From Prediction to Action: A Closed-Loop Approach for Data-Guided Network Resource Allocation

Yanan Bao, Huasen Wu, Xin Liu
Dept. of Computer Science, University of California, Davis, CA 95616, USA
{ynbao, hswu, xinliu}@ucdavis.edu

ABSTRACT

Machine learning methods have been widely used in modeling and predicting network user experience. In this paper, moving beyond user experience prediction, we propose a closed-loop approach that uses data-generated prediction models to explicitly guide resource allocation for user experience improvement. The closed-loop approach leverages and verifies the causal relation that often exists between certain feature values (e.g., bandwidth) and user experience in computer networks. The approach consists of three components: we train a neural network classifier to predict user experience, utilize the trained neural network classifier as the objective function to allocate network resource, and then evaluate user experience with allocated resource to (in)validate and adjust the original model. Specifically, we propose a dual decomposition algorithm to solve the neural network-based resource optimization problem, which is complex and non-convex. We further develop an iterative mechanism for classifier optimization. Numerical results show that the dual algorithm reduces the expected number of unsatisfied users by up to 2x compared with the baseline, and the optimized classifier further improves the performance by 50%.

Keywords

Classification; Resource Optimization; Computer Networks

1. INTRODUCTION

Based on network measurement and user behavior data, much recent research focuses on modeling and predicting user experience in computer networks using machine learning techniques, e.g., [1, 16, 6, 2, 10]. However, while it provides important insights, user experience prediction itself is usually not the ultimate goal in networks. Ideally, a network could identify users with poor experience and take proper actions proactively to improve their experience (e.g., by allocating additional bandwidth to selected users). Thus, we are facing a natural problem: given limited resources, how should we allocate them to multiple users to optimize the overall experience?

To answer this question, we advocate a closed-loop approach that uses data-generated prediction models to explicitly guide resource allocation for user experience optimization. This approach is illustrated in Fig. 1.

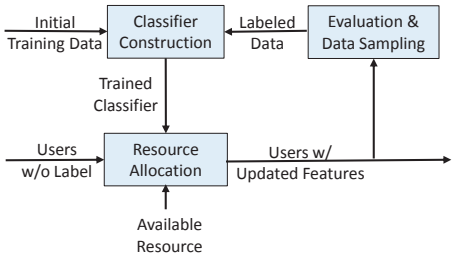


Figure 1: A closed-loop framework in data-driven resource allocation.

First, we start with a historical dataset with labeled user experience and the corresponding feature values (including network performance metrics). Based on the data, we construct an appropriate user experience prediction model to reflect the correlation between feature values and user experience. Then, we feed the constructed prediction model into the resource allocation component as the objective function to optimize resource allocation for users based on their real time feature values (network metrics). The output is an appropriate resource allocation and users with improved feature values. Last, the evaluation and data sampling component samples data after resource allocation, validates or invalidates the model, and adjusts the constructed prediction model as needed.

In this framework, we leverage existing machine learning methods for user experience prediction. Specifically, in this paper, we use the neural network prediction model. We focus on the resource allocation algorithms for the trained model (Sec. 4) and how to adjust the classification model based on evaluating resource allocation results to further improve performance (Sec. 5).

The proposed framework has two benefits. First, the constructed classifier illustrates a quantitative relationship between the feature values and the user experience. Using such a quantitative relationship and domain knowledge, we are able to allocate network resources more precisely to reduce the expected number of users with poor experience, in contrast to the typical approach of using abstract utili-

KDD '16, August 13-17, 2016, San Francisco, CA, USA

DOI: http://dx.doi.org/10.1145/2939672.2939871

ty functions for resource allocation. Second, the framework includes an evaluation component, where users are sampled after resource allocation to validate or invalidate the causal relationship hypothesis between the feature values and the user experience. This step also provides further opportunities to adjust the constructed prediction model, which is shown to be highly beneficial.

The proposed framework has several challenges. First, it is typically more challenging to optimize resource allocation based on prediction models derived from real data than to use utility functions with nice properties such as convexity [18, 8]. In our work, the neural network model generates a complicated function, requiring us to solve a non-convex and complex optimization problem. Second, choosing the best classifier is difficult for the following reasons: 1) The objective of the classifier is not to best classify all data samples, but to best guide resource allocation to improve user experience with respect to ground truth; and 2) resource allocation inevitably modifies the distribution of the users, making the best classifier a moving target.

