ReLU parameterization.

### 3.1 Mathematical Formulation

Let $x \in \mathbb{R}$ denote a scalar input. The Isotonic Layer maps $x$ to an output $y \in (0, 1)$ such that for any $x_1 \leq x_2$, the corresponding outputs satisfy $y(x_1) \leq y(x_2)$.

*Input Clipping.* The input is first clipped to a predefined range $[L, U]$:

$$\tilde{x} = \mathrm{clip}(x, L, U), \tag{1}$$

where $L$ and $U$ denote the lower and upper bounds, respectively.

In our implementation, we set the default bounds to $L = -17$, $U = 8$, with a bucket width $\Delta_b = 0.05$. These values are not theoretically unique, but are chosen as practical defaults based on empirical observations in large-scale click-through rate (CTR) and engagement prediction tasks. [2]

[2]For the sigmoid function $\sigma(z) = (1 + e^{-z})^{-1}$, saturation occurs when the derivative $\sigma'(z) = \sigma(z)(1 - \sigma(z))$ becomes negligibly small. At $z = -17$, $\sigma(-17) \approx 4.1 \times 10^{-8}$ and $\sigma'(-17) \approx 4.1 \times 10^{-8}$, while at $z = 8$, $\sigma(8) \approx 0.999665$ and $\sigma'(8) \approx 3.35 \times 10^{-4}$. Beyond this interval, both the output probability and its gradient change by less than $10^{-4}$ per unit change in $z$, implying that further variation in the logit has negligible effect on the output or optimization dynamics. Clipping to this range therefore introduces minimal approximation error while improving numerical stability and calibration robustness.

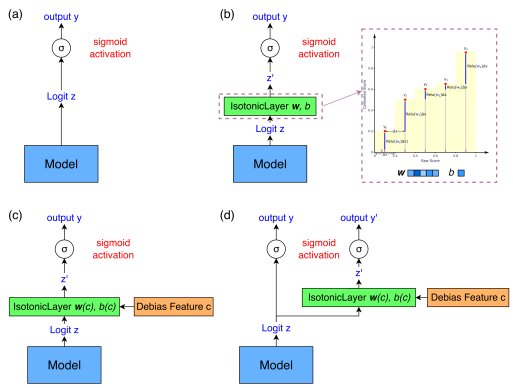


**Figure 1: Example use cases of the Isotonic Layer in ranking and calibration. (a) Standard prediction model without monotonic calibration. (b) Global isotonic calibration applied to model outputs with learnable bucket weights w and bias b. (c) Context-conditioned isotonic calibration via learned embeddings, where a context feature $c$ is mapped to a context-conditioned isotonic embedding $(\mathrm{w}_c, b_c)$. (d) Dual-tower architecture with joint optimization of relevance and context-aware isotonic calibration for debiasing.**

*Bucketization.* The interval $[L, U]$ is discretized into $N$ buckets of width $\Delta_b$:

$$N = \left\lceil \frac{U - L}{\Delta_b} \right\rceil + 1. \tag{2}$$

The clipped input $\tilde{x}$ is mapped to a bucket index $i \in \{0, 1, \dots, N-1\}$ via

$$i = \left\lfloor \frac{\tilde{x} - L + \Delta_b}{\Delta_b} \right\rfloor, \tag{3}$$

with $i$ clamped to the valid range $[0, N-1]$.

*Activation Vector Construction.* For each input $\tilde{x}$, an activation vector $\mathbf{a}(\tilde{x}) \in \mathbb{R}^N$ is constructed to represent the accumulated contribution of buckets:

$$a_j(\tilde{x}) = \begin{cases} \Delta_b, & j < i, \\ (\tilde{x} - L + \Delta_b) - i \cdot \Delta_b, & j = i, \\ 0, & j > i. \end{cases} \tag{4}$$

The fractional term captures the partial activation within the current bucket.

*Weighted Aggregation.* Each isotonic unit maintains a learnable weight vector $\mathbf{w} \in \mathbb{R}^N$ and bias $b \in \mathbb{R}$. To enforce monotonicity, weights are parameterized as

$$\mathbf{w}^+ = \mathrm{ReLU}(\mathbf{w}), \tag{5}$$

ensuring $\mathbf{w}^+ \geq 0$ element-wise. The pre-activation output is computed as

$$z(x) = \sum_{j=0}^{N-1} a_j(\tilde{x})\, w_j^+ + r + b, \tag{6}$$

where $r = L - \Delta_b$ is a constant residue offset for boundary alignment.

