Constrained Optimization for Homepage Relevance

Deepak Agarwal
LinkedIn
2029 Stierlin Ct
Mountain View, CA, USA
dagarwal@linkedin.com

Shaunak Chatterjee
LinkedIn
2029 Stierlin Ct
Mountain View, CA, USA
shchatterjee@linkedin.com

Yang Yang
LinkedIn
2029 Stierlin Ct
Mountain View, CA, USA
yyang@linkedin.com

Liang Zhang
LinkedIn
2029 Stierlin Ct
Mountain View, CA, USA
lizhang@linkedin.com

ABSTRACT

This paper considers an application of showing promotional widgets to web users on the homepage of a major professional social network site. The types of widgets include address book invitation, group join, friends' skill endorsement and so forth. The objective is to optimize user engagement under certain business constraints. User actions on each widget may have very different downstream utilities, and quantification of such utilities can sometimes be quite difficult. Since there are multiple widgets to rank when a user visits, launching a personalized model to simply optimize user engagement such as clicks is often inappropriate. In this paper we propose a scalable constrained optimization framework to solve this problem. We consider several different types of constraints according to the business needs for this application. We show through both offline experiments and online A/B tests that our optimization framework can lead to significant improvement in user engagement while satisfying the desired set of business objectives.

1. INTRODUCTION

Users spend a significant amount of time online consuming information and connecting with others on web sites. There are sites that are general and provide a wide variety of content like the homepage of Yahoo!, MSN and AOL; some are more domain specific that cater to content and information in a narrow area like sports, finance, movies and so on; we also have social network sites like Facebook, LinkedIn and Twitter that allow users to grow and disseminate information through their networks.

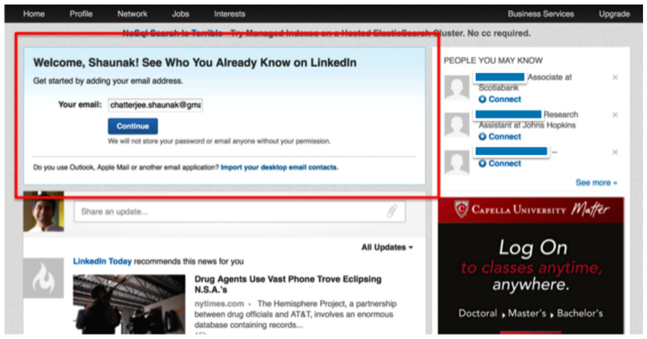
The homepage of such a web site serves as a distribution channel with deep links to various other pages. Recommending the best links personalized for each user is an important problem. Often, this is solved by using machine learned methods that optimize the likelihood of click (CTR) on various links of the homepage. But in many cases, the downstream values associated with clicks are disparate and hard to convert into a single value currency. For instance, a click that helps a user connect with another user may have different value compared to the one that sends the user to a news article page. In fact, the values may even depend on the user segment. A user who is new to a social network with very few connections may find a new connection more valuable than reading a news article. A user with thousands of connections may reach a point of diminishing returns when making a new connection. Such insights on values (by user segments) are often obtained through various macro studies conducted by organizations.

WWW 2015 Companion, May 18–22, 2015, Florence, Italy.
ACM 978-1-4503-3473-0/15/05.
http://dx.doi.org/10.1145/2740908.2745398 .

Shaping traffic downstream in such scenarios often entails balancing multiple competing objectives and becomes a non-trivial task. One approach is to create multiple sections on the homepage, each catered to a specific task. For instance, one could have a section to recommend articles, some to show ads, some to recommend people to connect to, and so on. This approach of *slotting* content is rigid since it takes a "one size fits all" approach. Another approach is to have business rules that decide the priorities of content (for different user segments). While this may help the business achieve its objectives in terms of downstream utilities, it often provides a suboptimal solution in terms of driving overall homepage engagement. A flexible framework that can simultaneously optimize metrics like CTR while incorporating various business constraints is desirable to mitigate inefficiencies introduced by using slotting and/or business rules.

An ideal approach to this problem would be to associate a long-term value to each click measured in a single currency. But this is often difficult due to issues with attribution and long-term value estimation. Also, a site often seeks to optimize more than one objective. For instance, a site may have multiple goals of maximizing user visit frequency by providing engaging content, increasing advertising revenue, increasing the virality of the network by connecting users to each other, growing audience base by encouraging users to invite their friends to join a site, and so on. One feasible approach is to optimize myopic objectives like CTR but subject to some constraints that are intuitive and help in driving long-term value the site seeks to attain. We provide a framework to solve this. Our approach is based on formulating the problem in an optimization framework and depends on input that is intuitive and easy to specify from a business perspective. While similar multi-objective optimization formulations have been proposed in various contexts before, this is the first instance that reports results based on experiments conducted on live traffic. At the time of writing, the method was fully deployed on a large social networking site.

Our **contributions** are as follows. We provide a principled framework to recommend content on a web site that optimizes short-term metrics like CTR while simultaneously satisfying various business constraints that are important to drive long-term value. We provide extensive analysis and describe our experience while deploying such a sophisticated method on a module on the homepage of a large social networking website (See a snapshot of the module in Figure 1 for the module on the site). To the best of our knowledge, this is the first time such an approach have been fully deployed on a real application used by hundreds of millions of users. Our results clearly show that the new approach significantly outperforms a strategy based on business rules that was being used by the site before our work. We also describe various empirical approaches we took to allow the optimization to work at scale.



~~NoSql Search Is Terrible - Try Managed Indexes on a Hosted ElasticSearch C~~luster. No cc required.

Welcome, Shaunak! See Who You Already Know on LinkedIn

Get started by adding your email address.

Your email: chatterjee.shaunak@gma

Continue

We will not store your password or email anyone without your permission.

Do you use Outlook, Apple Mail or another email application? Import your desktop email contacts.

[redacted] Associate at Scotiabank

[redacted] Research Assistant at Johns Hopkins

[redacted] --

LinkedIn Today recommends this news for you

Drug Agents Use Vast Phone Trove Eclipsing N.S.A.'s

nytimes.com · The Hemisphere Project, a partnership between drug officials and AT&T, involves an enormous database containing records...

Figure 1: A snapshot of the "address book import" widget.

The remainder of the paper is structured as follows. Section 2 introduces the problem formulation. The details of the constrained optimization framework and various applications of whole-page optimization are enumerated in Section 3. The online experimental setup that we used to demonstrate the utility of our approach and the experimental results are presented in Section 4. The related references are summarized in Section 5. We summarize our contributions and identify several future directions in Section 6.

2. PROBLEM FORMULATION

We consider the problem of maximizing user engagement (e.g. number of clicks) while satisfying certain business constraints. Typical examples of such constraints include: the minimum number of impressions certain items are served with, or in some cases the minimum number of clicks obtained by certain items. The constraints are usually imposed because clicks on different items can lead to different downstream utilities. For example, in a social network site, a click on accepting to invite people in the address book to join the social network maybe more valuable to the site than a click on a shared update from friends. However, the CTR for the latter item might be much higher than the CTR of the former item. Therefore, ranking items using predicted CTR alone may lead to an undesirable downstream traffic distribution; coarse constraints on downstream traffic volumes ensure the business requirements in terms of value are incorporated by the serving algorithm.

