Ads Allocation in Feed via Constrained Optimization

Jinyun Yan, Zhiyuan Xu, Birjodh Tiwana, Shaunak Chatterjee
LinkedIn Corporation
{jiyan,zhiyxu,btiwana,shchatterjee}@linkedin.com

ABSTRACT

Social networks and content publishing platforms have newsfeed applications, which show both organic content to drive engagement, and ads to drive revenue. This paper focuses on the problem of ads allocation in a newsfeed to achieve an optimal balance of revenue and engagement. To the best of our knowledge, we are the first to report practical solutions to this business-critical and popular problem in industry.

The paper describes how large-scale recommender system like feed ranking works, and why it is useful to consider ads allocation as a post-operation once the ranking of organic items and (separately) the ranking of ads are done. A set of computationally lightweight algorithms are proposed based on various sets of assumptions in the context of ads on the LinkedIn newsfeed. Through both offline simulation and online A/B tests, benefits of the proposed solutions are demonstrated. The best performing algorithm is currently *fully deployed* on the LinkedIn newsfeed and is serving all live traffic.

CCS CONCEPTS

• **Information systems** *Computational advertising*; *Online advertising*; *Social advertising*; *Rank aggregation*.

KEYWORDS

Social Networks, Computational advertising, Constrained optimization, User feedback modeling

ACM Reference Format:

Jinyun Yan, Zhiyuan Xu, Birjodh Tiwana, Shaunak Chatterjee. 2020. Ads Allocation in Feed via Constrained Optimization. In *26th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD '20), August 23–27, 2020, Virtual Event, USA*. ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3394486.3403391

1 INTRODUCTION

The newsfeed, also referred to simply as feed, is a popular product on many social network platforms. Facebook, Instagram, Pinterest, Twitter, Tiktok, and LinkedIn are some examples with a central feed product, where hundreds of millions of users consume content on a daily basis. Users visit feed for relevant content, which includes friends' and followees' updates, company news, group activities and job postings. We refer to such content, where creators do not pay to have their content shown to users, as organic content. Users engage with such content because of their own interest in the content or in the creator or both. Showing relevant organic content helps retain users and grow their long term engagement on the platform.

Monetization is another key consideration for most social network platforms and the common mechanism is to insert ads (which are "sponsored" updates in contrast to the aforementioned organic updates), in the feed. These ads have a native feel and often blend well with the surrounding organic updates. Such an ad product helps advertisers reach their target audience (thereby expanding and eventually earning profit), while simultaneously enabling the platform to build a viable business with the ads revenue. Search ads and display ads (often filled via demand side platforms (DSP) or ad networks) have a similar underpinning. There is very limited reported work on identifying the optimal number of search ads to insert at the top of the page [23].

To the best of our knowledge, there has been no reported work on **how to determine optimal positions for ads in the feed**, and **the effectiveness of various approaches in a real-world large-scale application**. This problem has more complexity than the search ads problem since ads can be inserted at any position. Given the popularity of social networks, and the importance of monetization via ads on feed, this is a very critical problem and is the focus of our current work. The methods we propose generalize to merging two (or more) content streams in a feed-like application, but certain artifacts are more likely to be observed in ads (e.g., the "gap effect" as explained in Section 4).

Organic items are the main driver for engagement, quantified by various affirmative user actions. The ranking objective for organic content is maximizing engagement. Ads, on the other hand, are ranked to maximize expected revenue and typically involve an auction, which determines pricing based on the order of ads. The importance of both these utilities and the vastly varying factors at play for each, result in the following two outcomes in most large-scale feed ranking systems: 1. Individual, separate systems determine ranking of organic items and ads (often with tons of custom information). 2. The blending layer, which comes after the individual ranking is computed, is required to respect the original ranking among items scored by the same system.

These two objectives are often in conflict on the feed because there are limited impression slots. This would not happen if ads were more engaging than organic items, but that is rarely the case. The problem we address is how to allocate impressions to ads in feed efficiently to balance engagement and revenue. One solutions is to simply use fixed slots. For instance, in search ads and display ads on publishers, as well as in some feed applications, ads are often allocated to pre-determined slots. However, such a solution can be quite suboptimal for feed as shown in the example in Figure 1.

One key requirement in blending two sets of results when they are optimizing for different utilities is a conversion factor among the utilities. In some cases, these could be specified by the business

KDD '20, August 23–27, 2020, Virtual Event, USA

ACM ISBN 978-1-4503-7998-4/20/08…$15.00
https://doi.org/10.1145/3394486.3403391

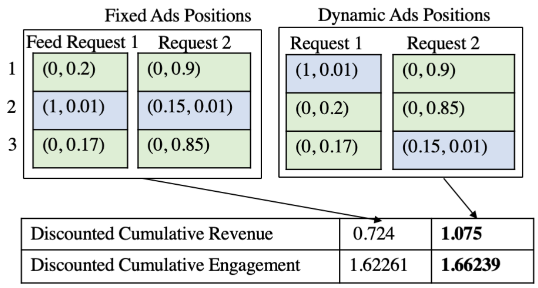

Figure 1: Fixed slotting can be very suboptimal. The 2-tuple represents the revenue and engagement utility for each item. Blue and green items are ads and organic items respectively. The dynamic allocation is simply summing up the two utilities to rank. The total utility reported has a positional discount akin to Discounted Cumulative Gain (DCG), defined in Section 5.

requirements directly (e.g., 1 organic update click is equal to $1.00). We also consider the alternative approach of using constrained optimization which does not directly specify the conversion factor, but instead requires specifying the amount of desired utility for each type (in a primal formulation). The conversion factors are then obtained as the optimal dual variables (as discussed in Section 4).

We first study the simplest formulation assuming all slots on each newsfeed are independent and identically distributed (i.i.d.). We then consider a few relaxations which are also observed in practice when we look at the ads blending problem in newsfeed. For these relaxations, we provide modified algorithms with slightly increased computational complexity and guaranteed optimality (under fewer assumptions). While the algorithms proposed are (by design) simple, they are quite effective in a real-world production system. We believe that practitioners, who are considering moving away from fixed slotting in their various blended content type applications, will find this both insightful and reassuring. For others who are already working on such solutions in large-scale systems, this may serve as a useful reference to compare notes and also find inspiration for an improvement or two.