Here we parameterize bucket weights using ReLU activations, which serves a dual purpose: (1) it strictly enforces the non-negativity constraint required for monotonicity, and (2) it ensures seamless integration with modern deep learning frameworks due to its computational efficiency and numerical stability. Because the output is computed as a linear function of $\mathbf{a}(\tilde{x})$, the entire layer remains fully differentiable almost everywhere and can be trained end-to-end via standard backpropagation.

*Output Activation.* The final output is obtained by applying a sigmoid transformation:

$$y(x) = \sigma(z(x)) = \frac{1}{1 + e^{-z(x)}}. \tag{7}$$

*Loss Function and Training.* The Isotonic Layer is trained end-to-end using standard task-specific loss functions applied to the sigmoid output $y(x)$. Given a labeled dataset $\{(x_i, t_i)\}_{i=1}^M$, where $t_i \in \{0, 1\}$ denotes the target label (e.g., click or engagement), we minimize a conventional probabilistic loss such as Binary Cross Entropy (BCE):

$$\mathcal{L}_{\mathrm{BCE}} = -\frac{1}{M} \sum_{i=1}^{M} \left[ t_i \log y(x_i) + (1 - t_i) \log(1 - y(x_i)) \right]. \tag{8}$$

In practice, the Isotonic Layer can be trained on the upstream model's dataset or a separate holdout set. While joint training optimizes relevance and calibration simultaneously, using a distinct dataset decouples structural bias from relevance learning. This separation is particularly effective for correcting systematic distortions—such as position or presentation bias—rather than merely improving raw accuracy. Consistent with prior work in post-hoc calibration and counterfactual learning, isolating calibration from estimation enhances robustness and generalization under distribution shift.

## 3.2 Monotonicity Guarantee

For any two inputs $x_1 \leq x_2$, their corresponding activation vectors satisfy

$$\mathbf{a}(x_1) \leq \mathbf{a}(x_2) \quad \text{(element-wise)}. \tag{9}$$

Since all bucket weights are non-negative ($w_j^+ \geq 0$), the pre-activation output is monotonic:

$$z(x_1) \leq z(x_2). \tag{10}$$

Because the sigmoid function is strictly monotonic, the final output preserves this ordering:

$$y(x_1) \leq y(x_2). \tag{11}$$

Thus, monotonic non-decreasing behavior is guaranteed by construction.

## 3.3 Expressiveness and Empirical Illustration

To evaluate the expressive capacity of the Isotonic Layer, we first conduct a synthetic experiment using the monotonic target function $y = x^2$ over $x \in [0, 1]$. To mimic the score distribution commonly observed in recommendation systems, the input is transformed into logit space via $\mathrm{logit}(x) = \ln \frac{x}{1-x}$ prior to being passed into the layer. The model is optimized using binary cross-entropy loss. As shown in **Figure 2**, the Isotonic Layer achieves a near-perfect approximation of the quadratic target, demonstrating that the proposed piecewise linear construction retains strong expressive power despite monotonic constraints.

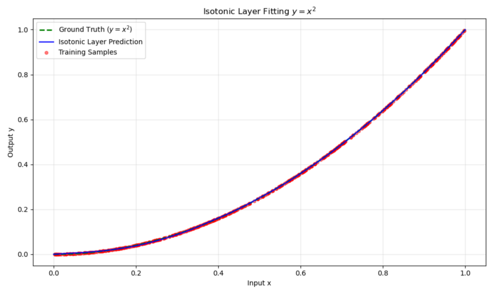


**Figure 2: Expressive capacity of the Isotonic Layer on a monotonic synthetic task ($y = x^2$).**

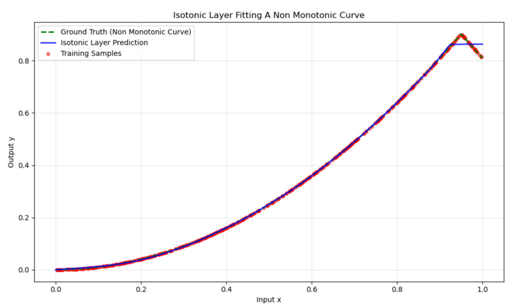


**Figure 3: Robustness to non-monotonic noise. The Isotonic Layer enforces global monotonic structure while smoothing local distortions in the target signal.**

We further evaluate robustness under non-monotonic distortions by constructing a piecewise target function defined as $y = x^2$ for $x \leq 0.95$ and $y = (1.9 - x)^2$ for $x > 0.95$. This setup simulates overfitting artifacts that may arise in real-world models due to noise or data sparsity. As illustrated in **Figure 3**, the Isotonic Layer effectively suppresses local inversions while recovering the optimal monotonic approximation, highlighting its ability to provide structural regularization under noisy or inconsistent inputs.