2.1 Homepage Widget Relevance

We introduce our motivating application to provide background and context. We focus on the problem of recommending personalized promotional widgets in the most prominent spot (shown in Figure 1) on the homepage of a major professional networking site. The purpose of this module is to promote some specific "channels", increase user awareness about certain product offerings and drive key business objectives. This module is shown to hundreds of millions of users on a weekly basis and generates significant user engagement actions and downstream page views. Some examples of the widgets include:

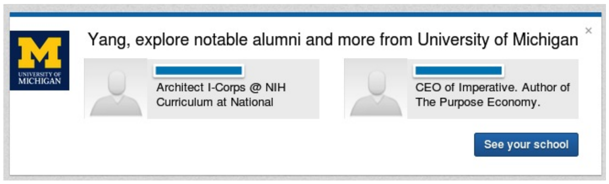
- *Address book import*: This widget directs a user to import her address book, in order to find more contacts to connect to, or send emails to her friends to invite them to become a member of the network.
- Education: This widget directs the user to the education channel page (see Figure 2a) which includes schools she attended, university rankings, field of study to explore, and so forth. This widget is designed mainly for students to find out education opportunities in the world for the field she is interested in.
- Who viewed my profile: This widget (Figure 2b) directs the user to a page which shows the list of recent visitors to the user's profile.
- People you may know (PYMK): This widget shows a list of recommended persons for the user to connect to.
- Endorsements: This widget presents the user with a small set of selected <name, skill> pairs (Figure 2c) among the user's connections, and the user can choose to endorse that one of her connections possesses the specified skill.
- Profile edit: This widget shows a specific link for the user to click and edit a certain section in her profile.
- Content-channel recommender: This widget (Figure 2d) shows various types of content channels that the user can follow, in order to get notifications when those channels publish new articles.
- Three widgets (variants of address book import, PYMK and profile edit) with modified interfaces and wording designed for new and dormant users to on-board them.

There is only one slot to render a widget on the homepage, but there are quite a few candidates. Different widgets have highly varying click through rates as well as diverse downstream utilities (the latter is quite evident from the list above). For example, the endorsement widget has a much higher CTR than address book import. However, from the business perspective, an address book import action may be worth more than a click to endorse someone's skills. Hence, optimizing for CTR alone may serve more endorsement widgets than address book imports, but this will not satisfy the business value requirements. In this paper, we try to solve this problem by using constrained optimization which tries to maximize the user engagement while satisfying several constraints derived from business requirements.

Notations: Throughout the paper we always index a user by i, an item (e.g. widget) by j, and context (e.g. time) by t. We assume the total number of items is J. We denote the binary response of whether the user i clicks the item j at context t as y_{ijt}, and correspondingly, we let x_{ijt} represent the probability of serving item j to user i at context t, and p_{ijt} be the probability of user i clicking item j at context t.

2.2 CTR Prediction

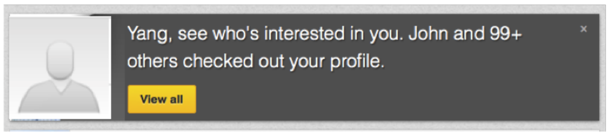
Before describing how we formulate the constrained optimization problem, let us first describe a formulation of the model to predict the CTR, which is an essential component of the framework. Given user i, item j, and context t, we need to build a model to predict the probability of the user clicking the item at the context, which is denoted as p_{ijt}. This has been a very popular topic in recent literature such as [1, 2, 7, 9, 10]. Since CTR prediction is not the main focus of this paper and is an independent component, without loss of generality, in this section we present a simple feature-based per-item logistic regression model. Other more advanced CTR prediction approaches can be applied to our constrained optimization framework without any difficulties.


Yang, explore notable alumni and more from University of Michigan

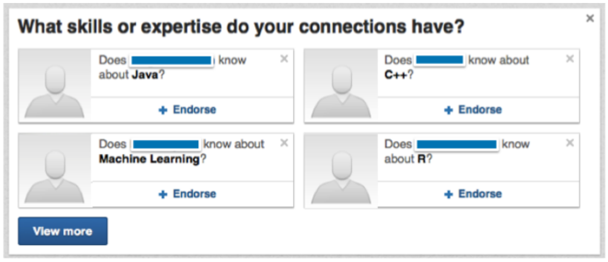
Architect I-Corps @ NIH Curriculum at National

CEO of Imperative. Author of The Purpose Economy.

(a) Education


Yang, see who's interested in you. John and 99+ others checked out your profile.

(b) Who viewed my profile


What skills or expertise do your connections have?

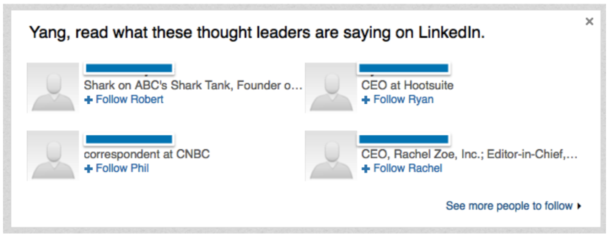
Does know about **Java**?

Does know about **C++**?

Does know about **Machine Learning**?

Does know about **R**?

(c) Endorsements


Yang, read what these thought leaders are saying on LinkedIn.

Shark on ABC's Shark Tank, Founder o...

CEO at Hootsuite

correspondent at CNBC

CEO, Rachel Zoe, Inc.; Editor-in-Chief,...

(d) Content-channel recommender

Figure 2: Snapshots of various widgets

Denote the entire feature set as $\boldsymbol{z}_{ijt}$, which includes all the user, item and context features, and interactions among these features if necessary. Also, denote the feature set for user i only as $\boldsymbol{f}_i$. Note that $\boldsymbol{f}_i \subseteq \boldsymbol{z}_{ijt}$. Given a binary response y_{ijt}, it is natural to assume a Bernoulli model

$$y_{ijt} \sim Bernoulli(p_{ijt}), \tag{1}$$

where p_{ijt} can be modeled with a logistic link function

$$\log(\frac{p_{ijt}}{1-p_{ijt}}) = (1, \boldsymbol{z}_{ijt})'\boldsymbol{\beta} + (1, \boldsymbol{f}_i)'\boldsymbol{\delta}_j, \tag{2}$$

where $\boldsymbol{\beta}$ is the set of the global coefficients, and $\boldsymbol{\delta}_j$ for all $j = 1 \cdots J$ are the per-item coefficients. Both $\boldsymbol{\beta}$ and $\boldsymbol{\delta}_j$ can have Gaussian priors such as

$$\boldsymbol{\beta} \sim N(\boldsymbol{0}, \sigma_\beta^2 \boldsymbol{I}), \quad \boldsymbol{\delta_j} \sim N(\boldsymbol{0}, \sigma_\delta^2 \boldsymbol{I}), \forall j = 1 \cdots J. \tag{3}$$

Note that the term $(1, \boldsymbol{z}_{ijt})'\boldsymbol{\beta}$ is purely feature-based, while $(1, \boldsymbol{f}_i)'\boldsymbol{\delta}_j$ is a per-item component. If there are very few observations for an item j, due to the prior shrinkage, $\boldsymbol{\delta}_j$ will be very close to $\boldsymbol{0}$. On the other hand, when an item j has a lot of data, $\boldsymbol{\delta}_j$ will become a meaningful set of coefficients that represent both the residual item-specific popularity and item affinity to the user features.

The posterior distribution of $\boldsymbol{\beta}$ and $\boldsymbol{\delta}_j$ can be obtained via Laplace approximation or variational approximation [8]. Denote the posterior mean of $\boldsymbol{\beta}$ and $\boldsymbol{\delta}_j$ as $\hat{\boldsymbol{\beta}}$ and $\hat{\boldsymbol{\delta}_j}$ respectively, the predicted probability of user i clicking on item j at context t becomes

$$\hat{p}_{ijt} = \frac{1}{1 + \exp(-(1, \boldsymbol{z}_{ijt})'\hat{\boldsymbol{\beta}} - (1, \boldsymbol{f}_i)'\hat{\boldsymbol{\delta}_j})}. \tag{4}$$

The estimated click probability $\hat{p}_{ijt}$ is used as input to our constrained optimization problem in later sections.