The primary contributions of our work are three-fold:

(1) We present a detailed overview of various considerations in feed ranking, pertaining to organic item and ads ranking and the advantages of having a second-phase, lightweight blending layer to determine ads' positions. This layer has some desired properties (with due reasoning provided):
   - Preserve prior ranking of organic items and ads
   - Low latency
   - Facilitate asynchronous model updates for any content stream ranking system
   - Handle ads seasonality and prevent ads blindness

(2) We propose a set of lightweight algorithms for allocating ads' positions in the feed.
   - We use a constrained optimization formulation to show optimality under some strong assumptions.
   - A few modifications of the simplest formulation are presented, where some of the assumptions are relaxed, and optimal strategies are derived for each of them.

(3) We report significant improvements in metrics on the LinkedIn feed [5] from using some of the proposed blending algorithms. This system and the reported improvements have been *fully deployed* on the LinkedIn feed.

The rest of the paper is structured as follows. Section 2 reviews the existing work in related problem spaces. In Section 3, we present a detailed overview of a feed ranking system and establish the desired properties of a blending solution. Section 4 proposes a set of algorithms which meet these requirements under various assumptions, many of which are true in the ads position allocation context as shown through offline simulations and online A/B tests in Section 5. Finally, Section 6 summarizes the paper and identifies some future directions.

2 RELATED WORK

The ads allocation problem in search result pages [23] (also in publishers' websites and apps) is a less complex problem. For search ads, they are placed in fixed positions and in separate regions (i.e., on top of the search result page). This can be a near-optimal allocation because users expect most relevant search results appearing on the top rather than any other positions, hence the top region is the best place to display ads. However in feed, users are in content consuming mode and often scroll down to see more, and while their attention is finite, it is more spread over the different positions.

A related line of work is aggregated search [7, 15]. However, the focus is on "vertical selection": when there are several different specialized verticals (e.g., products, news, videos) that may be relevant to the query, which verticals to select. The task of where to embed the selected vertical(s) in a search result page receives less attention, which is the primary focus of our work here. [18] presented a user study investigating how the blending of different sources affect users' click behavior, and concluded that the position of results matters for the blended presentation. In our work, we model a variant of position effect to generate better blended results.

Another line of related work is ranking with multiple objectives for search [17, 19]. Our problem statement has two key differences with the setting of these studies. One is ranking with multiple sources of relevance, for example, from user feedback and from human labels; the other is ranking with multiple measures (e.g., diversity, freshness) when each measure employs the same label source. The solution framework has two major directions: combining multiple objectives to an aggregated or graded objective; and aggregating multiple models trained for each objective. [1] provides a good survey of other ideas in the space. In our setting, the label sources (i.e., revenue and engagement) are from very different dimensions that cannot be combined or graded easily. And existing methods cannot easily trade off one objective against another.

We adopt a constrained optimization formulation akin to [3] to define our proposed solution. This solution assumes each position is independent, which is reasonable for several use cases. Examples include homepage optimization [2], notification delivery [14, 26] and ads bidding/pacing [11] among others. However, in our application, each position is not necessarily independent and identically distributed (i.i.d.) and users' (click) behavior can be affected by previous items in the list. We use offline simulations to estimate optimal parameters that balance both objectives in such scenarios.

3 SYSTEM OVERVIEW

We first describe the overview of a large scale recommender system like the Feed. It is based on how the LinkedIn Feed operates, but many design choices and requirements should be representative of Feed ranking systems in the industry. The overall objective of Feed ranking is to produce a combined list of organic items and ads to achieve an optimal balance of engagement and revenue.

3.1 Organic items ranking

Engagement on organic items is often quantified by users' actions, including clicks, likes, comments, shares, conversion, and/or dwell time. Organic content is the main driver of user engagement, because the creator and content are relevant and of interest to viewers. The objective is to rank organic content is maximizing expected user engagement. There is some excellent work in literature covering the key factors to consider in building such models, both in terms of signals and from a system architecture perspective [4, 5, 20]. Ads also drives a small portion of engagement through users' click and/or conversion activities.

The system complexity of such a recommender system could be high because of the number of features that need to be fetched, as well as the scoring complexity of the model used. In Figure 2, the "Organic ranking" module represents this module.

3.2 Ads ranking

Revenue is the key utility driven by ads, that are ranked by an auction based on the expected revenue utility of each impression. The overall objective of the ads ranking is to maximize revenue. One example of the expected revenue utility is calculated by $bid * pCTR$, where $pCTR$ is the predicted click-through rate. For generalized second price (GSP) auctions [13], the utility is the next ad's $bid * pCTR$. Organic content does not have any revenue utility.

The ads ranking systems are often quite big and complex. The complexity of this module, denoted by "Ads ranking" in Figure 2, comes from the number of signals that need to be stored and fetched efficiently and the scoring of the pCTR (or other related utilities) models[6], which is quite similar to organic ranking. One source of significant added complexity, that is unique to ads, is budget pacing and pricing modules. They need to operate intelligently with a very low latency to ensure campaigns enter auctions with the optimal bid and are charged correctly and timely.

3.3 Blending layer

The key objective of the blending layer, denoted by the "Re-rank" module in Figure 2, is to merge the two ranked lists of organic items and ads, and create a unified list which maximizes revenue and engagement. There are a few requirements from such a module:

- **Low latency**. The overall system needs to return the ranked list of organic content and ads with low latency, since long loading time is not a good user experience. Also, since the organic and ads ranking modules are quite heavyweight, the latency requirements on the blending module may be very strict.
- **Respect prior ranking while blending**. For organic items, the first stage ranking is very accurate and informed (i.e., uses more signals). For ads, additionally, the auction dynamics of a GSP auction mean that the ordering determines the charge to advertisers. Changing the ranking of either list is highly undesired. Hence, our blending algorithms take this as a constraint which in turns helps simplify the solution. We treat the estimated utilities coming from the initial ranking layer as accurate and expect them to be monotonically decreasing in each ranked list.
- **Modeling velocity**. In many medium to large sized companies, there are big teams working on the organic ranking and ads ranking modules separately (often, there are multiple teams focusing on various aspects within each module). It is imperative to have a setup that allows each of those modules to iterate asynchronously. This was one of the biggest factors in our decision to adopt the two-phase ranking, and have most of the complexity of ranking nuances be handled in the respective ranking modules. In order to ensure seamless iterations for the upstream ranking modules, we have to automatically adjust (or "auto-tune") certain parameters in our blending algorithm. We will revisit this detail in the Section 5.4.