## 3.4 Global Structure vs. Local Parametrization

Unlike standard multi-layer perceptrons, whose parameters operate locally and independently, the Isotonic Layer imposes a global structural constraint. Because the activation at input $x$ accumulates contributions from all preceding buckets, the output depends jointly on the entire prefix of the weight embedding. This induces strong spatial coupling across segments.

As a result, the model cannot arbitrarily modify the function shape in isolated regions, even under data sparsity. This global dependency acts as an implicit regularizer, preventing local overfitting or monotonicity violations while retaining high expressive capacity through fine-grained bucketization. Empirically, this property allows the Isotonic Layer to generalize reliably in regions with limited observations, a common challenge in large-scale recommendation systems.

## 3.5 Context-Conditioned Isotonic Embeddings

A key advantage of the **Isotonic Layer** is that its bucket weights function as a **learnable embedding** parameterizing a monotonic transformation. This allows for context-aware calibration, where the isotonic function adapts dynamically based on auxiliary features like display position or device type (see **Figure 1(c)**). For a context variable $c$, we define a context-conditioned weight embedding:

$$\mathbf{w}(c) = \mathbf{E}(c) \in \mathbb{R}^N, \tag{12}$$

where $\mathbf{E}(\cdot)$ is implemented via an embedding lookup or a lightweight neural projection. Each context induces a distinct isotonic calibration curve while preserving global monotonicity through the non-negativity constraint:

$$\mathbf{w}^+(c) = \mathrm{ReLU}(\mathbf{E}(c)). \tag{13}$$

The calibrated output is then computed as

$$y = \mathrm{IsotonicLayer}(x\,;\, \mathbf{w}^+(c), b(c)), \tag{14}$$

where both the bucket weights and bias may be conditioned on context. (code implementation see appendix A)

*Example: Position Bias.* In ranking systems, display position $p$ often introduces observational bias. By associating each position with a learnable isotonic embedding $\mathbf{E}_p$, the model captures position-specific monotonic distortions of the latent relevance score. This mechanism enables the calibration layer to dynamically "stretch" or "compress" score distributions, effectively modeling position bias as a learned monotonic transformation:

$$\hat{y} = f_{iso}(r; \mathbf{E}_p) \tag{15}$$

where $r$ is the raw relevance score and $f_{iso}$ is the position-conditioned isotonic function.

## 3.6 Dual-Tower Debiasing Architecture

We operationalize bias-aware learning by decomposing the model into two complementary components connected via the Isotonic Layer, as illustrated in **Figure 1(d)**:

- **Relevance Tower.** A standard neural network that estimates the latent utility $r$ of an item given user and item features, independent of exposure bias.
- **Isotonic Calibration Layer.** A context-conditioned monotonic mapping that transforms latent utility into the observed interaction space, capturing systematic distortions such as position bias or interface effects.

Formally, the observed prediction is given by

$$\hat{y} = f_{iso}(r\,;\, c), \tag{16}$$

where the context $c$ controls the isotonic embedding. This decoupling allows the model to learn bias patterns explicitly while preserving a clean separation between relevance estimation and calibration.

*Bias Neutralization at Inference.* A key practical advantage of this formulation is that bias can be neutralized at inference time. By fixing the context embedding to a reference value (e.g., a canonical display position) or by bypassing the calibration layer entirely, the system can produce rankings driven purely by latent utility:

$$\hat{r} \approx r. \tag{17}$$

This enables counterfactual evaluation, fair ranking, and policy simulation without retraining the relevance model, making the approach particularly attractive for large-scale production systems.

# 4 Isotonic Layer as a Calibration Framework

In many real-world applications, models suffer from significant overfitting and severe score shifts when training data is sparse. We encounter these challenges specifically in downstream modeling tasks, where labels represent deep conversions—such as offsite purchases in advertising or long-term retention—that are statistically sparse yet high in business value. In this section, we evaluate the Isotonic Layer as a standalone calibration framework for such tasks.

Our specific use case involves estimating the conditional probability of downstream session, defined as the probability that a member returns to the platform via a notification after an initial feed interaction (e.g., like, comment, or share):

$$P(\text{downstream session} \mid \text{user action})$$

This prediction is critical for optimizing notification triggering and ranking policies in large-scale recommender systems. By treating this task as a calibration problem, we leverage the monotonic constraints of the Isotonic Layer to stabilize predictions against the volatility inherent in sparse label distributions, thereby mitigating the score shifts common in traditional deep learning approaches.