2.3 Optimizing CTR with Constraints

While we have established that optimizing click through rate for the homepage widget module is not the best thing to do, it is usually also difficult to directly measure and calibrate downstream utilities and express them in a single currency for the different widgets. In our application, before we launched the relevance model, there was already a baseline system that serves widgets using a set of hand-tuned rules: in a high-level they give a global priority ordering of widgets to serve to the users. As a result, besides improving performance by launching models to maximize CTR, we also need to make sure we do not change the status quo impression and click distribution for each widget significantly. We formulate such business needs as constraints in the optimization.

It is worth noting that such scenarios would usually occur any time a relevance model tries to replace an existing system based on rules. The inability of expressing the utility of various downstream actions in one currency, is also common for any application on a website that contributes to multiple downstream funnels. Initially, we also attempted to elicit values for different widgets on a relative scale by interacting with product managers and business executives. We found it was easier to specify utilities in terms of downstream traffic composition instead of relative value.

2.3.1 *Optimization via Relative Utilities*

Our first formulation tries to solve a global per-widget relative utility given a pre-specified impression allocation distribution. Such pre-specified allocation of impressions per widget is obtained from the existing baseline system, such that we do not generate any negative downstream impact if relevance model is launched. Let I_j be the widget-specific impression constraint, specifying the minimum number of impressions that have to be allocated to widget j. For the widget module, I_j can be simply obtained via the number of impressions allocated to widget j in the baseline system with some tolerance (e.g., the minimum threshold can be 95% of the current number of impressions that item j is receiving). Also, say w_j is the relative downstream utility for widget j if it gets clicked. If we take a soft max approach, the serving probability x_{ijt} for user i and widget j at context t becomes:

$$x_{ijt} = \frac{w_j \hat{p}_{ijt}}{\sum_j w_j \hat{p}_{ijt}}. \tag{5}$$

The relative item-specific utilities w_j are hence obtained by solving the following equations:

$$\sum_{i,t} x_{ijt} = I_j, \forall j \in \{1, \dots, J\},$$
$$\sum_{j \in J} w_j = 1, \quad w_j \geq 0, \forall j \in \{1, \dots, J\}. \tag{6}$$

Note that the constraint $\sum_{j\in J} w_j = 1$ is to make sure the solution of such formulation is unique, as long as $\sum_j I_j = \sum_{i,t} 1$. w_j can be considered as the relative downstream utility if item j gets clicked.

2.3.2 Click Optimization with Impression Constraints

A better alternative would be to directly solve for personalized serving distribution variables x_{ijt}, for all users i, widgets j, and contexts t, where the objective is to maximize the total number of clicks, while satisfying the desired impression distribution. The optimization problem can thus be formulated as:

$$\min_{x_{ijt}} - \sum_{i,j,t} x_{ijt}\hat{p}_{ijt}, \\ s.t. \sum_{i,t} x_{ijt} \geq I_j, \forall j, \quad x_{ijt} \geq 0, \forall i,j,t, \quad \sum_j x_{ijt} = 1, \forall i,t. \tag{7}$$

Here, $\sum_{i,j,t} x_{ijt}\hat{p}_{ijt}$ is the expected total number of clicks, which represents the user engagement that we want to optimize for. The other two constraints $x_{ijt} \geq 0, \forall i,j,t$, and $\sum_j x_{ijt} = 1$ are to make sure x_{ijt} can represent the probability of serving item j to user i at context t.

Note that for the scale of websites (including our application of homepage widget relevance) which often have hundreds of millions or even billions of users, solving x_{ijt} for every user through this primal formulation is infeasible. Also, new users keep coming to the site, which requires us to solve x_{ijt} in real time with very low latency. Therefore, we follow [3] to use Lagrangian duality to solve for a small number of user-independent dual variables (one for each user-independent constraint). Given the solution to the dual, the primal variables x_{ijt} become independent, and it is computationally much cheaper to solve the optimal x_{ijt} using the dual variables. The mathematical details will be described in Section 3.

Due to the fact that the derivatives of Lagrangian vanish for linear programs [5], we add a quadratic term to the optimization function so that it becomes strongly convex:

$$\min_{x_{ijt}} \frac{\gamma}{2} \sum_{i,j,t} (x_{ijt} - q_{ijt})^2 - \sum_{i,j,t} x_{ijt}\hat{p}_{ijt}, \\ s.t. \sum_{i,t} x_{ijt} \geq I_j, \forall j, \quad x_{ijt} \geq 0, \forall i,j,t, \quad \sum_j x_{ijt} = 1, \forall i,t. \tag{8}$$

where q_{ijt} is a *pre-determined* serving distribution, and γ is a constant. Note that in this formulation we penalize the serving scheme for deviations from the pre-determined scheme $\boldsymbol{q}$, while the amount of penalization is controlled by γ, which can be made arbitrarily small to get closer to the original formulation. In this paper, we set q_{ijt} as a uniform (random) serving scheme, i.e. $q_{ijt} = 1/J$ for all i and t. The value of γ is set to be 0.1 such that this quadratic term does not have a big impact on the loss function while maintaining well-behaved strong convexity. We study the impact of γ in Section 4.5.

2.4 Adding Click Constraints

While the previous formulation achieves a desired impression distribution, the primary value for several widgets comes from the user interactions such as clicks. For example, if an item is an address-book import to send emails to invite friends to join the website, the email invitations are sent out after the user chooses to click on the import button. Hence, the critical business constraint is often on the clicks or actions on a particular widget, and not on impressions.

Therefore, if we would like to consider the tradeoff between user engagement such as clicks with the click-triggered downstream utilities, putting constraints on the number of clicks that each item receives can potentially provide better trade-off than using impression constraints.

Although directly replacing impression constraints with click constraints might sound reasonable, some widgets provide value through pure impressions as well. For instance, the "education" widget, which promotes a new product which might be unknown to many users, could achieve product awareness through just an impression. Therefore, we might have to use a mixture of impression and click constraints to satisfy the required business objectives. In our application, the impression constraints used in conjunction with click constraints were sometimes less tight than the standalone impression constraints since in the former case, there was also some support from the click constraint.

With the click constraints, the optimization problem now becomes

$$\min_{x_{ijt}} \frac{\gamma}{2} \sum_{i,j,t} (x_{ijt} - q_{ijt})^2 - \sum_{i,j,t} x_{ijt}\hat{p}_{ijt} \\ s.t. \sum_{i,t} x_{ijt} \geq I_j^*, \quad \sum_{i,t} x_{ijt}\hat{p}_{ijt} \geq C_j, \forall j, \\ x_{ijt} \geq 0, \forall i,j,t, \quad \sum_j x_{ijt} = 1, \forall i,t. \tag{9}$$

where I_j^* is the potentially relaxed version of the impression constraint for each widget j, and C_j is the constraint on the number of clicks.

We note that the formulations discussed in this section can be easily extended to many other use cases: For example, in some scenario we might want to put constraints on the aggregated number of impressions received by a group of widgets. Another use case is we can try to incorporate user's negative actions such as closing the widget, which also requires us to have a close action prediction model.

3. SCALABLE CONSTRAINED OPTIMIZATION FRAMEWORK

In this section, we describe our scalable constrained optimization framework to solve the various formulations introduced in Section 2. We also describe an example of the actual online serving infrastructure for the homepage widget relevance in Section 3.3.