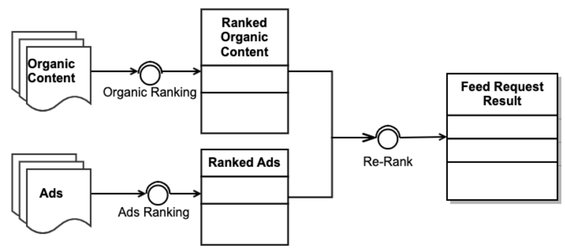

Figure 2: System Overview to rank organic content and ads together

A unified ranking alternative. A single unified module that ranks both item types together may afford some advantages because the blending decisions are made jointly with the ranking decisions, although this is a more complex problem to solve. Vickrey-Clarke-Groves (VCG) [22] would be a candidate auction mechanism in this case. However, there are quite a few disadvantages, especially for slightly large systems, that we have discussed above. Those factors led us to choose the two-phase solution.

3.4 User Experience Guardrails

The ads allocation problem on Feed also need some guardrails to prevent unintended member experiences. For instance, a member who is highly targeted by advertisers should not see a Feed full of ads, or several ads stacking at the top of the Feed. It is not a good user experience to see stacked ads on the bottom either. To handle such scenarios, we add user experience guardrails via two values, namely *top slot* and *min gap*. All the blending algorithms presented in Section 4 will account for these guardrails.

Definition 3.1. **top slot** defines the highest eligible position of the first ad. For example, *top slot* = 3 means that the first ad cannot be put before position 3.

Definition 3.2. **min gap** defines the minimum distance between two consecutive ads' positions. For example, *min gap* = 6 means that if there is an ad in position k, then the first eligible position for the next ad is $k + 7$.

3.5 Additional desired properties

There are two other considerations specific to ads that a good blending algorithm should satisfy. Results in Section 5.3.2 and 5.3.1 show how effective our algorithms are on these dimensions.

Adaptation to seasonality. Ads demand (i.e., the bids and budgets) may have a quarterly and annual periodic pattern. A good blending algorithm will increase (decrease) ad impressions when the revenue utilities are systemically higher (lower). In some applications, the organic engagement could also have such temporal patterns (e.g., higher engagement during holidays on platforms like Facebook) which the blending mechanism should help adapt to. This is in addition to the local fluctuations and variations in revenue and organic engagement utility that the blending algorithm is already helping to capitalize on.

Dynamic ads positions. Ads blindness is the term used to refer to the user behavior when a user gets used to seeing ads in a particular spot on the Feed or other surfaces. If users don't find ads as engaging, they may develop blindness for that regular spot, which may lead to not noticing relevant ads. If the blending mechanism places ads at different spots in different sessions for each user (to achieve its revenue and engagement maximization objectives), then that is an extra benefit to help counter phenomena like ads blindness.

With this context, we will present a set of blending algorithms that will have the following properties:

- Low computational complexity.
- Preserve the order of input rankings.
- Respect user experience guardrails.
- Have the flexibility to adapt to seasonal and local changes in ads demand and/or organic engagement.

4 PROBLEM FORMULATION AND ALGORITHMS

4.1 Problem Formulation

In the paper, we will use i to index impressions, and j to denote requests. The two key utilities corresponding to the two objectives are defined as follows:

Definition 4.1. **Expected Engagement Utility.** The expected engagement utility for an item being considered for impression i (in the feed request j) from a user is denoted as u_i. Particularly, u_i^o if it is an organic item, and u_i^a if it is an ad.

We omit the request index for brevity. Also, we do not introduce specific notations for items or item indices to keep the narrative clean. $\mathbf{u}^o$ refers to the engagement utilities of a list of candidates, where the list will be clear from the context.

Definition 4.2. **Expected Revenue Utility.** The expected revenue utility of an item being considered for impression i (from a user's feed request j) is denoted as r_i. Particularly, r_i^o for organic item and r_i^a for ad. Note that organic content has no revenue utility: $r_i^o = 0$.

Suppose that there are $\mathcal{J}$ requests in total. For every request j, there are N_j^o organic content candidates, and N_j^a ads to fill a total of $N_j = N_j^o + N_j^a$ slots. Organic candidates are ranked by expected engagement utility $\mathbf{u}^o$, while ads are ranked by expected revenue utility $\mathbf{r}^a$. Each item is associated with both utilities: r and u. For every impression slot i, we need to decide whether to pick the top item from the organic ranked list, or the one from ranked ads list, and then remove the winner from the corresponding ranked list. Let x_i be the variable to decide whether to show an ad in the impression (or slot) i. We formulate the problem as a constrained optimization problem as shown in Equation 1.

$$\begin{aligned}
&\text{maximize } \sum_i x_i r_i - \frac{w}{2}||\mathbf{x}||^2 \\
&\text{s.t. } \sum_i x_i u_i^a + (1 - x_i) u_i^o \geq C \\
&0 \leq x_i \leq 1, \forall i \in I
\end{aligned} \tag{1}$$

This formulation is to maximize revenue across all impressions (which are spread over all requests) such that the total engagement is larger or equal to some constant value (C in this case). C can be set to a fraction (i.e., δ) of the maximum possible engagement (e.g., when there are no ads), but that choice is an orthogonal consideration. *It should be noted that we could formulate the problem equivalently as an engagement maximization problem with a revenue constraint.* By varying the value of C above, and the revenue constraint in the alternate formulation, we would traverse the same Pareto-optimal curve between the two utilities.

We closely follow [3] (see specifically Section 3) to derive the solution. We cannot solve this constrained optimization on the fly since it is defined across several requests, some users may be new, and candidate items for each member are very dynamic. Hence, we use the Lagrangian dual from Equation 1 to obtain "optimal" primal serving plans for new requests as they arrive. The quadratic term in the objective function is added to introduce strong convexity into the problem and allow easy conversions from dual to primal solutions and vice-versa (the derivatives of the Lagrangian vanish in LPs [8]). Using the Lagrangian duals, the primal solution x_i can be obtained by Equation 2 under two conditions: 1. u_i^a, u_i^o and r_i^a are drawn from the same distribution as was the historical data used to solve the primal and obtain the optimal duals, 2. $w \longrightarrow 0$, which concentrates x_i to one of the vertices in the simplex unless there is a tie. $w \longrightarrow 0$ is also close to the original business problem.

$$x_i = \begin{cases} 1, & \text{if } r_i^a + \alpha u_i^a - \alpha u_i^o > 0 \\ 0, & \text{otherwise} \end{cases} \tag{2}$$

The parameter α (which is the optimal Lagrangian dual variable corresponding to the engagement constraint) is a function of C. Intuitively, this can be interpreted as a bid for engagement, This "shadow bid" **converts engagement into an equivalent monetization amount to enable direct comparison against revenue**. As we prioritize engagement with higher values of C, α will also increase. This ability to compare the two utilities on the basis of a unified currency is critical for any principled blending of organic items and ads. Table 1 shows the final value for both ads and organic content.