## 4.1 Motivation

The downstream engagement model is trained on relatively sparse and highly conditional data, as downstream sessions occur significantly less frequently than initial engagement events. As a result, the model is prone to overfitting and exhibits high variance in predicted probabilities, particularly in the tail regions of the score distribution.

In production, this instability manifests as undesirable score swings across model iterations and sensitivity to data shifts.

## 4.2 Calibration via Isotonic Layer

To address these challenges, we utilize the proposed Isotonic Layer as a robust post-hoc calibration module. We implement an **alternating training procedure** that explicitly decouples representation learning from probability calibration.

The training process consists of two distinct, serialized phases:

- **Representation Learning:** The primary downstream model is trained on engagement data to generate raw probability estimates $\hat{y}_{raw}$.
- **Calibration Optimization:** The base model parameters are frozen, and the Isotonic Layer is optimized over these static outputs using randomized session data to learn the mapping $f : \hat{y}_{raw} \to \hat{y}_{calib}$.

By enforcing a monotonic constraint on the raw model outputs, the Isotonic Layer ensures that final probabilities remain stable and consistent. This architectural decoupling effectively mitigates the volatility and score shifts prevalent in data-sparse regimes, providing a more reliable signal for downstream decision-making. To account for varying behavior across different contexts, we employ independent Isotonic Layers for distinct tasks and user segments, such as mobile versus desktop platforms.

## 4.3 Experiment Results

We evaluate the **Isotonic Layer** within a production downstream model using product data. The baseline is a **multi-task learning (MTL)** model that optimizes several binary objectives via shared MLP layers and task-specific towers.

*Offline Ranking Performance.* As shown in **Table 1**, the treatment model achieves substantial **Evaluation(Eval) AUC** gains (**+1.5%** and **+1.9%** for Downstream Share and Downstream Comment, respectively). These improvements demonstrate that the monotonic constraints of the Isotonic Layer effectively mitigate overfitting in data-sparse regimes.

**Table 1: Relative Performance Improvement of Evaluation AUC**

| Model | Share Eval AUC | Like Eval AUC | Comment Eval AUC |
|---|---|---|---|
| Baseline | - | - | - |
| Treatment | **+1.5%** | +0.0% | **+1.9%** |

*Prediction Stability and Calibration.* Online monitoring highlights the Isotonic Layer's capability in stabilizing model predictions. As illustrated in **Figure 4**, the treatment model exhibits significantly lower variance in daily average scores compared to the baseline. Furthermore, the post-calibration scores are notably lower, aligning more closely with the empirical distribution of the training labels and correcting the inherent overestimation typical of uncalibrated MTL models.


**Figure 4: Comparison of Online Prediction Scores. It monitors model average output scores in a daily basis, where X axis represents date and Y axis represents the probabilities.**

**Table 2: Online A/B Test Results: Relative Lift over Baseline**

| Metric | Relative Lift (Δ %) |
|---|---|
| Daily Active User Interaction | **+0.17%** |
| User Seeking knowledge or Suggestion | **+0.20%** |
| Another Topline Metrics (sensitive) | **+0.36%** |

*Online A/B Testing.* Live traffic experiments yield statistically significant lifts in core ecosystem metrics as reflected in **Table 2**. These results confirm the practical utility of the Isotonic Layer as a robust calibration framework for real-world industrial applications.

## 5 Isotonic Layer Debias Architecture and Experiment

We implement the Isotonic Layer within a multi-task dual-tower framework. This design allows for high-fidelity debiasing during training while maintaining low-latency inference.

### 5.1 Dual-Tower Architecture

The model utilizes a shared latent representation layer followed by task-specific prediction heads. As shown in **Figure 5**, for each engagement signal $s \in \{\text{Click}, \text{Skip}, \text{LongDwell}\}$, we deploy two parallel towers:

- **Inference Head:** Predicts a position-neutral relevance score $\hat{y}^{(s)}_{\text{inf}}$. This head is used exclusively during online serving.
- **Isotonic Calibration Head:** Learns a bias-aware probability $\hat{y}^{(s)}_{\text{iso}}$ by applying the Isotonic Layer to the shared representation, conditioned on display position and platform features.