3.1 Solving Global Relative Utilities

We first describe the approach to obtain the solution of the equations in Section 2.3.1. Let $\boldsymbol{w} = (w_1, \ldots, w_{J-1})$, $F_j(\boldsymbol{w}) = \sum_{i,t} x_{ijt} - I_j$, and $\boldsymbol{F}(\boldsymbol{w}) = \{F_j(\boldsymbol{w}) = 0, j = 1, \cdots J-1\}$. Equation (6) can be rewritten as a $J-1$ nonlinear system with linear constraints on $\boldsymbol{w}$, i.e.,

$$\boldsymbol{F}(w_1, \ldots, w_{J-1}) = \boldsymbol{0} \\ w_j \in [0,1], \forall j \in \{1, \ldots, J-1\} \quad \text{and} \quad \sum_{j=1}^{J-1} w_j \leq 1. \tag{10}$$

To solve this efficiently, we introduce the sum of squares merit function for $\boldsymbol{F}$, defined by

$$m(\boldsymbol{w}) = \frac{1}{2}||\boldsymbol{F}(\boldsymbol{w})||^2 = \frac{1}{2}\sum_{j=1}^{J-1} F_j^2(\boldsymbol{w}) \tag{11}$$

Any solution $\boldsymbol{w}^*$ of Equation (10) minimizes the merit function such that $m(\boldsymbol{w}^*) = 0$. Minimizing $m(\boldsymbol{w})$ with the linear constraints on $\boldsymbol{w}$ can be efficiently solved using Newton method with adaptive barrier algorithm to enforce constraints. Details can be found in [5].

For our experiments, we first obtain a vector $\bar{\boldsymbol{w}}$ by solving Equation (10) assuming $\hat{p}_{ijt} = \bar{p}_j$, with $\bar{p}_j$ being the average CTR per widget. It then becomes a linear system that is very easy to solve. Then, we initialize $\boldsymbol{w}$ by $\bar{\boldsymbol{w}}$, and solve the actual non-linear system in Equation (10) using Newton method with adaptive barrier algorithm. For our application it took about 30 minutes to obtain $\boldsymbol{w}$ that satisfies Equation (10) for half a million samples on a single machine.

3.2 Constrained optimization

We now consider solving the constrained optimization problem introduced in Section 2.3 and 2.4. Without loss of generality, we will focus on solving the constrained optimization in (9). The same approach can be applied to others in Section 2.3 and 2.4 trivially.

Finding a scalable solution to (9) is actually quite challenging due to two reasons: (a). The unknown primal variables x_{ijt} are the serving distributions for all the users at any contexts. For web applications with hundreds of millions of users, solving all x_{ijt} in a scalable way is non-trivial. (b). There are new users who continuously come into the system and do not exist in the training data. Hence, an approach to solve x_{ijt} for these users on the fly is required. To address these challenges, we refer to [3] for a practical and scalable solution.

3.2.1 Scalable Solution via Lagrangian Duality

The Lagrangian function of the primal problem in (9) is

$$\begin{aligned}\Lambda(\boldsymbol{x}, \boldsymbol{\mu}, \boldsymbol{\xi}, \boldsymbol{\nu}, \boldsymbol{\delta}) = {} & \frac{\gamma}{2}\sum_{i,j,t} ||x_{ijt} - q_{ijt}||^2 - \sum_{i,j,t} x_{ijt}\hat{p}_{ijt} \\ & - \sum_j \mu_j(\sum_{i,t} x_{ijt} - I_j^*) \\ & - \sum_j \xi_j(\sum_{i,t} x_{ijt}\hat{p}_{ijt} - C_j) \\ & - \sum_{i,j,t} \delta_{ijt}x_{ijt} - \sum_{i,t} \nu_{it}(\sum_j x_{ijt} - 1),\end{aligned}$$

where $\mu_j \geq 0$, $\xi_j \geq 0$, for all j, and $\delta_{i,j,t} \geq 0$ for all i, j and t. The dual variables $\boldsymbol{\mu}$, $\boldsymbol{\xi}$, $\boldsymbol{\nu}$, and $\boldsymbol{\delta}$ are introduced to make sure the constraints are satisfied. By letting $\frac{\partial \Lambda(\boldsymbol{x},\boldsymbol{\mu},\boldsymbol{\xi},\boldsymbol{\nu},\boldsymbol{\delta})}{\partial x_{i,j,t}} = 0$, we obtain

$$x_{ijt} = \frac{c_{ijt} + \nu_{it} + \delta_{ijt}}{\gamma}, \tag{12}$$

where

$$c_{ijt} = \gamma q_{ijt} + \mu_j + (1 + \xi_j)\hat{p}_{ijt}. \tag{13}$$

Hence, as long as all dual variables $\boldsymbol{\mu}$, $\boldsymbol{\xi}$, $\boldsymbol{\nu}$, and $\boldsymbol{\delta}$ are known, the serving plan x_{ijt} for any i, j, and t can be inferred using Equation (12). We also note that with q_{ijt} and $\hat{p}_{ijt}$ known, c_{ijt} only depends on μ_j and ξ_j, where $\boldsymbol{\mu}$ and $\boldsymbol{\xi}$ are J dimensional vectors, with J being the total number of items. However, ν_{it} and δ_{ijt} depend on user i and context t, and it can become quite high-dimensional when there are many users in the system. [3] provided an algorithm with theoretical proof that significantly simplifies the complexity, such that x_{ijt} can be directly obtained on the fly as long as $\boldsymbol{\mu}$ and $\boldsymbol{\xi}$ are known. Given the user i, context t, as well as the optimal dual plan $\boldsymbol{\mu}$ and $\boldsymbol{\xi}$, the scalable algorithm to obtain the optimal solution of x_{ijt} for all $j = 1, \cdots, J$ can be described in Algorithm 1, with the detailed proof in [3].

Algorithm 1 Fast fitting of primal given the dual

```
INPUT: User $i$, context $t$, a list of items $j = 1, \cdots, J$, dual variables $\boldsymbol{\mu}$ and $\boldsymbol{\xi}$, pre-determined serving plan $q_{ijt}$ for all $j$, and constant $\gamma$.
Obtain $\hat{p}_{ijt}$ for all the items $j = 1, \cdots, J$ using the CTR prediction model.
Let $c_{ijt} = \gamma q_{ijt} + \mu_j + (1 + \xi_j)\hat{p}_{ijt}$.
Order the items by $c_{ijt}$ such as $c_{i1t} \geq c_{i2t} \geq ... \geq c_{iJt}$.
Let $x_{ijt} = 0$ for all $j$ and $a = \gamma$.
for k=1 to J do
  if $c_{ikt} + (a - c_{ikt})/k \leq 0$, then
    $k = k - 1$
    break;
  else
    $a = a - c_{ikt}$
  end if
end for
Let $\nu_{it} = a/k$ and set $x_{ijt} = (c_{ijt} + \nu_{it})/\gamma$, for $j = 1, \cdots, k$.
```

3.2.2 Approximations for Solving the Dual

In this section, we describe our approach to obtain the optimal solutions to the J dimensional dual vectors $\boldsymbol{\mu}$ and $\boldsymbol{\xi}$ for the quadratic programming constrained optimization problems introduced in Section 2.3 and 2.4. In Section 3.2.1, we have already shown that as long as these two dual vectors are known, the primal x_{ijt} can be solved in a very scalable way using Algorithm 1.

Note that using the entire data set containing all users to solve the dual is still not scalable, and has the same complexity as solving all the primal variables. However, if J is small (e.g. 10 or 100), certain approximations can be made to solve the dual variables in a more scalable fashion. [3] suggested two approaches: (a). Use user features such as profile features or past interactions with the items to build clusters, and use a single x_{ljt} to represent all users' serving probabilities in cluster l for item j and context t. The dual can be solved with the cluster-wise primal variables, which is in a much smaller scale, and after the dual is obtained, we can still solve the personalized serving plans using Algorithm 1. (b). Randomly sample a small percentage of data, and solve the quadratic programing problem for the sampled data. After the dual variables are obtained, the serving plans for all the users can be easily solved. Comparing the two approaches, [3] shows that the random sampling empirically performs slightly better than the clustering approach.