Ad	$r_i^a + \alpha u_i^a$
Organic content	αu_i^o

Table 1: Score to compare ad and organic content at each position

4.2 Re-Rank Algorithm

Algorithm 1: Re-Ranker

Input: Ranked list of organic content L^o with size N^o, Ranked list of ads L^a with size N^a. Each item associates with revenue utility and engagement utility.
Input: min gap M, top slot T, shadow bid α
Output: Merged list L of organic content and ads.

Initialize $i = 0, j = 0, k = 0, L = [], prevIdx = 0$;
while $i < N^o$ *and* $j < N^a$ **do**
  if $k > T$ *and* $(k - prevIdx) > M$ *and* $(r^a[j] + \alpha u^a[j]) > \alpha u^o[i]$ **then**
    $L.append(L^a[j++])$
    $prevIdx = k$
  else
    $L.append(L^o[i++])$
  $k++$
while $i < N^o$ **do**
  $L.append(L^o[i++])$
Return L

Algorithm 1 describes a lightweight Re-Rank algorithm for the Re-Rank module in Figure 2. It is applied online and assumes that we have already obtained the value of shadow bid. The algorithm is essentially a merge operation (akin to the merge operation in Merge Sort [12]) which combines two ranked lists without altering the ordering in either list. If all slots are identical, this algorithm provides the optimal solution, considering the objective (as per Equation 1) and the guardrails.

4.3 Obtaining the shadow bid

There are three ways to obtain the value of shadow bid:

(1) *Optimal dual from historical data*: If it is feasible to obtain the value of C (a key business decision), then we can solve the optimization problem to obtain the optimal dual. This dual may be inaccurate because of two reasons: discrepancy in historical data with online data; and/or all slots are not independent.
(2) *From online A/B tests*: The second issue mentioned above can be alleviated by actually running a line-search on the α through randomized testing (i.e., a series of A/B tests). This is feasible because there is only one unknown parameter. If we were blending multiple content streams, then the number of relevant dual variables would be greater, and we would have to more heavily rely on the previous method.
(3) *From offline replay*: Another possibility is to use offline simulations or "replays" to obtain appropriate values of α. We will revisit offline simulations in more detail in Section 5.

4.4 Position Bias

Users view items on Feed in a top-down fashion, and are more likely to view and engage (e.g., click) more on top positions compared to bottom ones. We refer to this effect as position bias. For utility estimation (for both organic and revenue), a common practice is to use impressed position as a feature in the training data, and use a default value in online inference because the actual position is not available at that time. This position-agnostic estimates are consistent with the " all slots are identical" assumption.
Let $\mathbf{w} = (w_1, w_2, \ldots)$ where $1 \geq w_1 \geq w_2 \ldots \geq w_n \geq 0$ be the positional effect on the utility estimates because of the aforementioned bias. If the position bias is invariant to item type, that is $\mathbf{w}^a = \mathbf{w}^o = \mathbf{w}$, the Re-Rank algorithm described in Algorithm 1 is still valid, since w_k is applied to both estimates in comparison and cancels out. If $\mathbf{w}^a \neq \mathbf{w}^o$ or w_k is not monotonically decreasing, special considerations will be needed and some sub-optimality could incur. In this paper, we will consider the case where position bias only depends on position and is non-increasing.

4.5 Gap Effect

Another important factor that impacts users' response is the gap between consecutive ads. We formally define gap as follows.

Definition 4.3. **Gap.** Let $d = k' - k$ denote the gap between two consecutive ads placed at positions k and k' where $k < k'$.

It is a bit challenging to estimate the pure gap effect because there is a confounding position effect (as the position of the ad also varies when we change the gap). We tested with some randomized buckets to observe the effect of different fixed gaps on ads CTR at a fixed position. As Figure 3 shows, ads CTR drops with smaller gaps. We examined the impact to organic items as well. However, unlike the significant effect of the gap on ads' CTR, we did not observe significant impact on users feedback to organic items. For actual estimation of the gap effect, we added a gap feature in the ads CTR estimation process (which has the position term to handle position bias already) as follows.

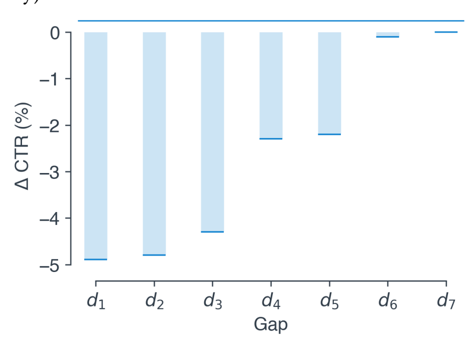

Figure 3: Ads CTR at fixed position conditioned on the ads gap where $d_1 < d_2 < \ldots < d_7$

The ads pCTR model (more details can be found in [6, 25]) has the following form.

$$g(E[y_{iu}]) = \boldsymbol{a}_{iu}^T \boldsymbol{z}_{iu} + \boldsymbol{b}_i^T \boldsymbol{v}_i \tag{3}$$

Where $g(y) = log\frac{y}{1-y}$ is the logit function; the label y_{iu} is 1 if the user u clicked on an ad i, otherwise 0 [1]; $\boldsymbol{a}_{iu}$ represents the overall feature vector for the (u, i) tuple, and includes user features, item features, and cross features between them; $\boldsymbol{b}_i$ is ID features related to the item including campaign id, advertiser id. The intercept term is covered inside the feature vector $\boldsymbol{a}_{iu}$ to simplify the notation. $\boldsymbol{z}_{iu}$ is the global coefficient vector for overall features, which is

[1]We reuse i to refer to items here because of the intuitive mapping and because impressions are not in consideration in this part

also called *fixed effect* coefficient, and $\boldsymbol{v_i}$ is item specific coefficient vector, i.e., the *random effect* coefficient.