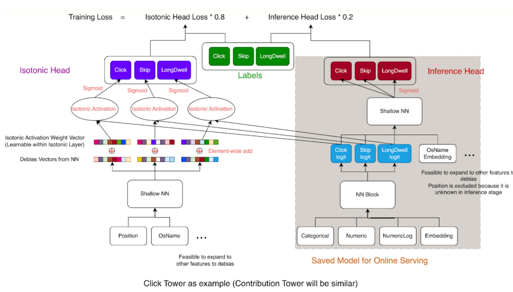


**Figure 5: Dual-head architecture for isotonic position debiasing. Each task group consists of a position-neutral inference head and a context-conditioned isotonic head. The isotonic head utilizes position and platform features during training to decouple exposure bias from intrinsic relevance. Loss weights for each head are optimized as hyperparameters to balance ranking performance and model stability.**

### 5.2 Joint Optimization

The system is optimized using a weighted Binary Cross-Entropy (BCE) loss:

$$\mathcal{L} = \sum_s \left( \alpha_s \text{BCE}(y^{(s)}, \hat{y}^{(s)}_{\text{inf}}) + \beta_s \text{BCE}(y^{(s)}, \hat{y}^{(s)}_{\text{iso}}) \right)$$

Through empirical tuning, we found that a relative weight of $0.7 \leq \beta_s \leq 0.8$ for the isotonic head yields the best debiasing results while ensuring the inference head remains well-calibrated for downstream value functions.

### 5.3 Offline and Online Performance

We evaluate the inference head's ability to isolate $P(\text{relevance})$ by benchmarking against a production baseline. **Table 3** summarizes the offline evaluation gains.

**Table 3: Offline Performance Comparison: Isotonic Model vs. Baseline in Like and Long Dwell ( dwell time > a threshold) Tasks**

| Task | Δ AUC (%) | Δ NE (%) |
|---|---|---|
| Baseline | - | - |
| Isotonic Head's Like Task | **0.81%** | **-0.51%** |
| Isotonic Head's Long Dwell Task | **1.02%** | **-0.73%** |
| Inference Head's Like Task | **0.10%** | 0.21% |
| Inference Head's Long Dwell Task | **0.11%** | 0.28% |

### 5.4 Online A/B Testing Results

The online results represent the impact of the positional debiasing model on live traffic metrics.

**Table 4: Online A/B Testing Results (Production Environment)**

| Metric | Δ Relative to Control |
|---|---|
| Subscription Weekly Active User | **+0.63%** |
| Daily Unique Professional Interactions | **+0.14%** |
| Job Session | **+0.14%** |
| Total Macrosessions | **+0.06%** |

### 5.5 Analysis and Key Takeaways

Experimental results demonstrate that the **Isotonic Calibration Layer** effectively decouples relevance from position bias, yielding consistent ranking improvements across diverse engagement signals.

- **Effective Bias Decomposition:** The model isolates $P(\text{relevance})$ from $P(\text{event} \mid \text{relevance}, \text{position})$, resulting in significant **AUC** gains. While we observe a slight regression in **Normalized Entropy (NE)**, this is a systemic consequence of removing extrinsic positional signals that baseline models typically overfit to, confirming that our approach captures a truer relevance signal.
- **Heterogeneity and Directionality of Bias:** Bias is non-uniform and task-specific; for instance, *Long Dwell* exhibit severe positive bias at top ranks, while *Skip* signals exhibit a reciprocal trend. We also observe a systematic divergence in the **Observed-to-Expected (O/E) ratio** ($O/E > 1$ at top positions; $O/E < 1$ at lower ranks), proving that a static propensity score cannot resolve the non-linearities captured by our isotonic approach.
- **Production Scalability:** To mitigate a **5% serving CPU overhead**, we employ a hybrid architecture: using the Isotonic Layer for robust debiasing during training and a lightweight shallow neural network for low-latency inference. This maintains ranking gains while satisfying production throughput requirements.
- **Loss Weighting Optimization:** The relative loss weights $(\alpha_s, \beta_s)$ are critical for system stability. Empirically, a calibration weight $\beta_s \in [0.7, 0.8]$ yields optimal online performance. Retaining a non-zero inference weight $\alpha_s$ is essential

to ensure predicted scores remain well-calibrated, preventing volatility in the downstream value function and maintaining ranking consistency.