In this paper, we adopt the random sampling approach, but apply a simple wrapper around it using Map-Reduce to achieve better performance. For a data set with N observations, we randomly sample K number of subsets, with each subset having M samples. The total number of primal variables x_{ijt} thus becomes $M * J$. For each subset k, we solve the quadratic programming to obtain optimal duals $\boldsymbol{\mu}^{(k)}$ and $\boldsymbol{\xi}^{(k)}$, and the final estimates of $\hat{\boldsymbol{\mu}}$ and $\hat{\boldsymbol{\xi}}$ are: $\hat{\boldsymbol{\mu}} = \sum_k \boldsymbol{\mu}^{(k)}/K$, and $\hat{\boldsymbol{\xi}} = \sum_k \boldsymbol{\xi}^{(k)}/K$. For our experiments where $J = 5$, we use $K = 100$ and $M = 300$ so that each quadratic programming solver can finish within 10-15 minutes. We have observed that this parameter setting works pretty well in general in terms of satisfying constraints except for a few cases. Details are described in Section 4.

It is worthwhile to point out that the random sampling needs to be done at the observation level instead of the user level, since some users can come to the site much more frequently than other users.

We also note that it is actually possible to apply recent developments of large-scale convex optimization technologies such as Alternating Direction Method of Multipliers (ADMM) [4] to solve the dual using all the data, which is one of our future work. The scalable computation of primal serving plans given the dual is still needed for the new users who come to the system continuously.

3.3 Online Serving Infrastructure

In this section, we describe an example of the online serving infrastructure for the problems described in Section 2. For simplicity we assume there is only a single slot to show the recommended items on the web page. The system consists of three key components:

- **CTR prediction model training**: An offline logistic regression model training process that regularly kicks off (e.g. weekly). Since the data can include hundreds of millions of samples, we adopt the large-scale logistic regression algorithms such as ADMM [4] and use Map-Reduce for divide-and-conquer and obtain the consensus across all the partitions.

- **Optimization solver**: Given the predicted probabilities of clicks for all the users, items and contexts, we either solve the global downstream utilities using the approach described in Section 3.1, or solve the dual variables from the constrained optimization using the Map-Reduce approach described in Section 3.2.2.

- **Online scoring**. When a user i comes to the website at a certain context t, a request is generated to the backend server to fetch the recommended item to show to the user. The system computes on the fly the predicted CTR for each eligible item based on the CTR prediction model, and also the serving plan using either Equation (5) for global relative utilities, or Algorithm 1 given the dual variables for constrained optimization problems in Section 2.3 and 2.4. After the serving probabilities x_{ijt} for all $j \in J$ are obtained, we draw a random sample from the multinomial distribution of the serving plan and serve the sampled item to the user.

4. EXPERIMENTS

In this section we show the offline and online experimental results with data collected from a major professional social network website. We first introduce our application and experimental environment in Section 4.1, and describe our data for model training and solving dual variables for constrained optimization in Section 4.2. In Section 4.3 we show the offline experimental results of the per-item CTR prediction models, and constrained optimization performance through replay. The online experiments of the constrained optimization approaches in terms of violation of constraints as well as the CTR performance are shown in Section 4.4. Finally in Section 4.5 we show some insights of the constrained optimization, by visualizing the effects for various values of γ and different types of constraints.

4.1 Homepage Widget Module

As mentioned in Section 2, the widgets module runs on a major professional social network with hundreds of million users worldwide, and it occupies the most prominent slot on the desktop homepage. The widget slot has tens of millions impressions on a weekly basis, and the widgets shown therein has a big impact and drives a lot of downstream user actions and page views. There are currently around 10 different widget candidates to show to the user — for the complete list, check Section 2.

For business privacy concerns, we shuffle and anonymize the widget names while reporting experimental results. For different segments of users, the candidate pool of widgets are different, i.e., not every widget is eligible for each user. For active users, there are 5 widgets (denoted as W_1 to W_5) in the candidate pool. The active users are defined to be the ones who have visited the website more than once in the last 30 days. For the other users who are either dormant or new to the site, there are 3 more on-boarding widgets that are eligible; the number of eligible widgets hence becomes 8 (denoted as W_1 to W_8).

Before we introduced relevance models into the homepage widget module, it was running a set of hand-tuned rules to determine which widget to show to each user. The rules were based on a non-personalized priority ordering among the widgets. For example, since it is believed that address book imports have highest business values among all the widgets, it was often picked as the first choice to show to users, regardless of their profile features, or whether they have interacted with this widget in the past. The widget serving system also has two important behaviors as listed below:

- **A cool-off system**. In order to avoid user fatigue, each widget has an impression cap per user in a certain time period. After the impression cap is reached for a particular user, the widget is made ineligible ("cooled off") for that user within the time period. Such a rule ensures that we do not keep showing the same widget to a user, regardless of how many times she has refreshed the page. The widget slot itself also had a similar cool-off (larger impression cap than the ones for individual widgets) to avoid over-exposing a user to the entire widget inventory. In general, the cool-off system provides a better user experience and allows users to have a more diversified widget exposure.

- **Handling null content**. We note that the widget serving system only controls which widget to serve to the user; it does not control what content to serve within each widget, if a widget is chosen. Each widget belongs to an independent service that determines the user interface and content to serve to the user. For some widgets, the requests to retrieve content are sometimes unsuccessful, which can be either due to latency timeout of the concerned service, or the fact that there is no appropriate content to show to this user for that widget type. In this case, even though the widget serving system may decide to serve this widget, there is nothing to show. Hence the system automatically falls back to not showing any widget for this user.

Our online serving system with the relevance model and constrained optimization is described in Section 3.3. It simply replaces the original hand-tuned rules in the system for the widget serving, but obeys the impression and click constraints obtained from the original system's allocation plans. It also inherits the cool-off system and the approach to handle null content. Note that these two system behaviors are not considered in our constrained optimization formulation, and hence has resulted in some violations of constraints in our online experiments. Embedding such practical system designs into our constrained optimization framework is quite challenging, but is an interesting future work.

Across our experiments we mainly consider the following approaches:

- **Control**: The original system which serves widgets using a set of manually-tuned business rules. It serves as the baseline of our online experiments, and also defines the desired impression and click constraints for our constrained optimization approaches.

- **UW-IMP**: The approach to learn global downstream utility weights given the impression constraints. The formulation is described in Section 2.3.1.

- **QP-IMP**: The quadratic programming formulation with impression constraints only. See Section 2.3.2 for details.

W_1	W_2	W_3	W_4	W_5
0.693	0.703	0.706	0.798	0.721

Table 1: AUC of the CTR prediction model for various widgets.

- **QP-CLK**: The quadratic programming formulation with impression constraints (can be relaxed) and click constraints. See Section 2.4 for details.

4.2 Our Data

For the purpose of training our CTR prediction model and solving the constrained optimization problems, we collected widget impressions and clicks data during the first 2 weeks of February, 2014. For evaluating the performance of our CTR prediction model as well as offline replay to make sure the constraints can be satisfied, we used the third week of February as the test period.

For training the CTR prediction model, directly using the data collected from the original system with hand-tuned rules would introduce significant serving bias. Therefore, for a small percentage of traffic, we apply the random serving scheme that randomly picks a widget from the candidate set for each user visit, while still respecting the cool-off rules. The data collected from such serving scheme is much more unbiased than the data collected from the baseline system. Our final model training and test data both contain a few million samples.