We add the feature d representing gap. β is the corresponding coefficient. The Equation 3 is then changed to Equation 4.

$$g(E[y_{iu|d}]) = \boldsymbol{a_{iu}^T z_{iu}} + \boldsymbol{b_i^T v_i} + \beta d \tag{4}$$

We express the gap effect θ_d as the result of a function shown in Equation 5.

$$g(E[y_{iu|d}]) = \frac{1}{1 + exp(-g(y_{iu|d}))} = \frac{1}{1 + exp(-g(y_{iu}))}\theta_d \tag{5}$$

We can show the gap effect is from an exponential function of gap by rewriting Equation 5 to Equation 6. The approximation holds because ads CTR is quite small and $exp(-g(y_{iu})) >> 1$.

$$\theta_d = \frac{1 + exp(-g(y_{iu}))}{1 + exp(-g(y_{iu|d}))} \approx \frac{exp(-g(y_{iu}))}{exp(-g(y_{iu|d}))} = exp(\beta d) \tag{6}$$

With this estimated gap effect, the new Re-Rank score for an ad with gap d becomes

$$score(i|d) = \theta_d(r_i + \alpha u_i^a)$$

while the Re-Rank score for the organic item αu_i^o is unchanged. Figure 4 shows the diagram of the modified Re-Rank algorithm with gap effect. The only additional step is computing θ_d using Equation 6, which is trivial. Note that when $\beta = 0$, Re-Rank with gap effect falls back to the original Re-Rank Algorithm 1.

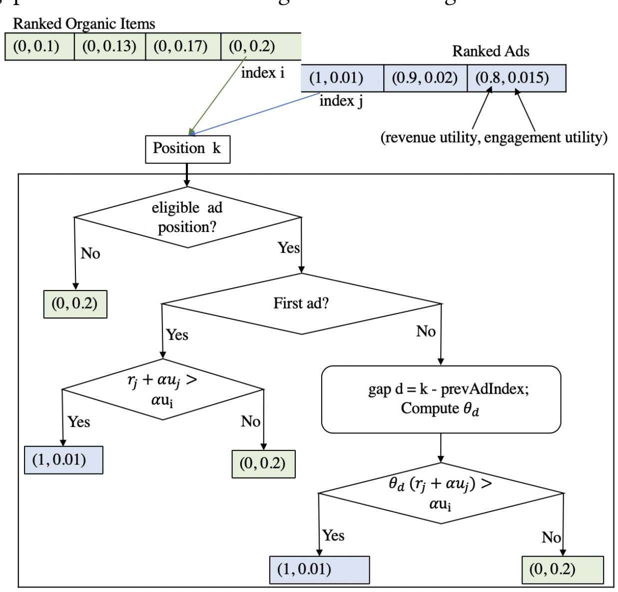

Figure 4: Re-Rank with gap effect. Here i, j are indices to traverse the ranked organic item and ads list respectively, as also in Algorithm 1. The 2-tuple represents the revenue and engagement utility.

4.6 Personalized Gap Effect

A natural extension is to model users' unique reaction to different gaps between ads, by considering cross features between user features and gaps. Let $\boldsymbol{c_{ud}}$ be the cross features and $\boldsymbol{\beta_{ud}}$ be the corresponding coefficients. Again, the global effect is included in the cross feature to simplify notations. Equation 7 shows the new form that represents the personalized gap effect.

$$\theta_{ud} = exp(\boldsymbol{c_{ud}^T \beta_{ud}}) \tag{7}$$

This formulation can improve the accuracy of the gap effect if indeed users have different preferences. It comes with an increased cost in computation time. Assuming there are N^u user features (which can be dynamic, and hence we can't pre-compute these values offline), for each eligible ad position at each request, we should score the gap effect with $(N^u + 1)$ time. The overall time complexity is then $O(N^u \times N^a + N^o)$. Table 2 shows the comparison of time complexity for each variant of our Re-Rank algorithm.

Algorithm	Time Complexity
Re-Rank Algorithm	$O(N^a + N^o)$
Re-Rank with Gap Effect	$O(N^a + N^o)$, because θ_d takes only O(1)
with Personalized Gap Effect	$O(N^a * N^u + N^o)$

Table 2: Compare time complexity for Re-Rank algorithm and its variants

In our historical data, we did not find much heterogeneity across users in the gap effect. Hence, we only ran experiments with the global gap effect. However, in applications where the gap effect varies across users, this extension can be handy. Also, any other form of estimating a multiplicative gap effect can also be used in an identical fashion in our Re-rank algorithm (as shown in Figure 4).

5 EXPERIMENTS

We now demonstrate the effectiveness of our proposed algorithms [2]. An offline evaluation mechanism is first defined, and a necessary extension proposed to handle the evaluation of the gap effect estimation. We then present our online experiment results that prove Re-Rank with gap effect is much more efficient than the vanilla version in both revenue and engagement. Finally, we revisit some of the desired properties of the blending algorithm and system (as outlined in Section 3) and see how we stacked up against them.

5.1 Offline Evaluation for Re-Rank Algorithms

Offline simulation or "replay" typically refers to the usage of historically observed data to estimate the effectiveness of new algorithms on live traffic. While it often has some approximation, it is nonetheless very useful in applications where the cost of an online test is high, or when the number of candidate treatments (i.e., algorithms in this case) are high. For offline replay, we need to define appropriate evaluation metrics first. Since we are dealing with two objectives, we define two evaluation metrics for the final ranking results. Given a ranking policy and total requests $\mathcal{J}$, the aggregated score for each objective is defined as follows:

Definition 5.1. **Discounted Cumulative Revenue (DCR)**

$$score(R) = \sum_{j \in \mathcal{J}} \sum_{k \in [N_j]} w_k r_{k,j}$$

where $[N_j]$ is the ranked list for the request j with size N_j, k is the position for each item in the ranked list in ascending order.

[2] In all our evaluations, some numbers have been obfuscated since exact revenue lifts, or gaps being used in production are deemed sensitive information. We report relative improvements and intelligently masked results to (hopefully) demonstrate the effectiveness of our methods.