## 6 Future Work

We identify several promising directions to extend the Isotonic Layer for more complex recommendation environments:

- **Multi-Dimensional and Partial Isotonicity:** We aim to extend the framework to multivariate monotonic constraints, ensuring outputs are simultaneously monotonic with respect to multiple dimensions such as item quality, integrity, and recency.
- **Context-Aware Granular Calibration:** While global calibration is common, achieving accuracy at the sub-segment level (e.g., individual advertisers or campaigns) remains difficult. By parameterizing the Isotonic Layer with high-dimensional context embeddings, we can learn millions of specialized calibration curves within a single unified model. Preliminary results show this approach can replace dozens of fragmented sub-models while improving both infrastructure simplicity and business metrics.
- **Adaptive Knot Optimization:** We plan to investigate mechanisms to dynamically adjust segment boundaries (knots) based on empirical activation distributions. By increasing knot density in high-probability feature regions, the Isotonic Layer acts as a **learnable, monotonic activation function**. This structural inductive bias can be integrated into the hidden layers of the ranking tower to guarantee monotonicity throughout the network.
- **Position Action Prediction (PAP):** To bridge the gap between "position-neutral" training and real-world inference where the rank is unknown, we propose the **PAP framework** (**Figure 6**). This involves a secondary task to predict the position distribution:

$$P(\text{pos} = i \mid \text{relevance})$$

  The model uses this distribution to perform soft-attention pooling over position embeddings, generating an "expected bias" representation. This allows for calibrated predictions that align more closely with observed events probabilities than relevance-only models.
- **Enhanced Off-Policy Evaluation (OPE):** Our framework offers a mechanism to predict counterfactual rewards by modeling the interaction between latent utility and positional bias. This enables the generation of high-fidelity, context-aware propensity scores, providing a robust alternative to traditional Inverse Propensity Scoring (IPS) for offline model validation.

## 7 Conclusion

In this paper, we presented the **Isotonic Layer**, a novel, differentiable framework designed to enforce global monotonicity within deep learning architectures. By reformulating traditional isotonic regression into a constrained dot-product of piecewise linear segments, we have bridged the gap between flexible neural representations and the structural necessity of monotonic calibration.

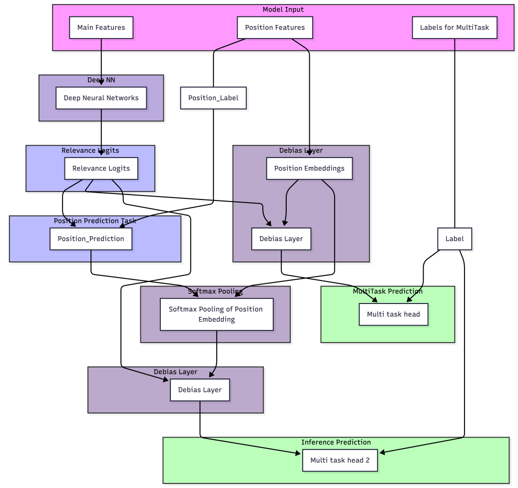


**Figure 6: The Position Action Prediction (PAP) Framework. To bridge the inference-time exposure gap, a secondary Position Estimation Task predicts a probabilistic distribution over potential display slots based on latent relevance. This distribution enables stochastic pooling of positional embeddings to synthesize an "expected bias" representation. By combining this bias with the relevance score, the model produces a calibrated observation probability, yielding higher ranking accuracy than relevance-only baselines.**

Our framework offers several key advantages over traditional post-hoc calibration methods. First, its **differentiable nature** allows for seamless end-to-end training within complex recommendation towers, enabling the model to learn optimal score mappings directly from raw data. Second, the introduction of **context-aware embeddings** empowers the architecture to perform highly granular, customized calibration—effectively solving the "long-tail" calibration problem for specific sub-segments such as individual advertisers or device types. This replaces the need for fragmented, high-maintenance infrastructures consisting of dozens of localized sub-models with a single, unified, and scalable architecture.

The **dual-task formulation** introduced here provides a robust mechanism for decoupling latent user utility from systemic confounding biases. Our extensive experimental results on real-world datasets, supplemented by successful large-scale production A/B tests, demonstrate that the Isotonic Layer significantly enhances calibration fidelity and ranking consistency without compromising model expressivity. In production, this approach has proven its ability to dramatically simplify system complexity while simultaneously improving top-line business metrics.

As industrial recommendation systems move toward greater transparency and fairness, the need for architectural inductive biases that respect domain priors is paramount. The Isotonic Layer serves as a scalable, robust, and model-agnostic foundation for this next generation of unbiased and well-calibrated machine learning systems.