The features we used for the CTR prediction model include context features such as the time of day and day of week, user's past interactions with each widget for the last 30 days (e.g. impressions, clicks, and closes), the user's activities across the website (e.g. past visits and interactions on different modules and channels), and the user profile features such as age, gender, industry, skills, and so forth. Since this application only has a few items, each item j has a significant amount of data to learn the per-item coefficients $\boldsymbol{\delta}_j$ – hence no item-specific features are needed.

For solving the quadratic programming problems for the constrained optimization, we used a random sample of the data generated from the hand-tuned baseline serving model. The underlying reason that we did not use data collected from the random serving scheme for this purpose is because we have observed that the user behavior such as number of page views in random bucket is very different from that in the actual serving bucket. Hence solving the quadratic programming using the user visit information from the baseline model provides better precision in terms of carving out personalized serving plans for each user while satisfying global impression and click constraints.

4.3 Offline Experiments

We first describe the performance of the CTR prediction model on a per-widget basis in Section 4.3.1, and evaluate the stability of our proposed Map-Reduce-based dual solver in Section 4.3.2. We show the satisfaction of the impression constraints for all the constrained optimization approaches in Section 4.3.3. For all offline experiments we limit our scope to the active user segment where 5 candidate widgets W_1 to W_5 are considered.

4.3.1 CTR Prediction Model Performance

We evaluate the performance of the CTR prediction model using the data generated from the random serving scheme during the test period. The test AUC of for all the 5 widgets can be seen in Table 1). In general the performances across all the widgets are reasonable, although there are some differences of the AUC for different widgets, e.g. the AUC for W_1 (0.693) is significantly worse than that for W_4 (0.798). Since the CTR prediction is not the focus of this paper, and the performance can often be improved by adding more useful features, we do not discuss this further here.

4.3.2 Evaluation of Solving Dual

In this subsection, we evaluate the stability of our proposed Map-Reduce based algorithm for solving the dual described in Section 3.2.2. The objective here is to make sure that using $M = 300$ is good enough, where $5 \times M$ indicates the number of primal variables x_{ijt} to solve along with the dual. We note that the amount of time to solve the quadratic programming increases exponentially as the size of M grows, for example, for solving QP-IMP and $K = 100$, when $M = 300$ it usually takes 10-15 minutes to solve the dual, while for $M = 500$ it can take up to 50 minutes. Hence it is important to strike the right balance between the accuracy and the computational efficiency.

We evaluate the choice of M using the following approach given $K = 100$ for QP-IMP: For each value of M (up to 500), we run the algorithm described in Section 3.2.2 for 100 times, and observe the standard deviation for the dual variable vector $\boldsymbol{\mu}$. The result is shown in Figure 3. It is clearly seen that the duals start to converge after $M = 200$. We also observe that the differences between the duals obtained from $M = 300$ and $M = 500$ are extremely small. Hence, $M = 300$ is a good choice for this application.

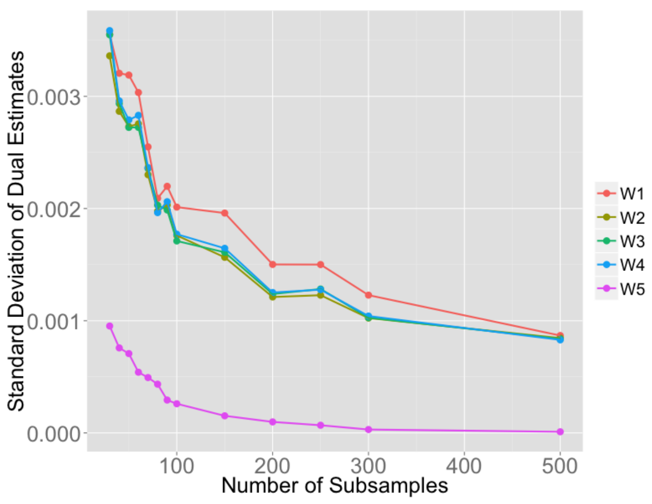


Figure 3: QP-IMP: The standard deviation of dual vector $\boldsymbol{\mu}$ by running the dual solving algorithm 100 times, for different values of M and $K = 100$.

4.3.3 Offline Replay of Impression Constraints

Given the CTR prediction model and the dual variables, we apply Algorithm 1 to obtain the serving plan for each user in test data, which contains all the users who saw any widget during the test period. The serving probabilities x_{ijt} for each user i, widget j and context t are then aggregated to the global expected number of impressions per widget, and we test if they match the desired impression allocation, which is simply $I_j / \sum_j I_j$ for each item j.

In Table 2 we show the replay performance. Both UW-IMP and QP-IMP were able to precisely match the desired allocation of impressions. In QP-CLK, click constraints were added to ensure that each widget received enough number of clicks – the impression constraints for high-engagement widgets (W_1 and W_5) were also discarded, since our business partners felt that it was not necessary as long as those click constraints were satisfied. From the table it is clear that QP-CLK was also able to satisfy all the impression constraints, and it was interesting to observe the effect of adding click constraints was to move some traffic from the high-CTR widget W_1 to the low-CTR widget W_4, so that W_4 can receive more clicks. We will provide more insights on why this is happening in Section 4.4 and 4.5

	Widget impression %			
Widget	Desired	UW-IMP	QP-IMP	QP-CLK
W_1	20%(-)	20%	20%	16%
W_2	8%	8%	8%	8%
W_3	7%	7%	7%	7%
W_4	52%	52%	52%	55%
W_5	13%(-)	13%	13%	14%

Table 2: Impression distribution achieved by UW-IMP, QP-IMP and QP-CLK in offline replay. In Desired column, 20%(-) means the desired impression% for W_1 is 20%, but for QP-CLK, it is removed. Same for 13%(-) for W_5.

4.4 Online Experiments

We have run two online A/B test experiments on different segments of users during different time periods. The first experiment was run from Mar 2, 2014 to Mar 11, 2014, with Control running with the majority of the traffic, and UW-IMP, QP-IMP and QP-CLK being the treatments. This experiment was targeted upon the active user segment, hence for each user there are 5 widget candidates (denoted as W_1 to W_5). The results are discussed in Section 4.4.1.

In the second experiment that was run in May, 2014, we had to carve out a constrained optimization solution for a group of 3 onboarding widgets specifically designed for relatively inactive users (i.e., users who visited the website once or less in the last 30 days). The business requirement was to guarantee a certain number of impressions and clicks for the entire group of widgets (denoted as W_6, W_7 and W_8), without any particular specification of the internal distribution. The original set of widgets (W_1 to W_5) for active users were also eligible for display, but did not have any constraints on them since they were not designed for such inactive users. We show the experimental results in Section 4.4.2.

4.4.1 Experiments on Active User Segments

In Table 3, we show the actual impression distribution achieved by Control, UW-IMP, QP-IMP and QP-CLK in online A/B tests for the active user segment. And in Table 4 we show the actual allocation of clicks for all these approaches. We also show the lift percentage in terms of total number of clicks received for each widget, and the CTR of each widget. We note that our ultimate goal is to maximize user engagement (i.e. CTR) while making sure the total number of clicks received by each widget can satisfy the constraints as much as possible. By reading the results carefully, we observe the following:

- **Overall Engagement**. QP-CLK performed the best in terms of the CTR (51.3% lift over the baseline), and it was able to achieve very big CTR lifts across the widgets with high percentages of impressions and clicks (W_1, W_4 and W_5). QP-IMP not only performed worse than QP-CLK in terms of overall CTR, it also dropped the CTR of W_4 by 13%. Considering W_4 is an important widget (desired impression percentage 52%), this is in fact a major problem for QP-IMP. Although UW-IMP gave more uniform CTR lifts across all widgets, it performed 15% worse than QP-CLK in terms of global CTR.