Definition 5.2. **Discounted Cumulative Engagement (DCE)**

$$score(E) = \sum_{j \in \mathcal{J}} \sum_{k \in [N_j]} w_k u_{k,j}$$

For the positional bias (introduced in Section 4.4), without loss of generality, we assume $\mathbf{w} = (w_1, w_2, \ldots)$ are the non-increasing weights of positions, i.e., $1 \geq w_1 \geq w_2 \ldots \geq w_n \geq 0$. This captures a global bias towards position of a list: an item placed in the top position of the list is more likely to be impressed and clicked than an item placed below it. The weight vector we use is as follows:

$$w_k = \frac{1}{log_2(k+1)}$$

This is also the weight used in discounted cumulative gain (DCG) and normalized discounted cumulative gain (NDCG) [9], which are very popular metrics in evaluating ranking algorithms. Our evaluation metrics are essentially DCG of revenue and DCG of engagement. Each value of shadow bid α will produce a pair of metrics. By varying α, we can obtain a Pareto-optimal frontier [10] for revenue and engagement. Offline replay can be used to generate or identify a good candidate set of shadow bids (or narrow down from a larger list). These promising shadow bids can then be evaluated in online A/B tests.

Data Setting. We randomly sample around 100,000 feed requests from the historical logs of the LinkedIn Feed. Each request has hundreds of organic content candidates and ads candidates. Revenue utilities and engagement utilities for each candidate item, estimated with a default fixed position, are generated before the Re-Rank layer, and are tracked in the logs. A better ad allocation algorithm should map to a better Pareto-optimal tradeoff curve. The value of β (the gap effect estimator defined in Section 4.5) decides the shape of the curve, and the hyper-parameter α (i.e., the shadow bid) decides the operating point on the curve. We learn β through CTR prediction task, which has marginal improvement of accuracy with gap effect.

We picked several values of β to demonstrate the difference. For each β, we select a set of αs. We re-rank feed requests based on the chosen β and α, then compute DCE and DCR. Figure 5 shows the performance curves with different values of β.

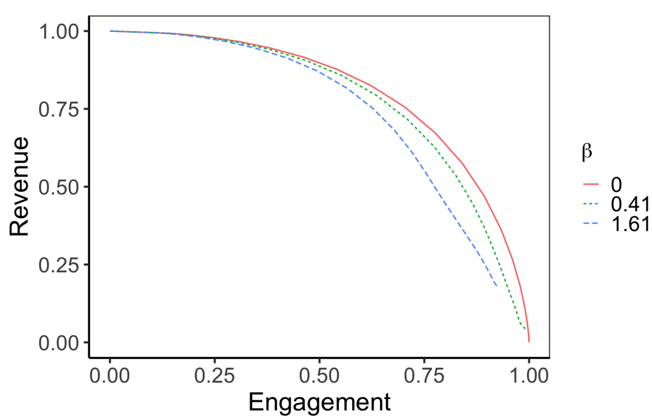

Figure 5: Revenue and engagement tradeoff curve. Each curve maps to a different β in Re-Rank with gap effect algorithm. Both x- and y-axis are scaled such that numbers are between 0 and 1.

The results suggest that $\beta = 0$ is the most efficient value. *This is because the evaluation metrics share the assumption that all slots are independent and there is no gap effect.* To bridge the gap in the faulty evaluation metric, we compose a gap weight, which is similar to the position weight used in DCR, and define a more appropriate evaluation metric for this context in Definition 5.3. The DCE metric remains the same, since organic items did not demonstrate any gap effect.

Definition 5.3. **Discounted Cumulative Revenue with Gap Effect**

$$score(R) = \sum_{j \in \mathcal{J}} \sum_{k \in [N_j]} \frac{r_{k,j}}{log_2(k+1)} log_{10}(d+c)$$

where $log_{10}(d+c)$ is fitted with observed gap effect in Figure 3. The exact value of c is undisclosed since it reflects very specific user behavior information.

With this modified metric, we re-evaluate the trade-off for the same set of β and α. Figure 6 shows that $\beta = 0.41$ results in the best trade-off curve, and $\beta = 0$ has the worst performance.

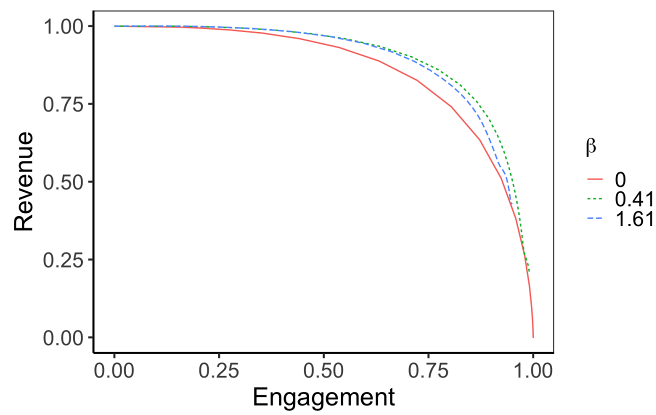

Figure 6: Revenue and engagement tradeoff curve with the modified DCR. A gap effect aware evaluation metric is necessary to show the usefulness of gap effect aware algorithms.

As we see from the above results, different evaluation metrics can lead to diametrically opposite conclusions. The majority of existing literature relies on DCG (or NDCG), which has a strong assumption, to evaluate ranking problems. **Blending different types of items can violate such assumptions in certain cases**, hence it's important to select an appropriate evaluation metric. To demonstrate the fidelity of the newly proposed evaluation metric, we now compare its online performance to the $\beta = 0$ variant (which performed best with DCG).

5.2 Online Results

Online A/B tests that directly measure each objective with live traffic, are the most reliable measure of any algorithm's success. We do not disclose the exact values of α and β as they are deemed sensitive. However, the methods that we have described so far should enable practitioners to estimate appropriate values for their own applications. Since each algorithm maps to a curve by varying α, it is unfair to compare two different operating points from two curves. Since it's an ads allocation problem, we instead fix the amount of ads impressions for both algorithms.

We test four groups A, B, C, D, each of which is compared to a group with matched ads impressions from a baseline algorithm (no gap effect, i.e., $\beta = 0$). Each group corresponds to a certain number of ads impressions with gap effect (i.e. with the same β ($\neq 0$) and different αs). The baseline algorithm is Re-Rank algorithm without gap effect (i.e. $\beta = 0$, with appropriate α values to match impression volume with its corresponding gap-effect bucket). *We ran each variant on 2% of live traffic on LinkedIn for a week.* Using group A's impression volume as the baseline, group B has 4% more impressions than group A, group C has 10% more impressions than group A, and group D has 20% more impressions than group A. Table 3 shows the results. It is clear that re-rank with gap effect is **much more efficient** than the baseline.