In this paper, we present a holistic solution using the closed-loop framework in data-driven resource allocation. Specifically, we make the following contributions:

- We propose using the closed-loop framework in data-driven resource allocation. Inspired by the data-driven framework introduced in [3], our framework incorporates the critical component of evaluation and feedback, which allows model validation and adjustment.
- We use a dual decomposition algorithm to solve the neural-network-based optimization problem, which is complex and non-convex. The algorithm applies to scenarios with multiple types of resources, and enables resource allocation that is both continuous and discrete (Sec. 4).
- We formulate the classifier selection problem as a global functional optimization problem, which is highly complex. We develop a set of principles and an iterative mechanism to systematically select better classifiers. (Sec. 5).
- We evaluate the proposed solution using both synthetic and real world datasets (Sec. 6). Results indicate our solution can reduce the expected number of unsatisfied users by up to 2x compared to the baseline, while the classifier optimization further improves the performance by 50%.

2. RELATED WORK

The framework of data-guided resource allocation is first proposed in our previous work [3] that includes the classifier construction and resource allocation components. In [3], we utilize a logistic regression classifier learned on labeled data to guide resource allocation on unlabeled data. In comparison, in this paper, we consider the neural network model, which is more general, yet more challenging when used as the objective function for resource allocation. More importantly, we study the closed-loop approach that allows us to evaluate the impact of resource allocation and to select a better classifier, which are critical, yet not considered earlier.

User experience has been studied extensively recent years [1, 16, 6, 2, 10]. In [1], the authors use a month-long anonymous data collected from a cellular network provider to study Quality of Experience (QoE) metrics including session length, abandonment rate, and partial download ratio. The relation between mobile video streaming performance and user engagement from the perspective of network operators is discussed in [16]. Using 27 TB video streaming traffic from more than 37 million flows, the authors observe strong correlations between many network features and abandonment or skip rates.

A large body of literature considers learning-based cost-efficient decision making in other applications. For example, [9] discusses a high-level pipeline of data collection, predictive model, and decision analysis. Specific combinations of prediction and actions have been proposed in many areas, such as clinical treatment [4] and route planning [20, 19, 14]. Certain preliminary studies have combined learning and resource allocation in data center networks [11, 5].

Network Utility Maximization (NUM) has been extensively studied, e.g., in [12, 7, 8, 21, 22, 23]. The difference between our work and their work lies in three aspects: 1) Our utility function is learned from real datasets, and thus is more complicated; 2) Our problem includes multiple types of resources, which makes the problem more challenging; 3) We consider a closed-loop approach that optimizes the utility function based on the feedback from the real system.

3. SYSTEM MODEL

In this section, we first present the notation used in this paper, including the neural network classifier. Furthermore, we discuss the research questions and introduce a baseline algorithm for resource allocation.

3.1 Features and Resources

Consider users in a D-dimensional feature space, i.e., for user i, we have

$$\boldsymbol{x_i} = [1, x_{i,1}, x_{i,2}, ..., x_{i,D}]^T, \tag{1}$$

where $x_{i,d}$ is the value of feature d ($d = 1, 2, ..., D$), and the constant feature with value 1 is included for the ease of notation. Each user is associated with a label y_i, which is either **positive** (a user with poor experience) or **negative**.

There are K types of resources, and the resources allocated to user i are denoted by

$$\boldsymbol{r_i} = [r_{i,1}, r_{i,2}, ..., r_{i,K}]^T. \tag{2}$$

We assume that the relationship between the allocated resources and the change of feature values is captured by a function $\boldsymbol{g}(\cdot)$. Given $\boldsymbol{r_i}$ amount of resources, user i has its feature vector updated to

$$\boldsymbol{x_i}' = \boldsymbol{g}(\boldsymbol{x_i}, \boldsymbol{r_i}). \tag{3}$$

Domain knowledge plays a significant role here in deciding the function $\boldsymbol{g}(\cdot)$. In computer networks, such domain knowledge typically exists, e.g., how bandwidth or transmission power allocation affects users' throughput. Such domain knowledge is widely used in current network resource allocation schemes.

Clearly, not all feature values can be altered through resource allocation. For example, the device type is a static feature that does not change regardless of network resource

allocation. We focus on the subset of features, named controllable features, whose values can be altered by resource allocation. The term "resource allocation" refers to general actions that a network can take (potentially in cooperation with users/content providers). Examples include bandwidth allocation, frequency allocation, transmission power allocation, device reconnection, increasing buffering (e.g., for video streaming), content prefetching or caching, selecting multiple networks/interfaces, processing power allocation (e.g., in a cloudlet setting), etc. Note that the resources could be new types of resources that are not used yet by the system, or additional resources. For instance, in wireless communications users have already been allocated with bandwidth. Additional bandwidth could be reserved to serve the potentially unsatisfied customers.