- **Impression constraints**. We note that UW-IMP performed better than QP-IMP in terms of satisfying the impression constraints. This is a bit surprising to us, and it happened mainly due to two reasons: (a) Our Map-Reduce solution for the dual is only an approximation, while the solution for UW-IMP is accurate. (b). The cool-off system has introduced a fair amount of noise that the offline quadratic programming solver has not considered.

- **Click constraints**. The click constraints for QP-CLK had more-or-less been satisfied, except for W_2 (-10%) and W_3 (-61%). Also note that the impression constraints of these two widgets had never been met for all three approaches: UW-IMP, QP-IMP and QP-CLK. This was actually due to the fact that these two widgets often return null results in an unpredictable fashion. Since our offline quadratic programming did not consider such scenarios, the constraints for these two widgets often get violated in practice.

	Widget impression %			
Widget	Desired	UW-IMP	QP-IMP	QP-CLK
W_1	20%(-)	20.2%	12.8%	13.5%
W_2	8%	5.1%	5.5%	6.1%
W_3	7%	3.1%	2.6%	2.4%
W_4	52%	57.4%	69.2%	63.6%
W_5	13%(-)	14.2%	9.9%	14.4%

Table 3: Impression distribution achieved by UW-IMP, QP-IMP and QP-CLK in online experiments. In Desired column, 20%(-) means the desired impression% for W_1 is 20%, but for QP-CLK, it is removed. Same for 13%(-) for W_5.

One thing that we noticed after launching the QP-IMP experiment online, was that the low-CTR widget W_4 was having significantly lower CTR than Control (-13%), while being allocated comparable number of impressions. We hypothesized that QP-IMP was mainly showing W_4 to less engaged users, while for highly-engaged users it showed more W_5 (a high-CTR widget) instead of W_4. By doing this, QP-IMP was able to maximize overall engagement while satisfying the impression constraints. To validate our hypothesis, we segmented our users based on overall widget CTR for the months of January and February in 2014, and observed the impression ratio of W_5 to W_4 for each segment under various approaches.

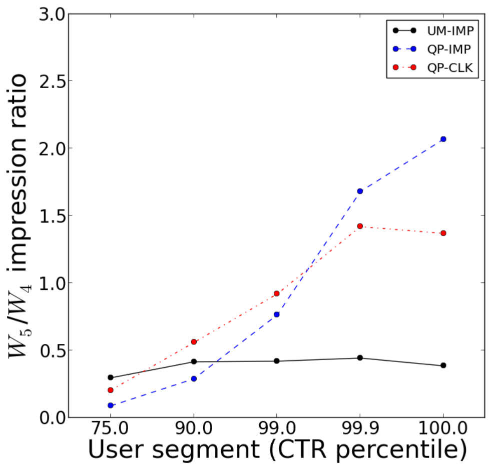


Figure 4: Impression ratio of W_5 to W_4 for different user segments, for the various models.

Our findings (as shown in Figure 4) validated our hypothesis. UM-IMP showed the same impression ratio across all segments, thus completely missing out on the opportunity to increase engagement. QP-IMP exploited the opportunity to the fullest, and tried to serve much less W_4 for the highly engaged users. QP-CLK, with the addition of the appropriate click constraint, striked the right balance between maximizing engagement and ensuring enough engaged users see W_4 for satisfying the click constraints.

4.4.2 Group Constraints

In a separate experiment targeting the new and dormant users, we needed to design a constrained optimization based on the busi-

	Widget click distribution %									
	Click %				Click lift %			CTR lift %		
Widget	Control	UW-IMP	QP-IMP	QP-CLK	UW-IMP	QP-IMP	QP-CLK	UW-IMP	QP-IMP	QP-CLK
Overall	-	-	-	-	46.3%	54.8%	**63.4%**	36.6%	43.8%	**51.3%**
W_1	49.4%	38.1%	38.1%	29.2%	13.0%	19.0%	**-3.0%**	-8.0%	52.0%	**16.0%**
W_2	2.7%	2.4%	2.0%	1.5%	28.0%	17.0%	-10.0%	35.0%	16.0%	-20.0%
W_3	0.8%	0.5%	0.2%	0.2%	-13.0%	-59.0%	-61.0%	41.0%	-20.0%	-20.0%
W_4	27.3%	21.1%	18.2%	22.5%	13.0%	3.0%	**35.0%**	16.0%	-13.0%	**24.0%**
W_5	19.8%	37.9%	41.5%	46.6%	181.0%	225.0%	**285.0%**	50.0%	148.0%	**102.0%**

Table 4: The click allocation distribution of Control, UW-IMP, QP-IMP, and QP-CLK in the online A/B test for the active user segment. It also shows the lift percentage in terms of number of clicks and CTR for each approach versus control.

	Widget impression %		
Widget	Desired	Control	QP-GRP-IMP
W_1	-	1.3%	4.6%
W_2	-	0.1%	0.2%
W_3	-	1.5%	0.2%
W_4	-	4.3%	1.3%
W_5	-	0.2%	3.2%
$W_6 - W_8$	90.0%	88.9%	89.8%
W_6	-	66.3%	55.4%
W_7	-	14.5%	21.2%
W_8	-	8.1%	13.2%

Table 5: Impression distribution achieved by QP-GRP-IMP in online experiments.

		Widget click distribution %		
Widget	Old %	New %	Click lift %	CTR lift %
Overall	-	-	29.1%	38.3%
$W_1 - W_5$	10.9%	15.3%	82.3%	51.8%
$W_6 - W_8$	89.1%	84.7%	44.0%	31.8%
W_6	73.2%	66.2%	18.6%	51.5%
W_7	9.6%	9.8%	33.2%	-2.3%
W_8	6.3%	8.7%	82.8%	14.3%

Table 6: Click distribution achieved by QP-GRP-IMP in online experiments. It also shows the lift in terms of total number of clicks and CTR for each widget.

ness requirement to guarantee a certain number of impressions and clicks for a group of 3 specifically-designed on-boarding widgets (W_6 to W_8). The original 5 widgets for the active users were still eligible, but did not have any constraints on them since they were not designed for such segment of users. For this experiment, we simply extended our constrained optimization framework to handle group impression and group click constraints (and a mix of them as well). The exact formulation we used can be mathematically expressed as:

$$\begin{aligned} &\min_{x_{ijt}} \frac{\gamma}{2} \sum_{i,j,t} ||x_{ijt} - q_{ijt}||^2 - \sum_{i,j,t} x_{ijt}\hat{p}_{ijt}, \\ &s.t. \sum_{i,t,j \in G} x_{ijt} \geq I_G, \forall G, \quad x_{ijt} \geq 0, \forall i,j,t, \\ &\sum_{j} x_{ijt} = 1, \forall i,t. \end{aligned} \tag{14}$$

where G denotes the group of widgets $\{W_6, W_7, W_8\}$. We denote this approach as QP-GRP-IMP. The only constraint in this experiment was that the widget group G received at least 90% of the overall widget impressions for this user segment. Table 5 shows that QP-GRP-IMP was able to satisfy this constraint well, and Table 6 shows that QP-GRP-IMP had generated almost uniformly significant lifts for both the number of clicks and CTR across all widget groups.

Comparing this experiment with the active user one in Section 4.4.1, we observed that this experiment performed much better in terms of both satisfying the constraints and matching the online results with offline replay. We believe it is because inactive users visited the site much less frequently, and hence the widget cool-off rules were rarely activated. As a result, the online serving condition became very similar to our quadratic programming setup.

4.5 Analysis of Decision Boundaries

While the method to convert the predicted click probabilities into a serving probability distribution is dependent on the optimal dual weights, it is interesting to visualize this transformation, as we vary different parameters in the system. This also helps us better understand what type of users are being served particular widgets, and how much diversity exists in our serving scheme.