Online results also directionally match the offline curves in Figure 6. The difference in the middle is bigger as shown in Groups B and C. In the extreme cases of all or no ads, all Re-Rank algorithms will achieve same engagement and same revenue (i.e., 0). Group D gets close to the extreme because of the guardrails and high ad impressions.

Test group (Impressions)	Engagement lift	Revenue lift
A (N)	0.1%	2%
B (1.04N)	0.8%	3.9%
C (1.1N)	0.69%	5.32%
D (1.2N)	0.37%	1.00%

Table 3: Online A/B test results of Re-Rank with gap effect v.s. Re-Rank algorithm. Each row is a comparison under a fixed amount of ads impressions, and group A serves as the baseline. All numbers are statistically significant with p-val < 0.0001.

The Re-Rank with gap effect algorithm enables organic items with high engagement utilities to be placed in better positions that align better with users' preference, it achieves moderate organic engagement lift. The lift number on engagement metrics looks small, but the absolute value is significant, especially since we do not modify the engagement estimate directly but only change ads allocation results. The revenue lift largely comes from the average cost per click (CPC) increase. Ads with high expected revenue utility has a better chance to show at high positions, which results in increased impressions for such ads. These ads with high revenue utility also often have higher CPC (revenue utility and CPC are highly correlated). We used Mantel-Haenszel Method [16] to compute reweighed CPC that takes stratification into account to address Simpson's paradox. The reweighed CPC is neutral, and proves that revenue lift is not from a systematic price increase, but due to more "expensive" ads getting more results.

5.3 Properties of Re-Rank with Gap Effect

We now revisit some of the desired properties of the blending layer, and see how Re-Rank performs on those aspects.

5.3.1 Diverse Ads Allocation Experience. Comparing the Re-Rank with gap effect algorithm to that without gap effect, ads are placed in more diverse positions. More than 70% users see ads with more than one gap in the new algorithm versus 40% users in the simple Re-Rank algorithm. Figure 7 shows the distribution of users for each unique number of ads gaps. Only users with at least two feed requests per day were included. Such diverse experience can help in reducing ads blindness.

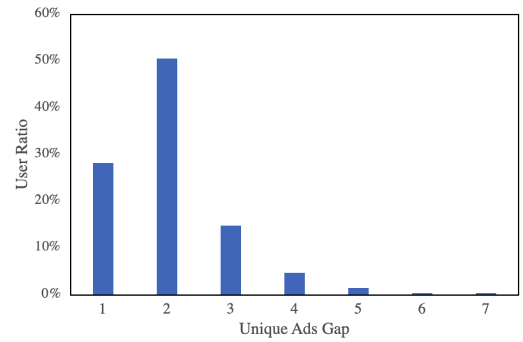

Figure 7: Most users experienced some diversity in the set of gaps between ads when Re-Rank with gap effect was used.

5.3.2 Elasticity to Demand. Our ads ecosystem, like on most other platforms, has a strong quarterly and annual seasonality pattern in demand. The increase in demand usually means an increase in total budget and the total number of campaigns. As a consequence, the proportion of ads with higher expected revenue utility per impression also increases. In contrast, the decrease in demand will reduce the portion of high revenue utility ads. If the quantity of ads impressions is fixed (e.g., with fixed slotting), the increase of demand will lead to more competition hence price per impression will increase. In this case, advertisers' return on investment (ROI) will get hurt. On the contrary, when the demand gets lower, we could provide a better user experience by showing less ads, compared to the same fixed amount of ads shown in high demand season.

It is critical to use an efficient ads allocation algorithm that is responsive to the change of demand. Without hurting engagement, when the demand is high, ads can win more (and better) positions, and when the demand is low, organic items can "take back" these coveted positions from ads. We conducted a 6 months period online A/B test for our Re-Rank with gap effect algorithm, and observed ads position distribution shifts between low demand and high demand period, as shown in Figure 8.

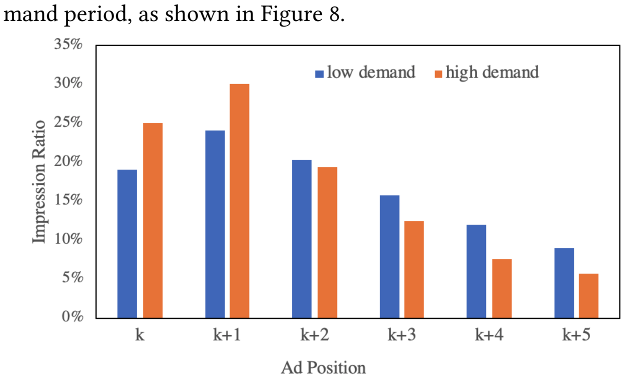

Figure 8: The impression distribution of the position for the second ad shifts more to higher positions at high demand season, with significant upside to revenue.

5.4 Automated Calibration

In our design, utility scores for organic content and for ads are provided by two separate ranking systems, as described in Section

3. Each ranking system operates and evolves on its own. Such decoupling facilitates modeling velocity for both organic content and ads, but also imposes an additional challenge on the Re-Rank layer. If the score distribution changes dramatically from either upstream ranking module, it could have significant, and inadvertent impact on the final ranking (e.g., ad CTR prediction goes up by 10%). As a result, we need to solve a calibration challenge.

If the utility score is point-wise prediction, e.g., the probability of click, we leverage **isotonic regression** [24] to bring predicted probability close to observed user response. If the utility score is not point-wise prediction but a combination of multiple point-wise predictions, we leverage **Thompson Sampling** [21] to find a global calibration factor for the new score distribution with online traffic. The objective is to match a metric of interest between a new model and the control model. We assume the metric to match is a function of the calibration factor, and that function is drawn from a Guassian Process prior with a covariance function. The algorithm has explore and exploit stages, and repeatedly estimates the value of the factor until convergence. Our calibration framework works well in practice. It facilitates the process to ramp new models in the first rank layer and reduces manual tuning burden significantly.

6 CONCLUSION

In this paper, we discuss different approaches to allocate optimal positions to ads in a newsfeed application to obtain an optimal trade-off between revenue and engagement. Using the foundations of constrained optimization, we present a set of blending algorithms (termed Re-Rank), which are optimal under various assumptions (all slots i.i.d., positional bias, gap effect). We discuss many effects which are important considerations for slotting ads into feed through randomized online tests performed on the LinkedIn feed.