## References


[1] Yimeng Bai, Shunyu Zhang, Yang Zhang, Hu Liu, Wentian Bao, Enyun Yu, Fuli Feng, and Wenwu Ou. 2025. Unconstrained Monotonic Calibration of Predictions in Deep Ranking Systems. In *Proceedings of the 48th International ACM SIGIR Conference on Research and Development in Information Retrieval* (Padua, Italy) *(SIGIR '25)*. Association for Computing Machinery, New York, NY, USA, 937–947. doi:10.1145/3726302.3730105

[2] Fedor Borisyuk and Others. 2024. LiRank: Industrial Large Scale Ranking Models at LinkedIn. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining* (Barcelona, Spain) *(KDD '24)*. Association for Computing Machinery, New York, NY, USA, 4804–4815. doi:10.1145/3637528.3671561

[3] Filippo Carnovalini, Antonio Rodà, and Geraint A. Wiggins. 2025. Popularity Bias in Recommender Systems: The Search for Fairness in the Long Tail. *Information* 16, 2 (2025), 151. doi:10.3390/info16020151 Narrative review of popularity bias impacts and mitigation in RS, emphasizing fairness concerns.

[4] Jiawei Chen, Hande Dong, Yang Qiu, Xiangnan He, Xin Xin, Liang Chen, Guli Lin, and Keping Yang. 2021. AutoDebias: Learning to Debias for Recommendation. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '21)*. ACM, 21–30. doi:10.1145/3404835.3462919

[5] Savvina Daniil, Manel Slokom, Mirjam C. Cuper, Cynthia C. S. Liem, Jacco van Ossenbruggen, and Laura Hollink. 2025. Invariant Debiasing Learning for Recommendation via Biased Imputation. *Information Processing & Management* 62 (2025), 104028. doi:10.1016/j.ipm.2024.104028 Applies invariant learning and imputation to improve unbiased preference modeling.

[6] de Leeuw, K J Hornik, and P Mair. 2009. Isotone optimization in R: Pool-adjacent-violators (PAVA) and active set methods. *Journal of Statistical Software* 32(5) (2009), 1–24.

[7] Yupu Guo, Fei Cai, Xin Zhang, Jianming Zheng, and Honghui Chen. 2023. Disentangled Variational Auto-encoder Enhanced by Counterfactual Data for Debiasing Recommendation. *arXiv preprint* (2023). https://arxiv.org/abs/2306.15961 DB-VAE disentangles bias types and leverages counterfactual data.

[8] Jin Huang, Harrie Oosterhuis, Masoud Mansoury, Herke van Hoof, and Maarten de Rijke. 2024. Going Beyond Popularity and Positivity Bias: Correcting for Multifactorial Bias in Recommender Systems. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*. 416–426. doi:10.1145/3626772.3657749 Extends debiasing beyond single-factor bias by modeling both popularity and positivity in RS.

[9] Zhirong Huang, Shichao Zhang, Debo Cheng, Jiuyong Li, Lin Liu, and Guixian Zhang. 2024. Debiased Contrastive Representation Learning for Mitigating Dual Biases in Recommender Systems. *arXiv preprint* (2024). https://arxiv.org/abs/2408.09646 Employs contrastive learning to jointly mitigate popularity and conformity biases in RS training.

[10] Thorsten Joachims and Others. 2017. Unbiased Learning-to-Rank with Biased Feedback. In *Proceedings of the Tenth ACM International Conference on Web Search and Data Mining*. ACM, 781–789.

[11] Anastasiia Klimashevskaia, Dietmar Jannach, Mehdi Elahi, and Christoph Trattner. 2024. Addressing Popularity Bias in Recommender Systems: Survey, Metrics and Mitigation. *arXiv preprint* (2024). https://arxiv.org/abs/2308.01118 Comprehensive survey on popularity bias definitions, metrics, and mitigation strategies.

[12] Huishi Luo and Others. 2025. ORCA: Mitigating Over-Reliance for Multi-Task Dwell Time Prediction with Causal Decoupling. In *Proceedings of the 34th ACM International Conference on Information and Knowledge Management* (Seoul, Republic of Korea) *(CIKM '25)*. Association for Computing Machinery, New York, NY, USA, 4996–5000. doi:10.1145/3746252.3760898

[13] Yiming Ma and Others. 2020. Deep Isotonic Promotion Network. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. Virtual Event. Models monotonic incentive-response curves with isotonic embeddings.

[14] Masoud Mansoury, Jin Huang, Mykola Pechenizkiy, Herke van Hoof, and Maarten de Rijke. 2026. The Unfairness of Multifactorial Bias in Recommendation. *arXiv preprint* (2026). https://arxiv.org/abs/2601.12828 Analyzes combined effects of popularity and positivity bias on exposure fairness.