Since visualizing a multi-dimensional serving distribution is difficult, we simplify the problem to a 2-widget system that only includes the low-CTR widget W_4 and the high CTR widget W_5, and show the effects with variations of constraints and parameters.

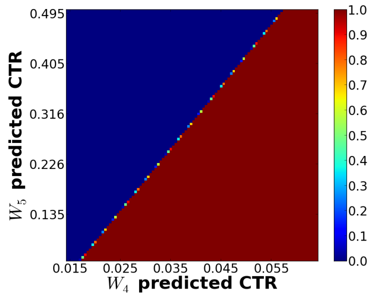
Figure 5 shows the probability of serving W_4 for different values of the predicted CTR pair <$\hat{p}_{W_4}, \hat{p}_{W_5}$>. In the top row, we show for QP-CLK how the serving probability varies as we change γ. As this parameter shrinks, the serving distribution becomes more deterministic, as seen in Figure 5a (γ = 0.001). For larger values of γ, we are able to obtain more randomized serving schemes with increased user-level widget diversity.

We also try to understand how the dual to primal transformation changes, as we exclude some of the constraints from QP-CLK with γ = 0.1. The decision boundaries of original QP-CLK with both impression and click constraints can be seen in Figure 5b. The unconstrained system (Figure 5f) rarely shows W_4, since the CTR of W_5 is much greater than W_4. The impression constraint only provides a small boost to the serving probability of W_4, while applying the click constraint provides a much larger boost (Figure 5e and 5d respectively).

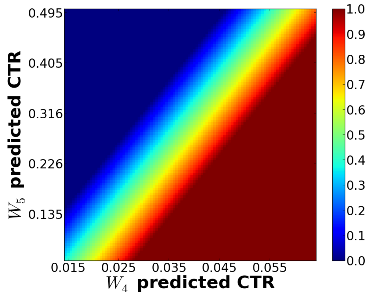
5. RELATED WORK

There is rich literature on methods to recommend content to users on web portals to maximize a one dimensional short-term metric like CTR [1, 7, 9, 10]. But little work have been done to extend the setting to multi-objective optimization except for [3]. Our current work uses the theoretical work proposed in that paper. However, while [3] provides a rigorous empirical analysis based on offline analysis on data in the context of Yahoo! front page, it did not report on online experiments run on live traffic. In this paper, we provide extensive analysis (both offline and online) on a new application for a social network site. To the best of our knowledge, this is the first such case study that presents results obtained by deploying the multi-objective optimization framework on a large real application. We believe this would encourage many more applications to adopt such a methodology in the future.

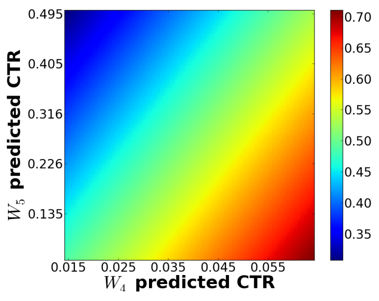
We also note that our framework is different from the computational advertising problem in [6] where an advertiser specifies a



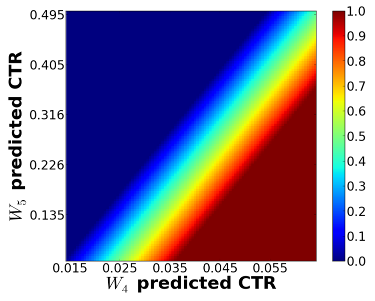
(a) QP-CLK, $\gamma = 0.001$



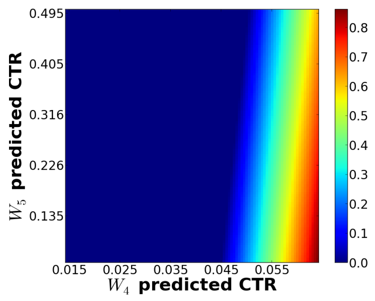
(b) QP-CLK, $\gamma = 0.1$



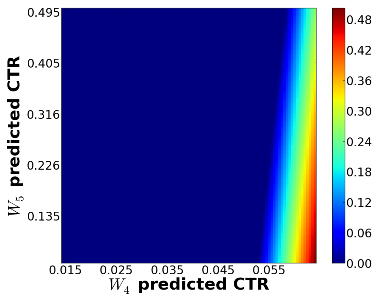
(c) QP-CLK, $\gamma = 1.0$



(d) Click constraint, $\gamma = 0.1$



(e) Impression constraint, $\gamma = 0.1$



(f) No constraint, $\gamma = 0.1$

Figure 5: Decision boundaries

bid, which is the amount she is willing to pay when a user performs a positive action on the ad. Since the bids are all in the same currency, it is easy to derive the value of an action on ads and rank. In our scenario, we do not have a bid on various actions. The optimization framework used can be thought of as a way to elicit such utilities through a series of overall constraints obtained through various business considerations.

6. CONCLUSION

In this paper we proposed a scalable constrained optimization framework to optimize user engagement under certain business constraints, for the specific application of showing promotional widgets to web users on the homepage of a major professional social network site. Several different types of constraints according to the business needs are considered and tested in both offline experiments and online A/B tests. The proposed system has significantly improved over the original baseline with hand-tuned rules, and hence has been launched to serve full traffic in production for the web site. Our framework has provided great flexibility in specifying the business needs that the widget module has to deliver. The ease with which the ever-changing business needs can be expressed, has resulted in timely boost to growth, content and other facets of the professional network.

For future work, we plan to solve for the dual with a much larger data sample for the quadratic program, using large-scale convex optimization algorithms like Alternating Direction Method of Multipliers (ADMM) [4]. We are also incorporating some aspects of the online system in our offline relevance models — these include the cool-off behavior and data availability issues. Finally, the framework can also be used to optimize for other user experience metrics more explicitly — e.g., widget diversity, close rates, downstream actions.

7. REFERENCES

[1] D. Agarwal and B.-C. Chen. Regression-based latent factor models. In *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 19–28. ACM, 2009.

[2] D. Agarwal, B.-C. Chen, and P. Elango. Spatio-temporal models for estimating click-through rate. In *Proceedings of the 18th international conference on World wide web*, pages 21–30. ACM, 2009.

[3] D. Agarwal, B.-C. Chen, P. Elango, and X. Wang. Personalized click shaping through lagrangian duality for online recommendation. In *Proceedings of the 35th international ACM SIGIR conference on Research and development in information retrieval*, pages 485–494. ACM, 2012.

[4] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends® in Machine Learning*, 3(1):1–122, 2011.

[5] S. P. Boyd and L. Vandenberghe. *Convex optimization*. Cambridge university press, 2004.

[6] A. Z. Broder. Computational advertising. In *SODA*, volume 8, pages 992–992, 2008.

[7] A. S. Das, M. Datar, A. Garg, and S. Rajaram. Google news personalization: scalable online collaborative filtering. In *Proceedings of the 16th international conference on World Wide Web*, pages 271–280. ACM, 2007.

[8] T. Jaakkola and M. Jordan. A variational approach to bayesian logistic regression models and their extensions. In *Sixth International Workshop on Artificial Intelligence and Statistics*. Citeseer, 1997.

[9] J. Liu, P. Dolan, and E. R. Pedersen. Personalized news recommendation based on click behavior. In *Proceedings of the 15th international conference on Intelligent user interfaces*, pages 31–40. ACM, 2010.

[10] D. H. Stern, R. Herbrich, and T. Graepel. Matchbox: large scale online bayesian recommendations. In *Proceedings of the 18th international conference on World wide web*, pages 111–120. ACM, 2009.