To the best of our knowledge, this is the first reported work on this problem, which is very critical in the internet industry today. We are optimistic that this will help practitioners design their feed applications with monetization considerations in a more optimal fashion, and also encourage future publications in this space.

In terms of future directions, devising algorithms when the gap effect has more complex forms, or positional effects are non-monotonic are interesting challenges. Proving a general bound on performance for the current algorithm in the general setting would also be useful. Finally, ads allocation is the end component in a very dynamic ads system and there are feedback loops (e.g., fewer ad impressions can increase advertiser bids in an automated bidding system). Being more cognizant of those effects and eventually optimizing the bigger system can be very useful.

REFERENCES

[1] Gediminas Adomavicius, Nikos Manouselis, and YoungOk Kwon. 2011. Multi-criteria recommender systems. In *Recommender systems handbook*. Springer, 769–803.

[2] Deepak Agarwal, Shaunak Chatterjee, Yang Yang, and Liang Zhang. 2015. Constrained optimization for homepage relevance. In *Proceedings of the 24th International Conference on World Wide Web*. ACM, 375–384.

[3] Deepak Agarwal, Bee-Chung Chen, Pradheep Elango, and Xuanhui Wang. 2012. Personalized Click Shaping Through Lagrangian Duality for Online Recommendation. In *SIGIR* (Portland, Oregon, USA). ACM, New York, NY, USA, 485–494.

[4] Deepak Agarwal, Bee-Chung Chen, Rupesh Gupta, Joshua Hartman, Qi He, Anand Iyer, Sumanth Kolar, Yiming Ma, Pannagadatta Shivaswamy, Ajit Singh, et al. 2014. Activity ranking in LinkedIn feed. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*. 1603–1612.

[5] Deepak Agarwal, Bee-Chung Chen, Qi He, Zhenhao Hua, Guy Lebanon, Yiming Ma, Pannagadatta Shivaswamy, Hsiao-Ping Tseng, Jaewon Yang, and Liang Zhang. 2015. Personalizing linkedin feed. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. 1651–1660.

[6] Deepak Agarwal, Bo Long, Jonathan Traupman, Doris Xin, and Liang Zhang. 2014. Laser: A scalable response prediction platform for online advertising. In *Proceedings of the 7th ACM international conference on Web search and data mining*. 173–182.

[7] Jaime Arguello, Fernando Diaz, Jamie Callan, and Ben Carterette. 2011. A methodology for evaluating aggregated search results. In *European Conference on Information Retrieval*. Springer, 141–152.

[8] Stephen Boyd and Lieven Vandenberghe. 2004. *Convex optimization*. Cambridge university press.

[9] Christopher Burges, Tal Shaked, Erin Renshaw, Ari Lazier, Matt Deeds, Nicole Hamilton, and Gregory N Hullender. 2005. Learning to rank using gradient descent. In *Proceedings of the 22nd International Conference on Machine learning (ICML-05)*. 89–96.

[10] Yair Censor. 1977. Pareto optimality in multiobjective problems. *Applied Mathematics and Optimization* 4, 1 (1977), 41–59.

[11] Ye Chen, Pavel Berkhin, Bo Anderson, and Nikhil R Devanur. 2011. Real-time bidding algorithms for performance-based display ad allocation. In *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 1307–1315.

[12] Thomas H Cormen, Charles E Leiserson, Ronald L Rivest, and Clifford Stein. 2009. *Introduction to algorithms*. MIT press.

[13] Benjamin Edelman, Michael Ostrovsky, and Michael Schwarz. 2007. Internet advertising and the generalized second-price auction: Selling billions of dollars worth of keywords. *American economic review* 97, 1 (2007), 242–259.

[14] Yan Gao, Viral Gupta, Jinyun Yan, Changji Shi, Zhongen Tao, PJ Xiao, Curtis Wang, Shipeng Yu, Romer Rosales, Ajith Muralidharan, and Shaunak Chatterjee. 2018. Near Real-time Optimization of Activity-based Notifications. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. ACM, 283–292.

[15] Mounia Lalmas. 2011. Aggregated search. In *Advanced Topics in Information Retrieval*. Springer, 109–123.

[16] Nathan Mantel and William Haenszel. 1959. Statistical aspects of the analysis of data from retrospective studies of disease. *Journal of the National Cancer Institute* 22, 4 (1959), 719–748.

[17] Mario Rodriguez, Christian Posse, and Ethan Zhang. 2012. Multiple objective optimization in recommender systems. In *Proceedings of the sixth ACM conference on Recommender systems*. ACM, 11–18.

[18] Shanu Sushmita, Hideo Joho, Mounia Lalmas, and Robert Villa. 2010. Factors affecting click-through behavior in aggregated search interfaces. In *Proceedings of the 19th ACM international conference on Information and knowledge management*. 519–528.

[19] Krysta M Svore, Maksims N Volkovs, and Christopher JC Burges. 2011. Learning to rank with multiple objective functions. In *Proceedings of the 20th international conference on World wide web*. ACM, 367–376.

[20] Liang Tang, Bo Long, Bee-Chung Chen, and Deepak Agarwal. 2016. An empirical study on recommendation with multiple types of feedback. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. 283–292.

[21] William R. Thompson. 1933. On the likelihood that one unknown probability exceeds another in view of the evidence of two samples. *Biometrika* 47 (1933), 285–294.

[22] Hal R Varian and Christopher Harris. 2014. The VCG auction in theory and practice. *American Economic Review* 104, 5 (2014), 442–45.

[23] Bo Wang, Zhaonan Li, Jie Tang, Kuo Zhang, Songcan Chen, and Liyun Ru. 2011. Learning to advertise: how many ads are enough?. In *Pacific-Asia Conference on Knowledge Discovery and Data Mining*. Springer, 506–518.

[24] 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.

[25] XianXing Zhang, Yitong Zhou, Yiming Ma, Bee-Chung Chen, Liang Zhang, and Deepak Agarwal. 2016. Glmix: Generalized linear mixed models for large-scale response prediction. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 363–372.

[26] Bo Zhao, Koichiro Narita, Burkay Orten, and John Egan. 2018. Notification Volume Control and Optimization System at Pinterest. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. ACM, 1012–1020.