[15] John Platt. 1999. Probabilistic Outputs for Support Vector Machines and Comparisons to Regularized Likelihood Methods. In *Advances in Large Margin Classifiers*. MIT Press, 61–74. Classic description of Platt scaling, widely used for calibration in ML.

[16] Yitao Shen, Yue Wang, Xingyu Lu, Feng Qi, Jia Yan, Yixiang Mu, Yao Yang, YiFan Peng, and Jinjie Gu. 2021. A framework for massive scale personalized promotion. arXiv:2108.12100 [cs.LG] https://arxiv.org/abs/2108.12100

[17] Bianca Zadrozny and Charles Elkan. 2002. Transforming Classifier Scores into Accurate Multiclass Probability Estimates. In *Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 694–699. doi:10.1145/775047.775151

[18] Kuiyu Zhu, Tao Qin, Pinghui Wang, and Xin Wang. 2025. Adversarial propensity weighting for debiasing in collaborative filtering. In *Proceedings of the Thirty-Fourth International Joint Conference on Artificial Intelligence* (Montreal, Canada) *(IJCAI '25)*. Article 412, 9 pages. doi:10.24963/ijcai.2025/412


## A Implementation Details

The Isotonic Layer is designed to be a "plug-and-play" component. By enforcing non-negativity on weights, we ensure a monotonic relationship through a piecewise linear fitting. Below, we provide a PyTorch implementation that leverages optimized tensor operations for efficiency.

```
import torch
import torch.nn as nn
import torch.nn.functional as F

class IsotonicLayer(nn.Module):
    """
    Isotonic Layer implementation in PyTorch.
    Ensures monotonicity via piecewise linear
    fitting with non-negative weights.
    """
    def __init__(
        self,
        units=12, # unit is the task number
        lb=-17.0, # lower bound for logits
        ub=8.0, # upper bound for logits
        step=0.2,
        w_init_factor=0.1
    ):
        super(IsotonicLayer, self).__init__()
        self.units = units
        self.lb = lb
        self.ub = ub
        self.step = step
        self.num_buckets = int((ub - lb) / step) + 1
        self.residue = lb - step

        # Learnable parameters
        self.v = nn.Parameter(
            torch.ones(units, self.num_buckets)
            * w_init_factor
        )
        self.b = nn.Parameter(torch.zeros(units))

    def forward(self, x, calibration_embedding=None):
        # x shape: [batch_size, units] or [batch_size, 1]
        if x.dim() == 1:
            x = x.unsqueeze(1).expand(-1, self.units)

        batch_size = x.shape[0]
        device = x.device

        # 1. Clip and normalize inputs
        x_clipped = torch.clamp(
            x, self.lb + 1e-9, self.ub - 1e-9
        )

        # 2. Calculate indices for piecewise segments
        # index = floor((x - lb + step) / step)
        indx = (
            (x_clipped - self.lb + self.step)
            / self.step
        ).long()
        indx = torch.clamp(indx, 0, self.num_buckets - 1)
```

```
    # 3. Construct activation vector (Step-based)
    range_vec = torch.arange(
        self.num_buckets, device=device
    ).view(1, 1, -1)
    # Expand index for broadcasting: [batch_size, units, 1]
    expand_indx = indx.unsqueeze(2)

    # Fully activated buckets (where range < index)
    # get the full step value
    activation_vector = torch.where(
        range_vec < expand_indx,
        torch.tensor(self.step, device=device),
        torch.tensor(0.0, device=device)
    )

    # 4. Handle the residue (delta) in the current bucket
    delta = (
        x_clipped - self.lb + self.step)
        - (indx.float() * self.step
    )

    # Use scatter to add the delta at the specific index
    # final_activation shape: [batch_size, units, num_buckets]
    final_activation = activation_vector.clone()
    final_activation.scatter_(2, expand_indx, delta.unsqueeze(2))

    # 5. Enforce non-negativity (Isotonic Constraint)
    # w = ReLU(v) ensures monotonic non-decreasing output
    weights = F.relu(self.v)
    if calibration_embedding is not None:
        weights = F.relu(self.v + calibration_embedding)

    # 6. Dot product fitting:
    # y = sum(activation * w) + offset + bias
    logits = torch.sum(
        final_activation * weights, dim=2
    ) + self.residue + self.b

    return torch.sigmoid(logits)
```

Received xx mm 2026; revised dd mm 2026; accepted dd mm 2026