We note that it is highly likely that network resource allocation could only improve user experience of certain users, but not all. The problem formulation here applies to the case where resources are scarce. When the resources are unlimited, then likely the user experience is constrained by other factors, and could not be solved by network resource allocation.

3.2 Neural Network Classifier

In the classifier construction component, the network starts with a set of samples with user experience labels. The objective is to model the relationship between feature values and the corresponding labels. Different classifiers can be adopted in the proposed framework, such as logistic regression [3], random forests and neural networks, depending on the application scenarios.

We consider a neural network with three layers: input layer, hidden layer and output layer. A three-layer neural network can be realized by a two-layer logistic regression.

Let $\boldsymbol{w} = [w_0, w_1, w_2, ..., w_D]^T$ be the weights of a logistic regression model. Then, the likelihood of instance $\boldsymbol{x}$ being positive is

$$\eta(\boldsymbol{w}, \boldsymbol{x}) = \frac{1}{1+\exp(-\boldsymbol{w}^T\boldsymbol{x})}. \quad (4)$$

Let H be the number of hidden neural nodes in the hidden layer, and $\boldsymbol{w}_1$, $\boldsymbol{w}_2$,..., $\boldsymbol{w}_H$ be the corresponding weights from input features to the hidden nodes. The weight vector from the hidden nodes to the output node is denoted by $\boldsymbol{w_0} = [w_{0,0}, w_{0,1}, w_{0,2}, ..., w_{0,H}]^T$. Note that $\boldsymbol{w_0}, \boldsymbol{w}_1$, ..., $\boldsymbol{w}_H$ are the parameters that need to be learned from the training data.

Given the trained neural network classifier $f(\cdot)$, the likelihood of $\boldsymbol{x}$ being positive is

$$f(\boldsymbol{x}) = \eta\left(\boldsymbol{w}_0, [1, \eta(\boldsymbol{w}_1, \boldsymbol{x}), \eta(\boldsymbol{w}_2, \boldsymbol{x}), ..., \eta(\boldsymbol{w}_H, \boldsymbol{x})]^T\right). \quad (5)$$

3.3 Problem Statement

In this paper, we study the following two questions: 1) Given a classifier, how to allocate resources to the instances in order to reduce the expected number of positives? 2) How to improve the learned classifier by incorporating new samples?

The first question implicitly hypothesizes a causal relationship between feature values and user experience, in which case, allocating resources to change feature values would improve user experience. We can first establish this relationship by using domain knowledge. For example, according to domain knowledge, user experience improves with increased bandwidth allocation. Then we can further validate or invalidate this causal relationship by the evaluation and data sampling component. We can see that the causal relationship diminishes after a certain threshold, as shown in one of the evaluations (This is partially captured by user side resource constraints in Eq. (8)). This closed-loop approach enables the practical application of data-driven resource allocation. It also addresses a drawback in the traditional abstract-utility-function-based resource allocation, where the causal relationship is not evaluated, but taken for granted.

With a learned classifier, a typical **baseline** algorithm allocates available resources as follows: in runtime, one first uses the learned classifier to predict the labels of the users. Then it allocates resources evenly among the predicted positives. In other words, the baseline algorithm uses the prediction model along with its trained weights to generate the prediction results (i.e., positives and negatives), and then allocates resources based on the predictions. Note that the learned classifier identifies a quantitative relationship between the network performance metrics and user experience. However, this information is ignored in the resource allocation step in the **baseline** algorithm.

By contrast, we explicitly use the quantitative relationship captured by the trained classifier (i.e., Eq. (5)): We incorporate the information in the resource optimization objective functions (Sec. 4). Numerical results show that our proposed framework significantly improves the performance, because it explicitly leverages the quantitative relationship to allocate resources to users.

The above approach raises the question of what a good classifier should be. We note that : 1) The objective of the classifier is not to best classify all data samples, but to best guide resource allocation in order to improve user experience with respect to the ground truth; and 2) resource allocation inevitably modifies the user distribution, and thus changes the best classifier. Therefore, we need to adapt the classifier to better quantify the relationship in the targeted regions, where users are distributed as the result of resource allocation. We study how to select such a classifier in Sec. 5.

4. RESOURCE ALLOCATION

In this section, we consider resource allocation based on a neural network model learned from the training data. We start with the problem formulation, and then propose a dual algorithm to obtain a proper solution. Finally, based on the intuition obtained from the dual algorithm, we propose a Goal algorithm with lower complexity.

4.1 Resource Allocation Problem

Assume there are M users in the system, and the overall available resources are $\boldsymbol{R} \in \mathbb{R}^{K\times 1}$. Given the classifier $f(\cdot)$, the multi-user resource optimization problem is formulated as

$$\textbf{(P-0)} \quad \min_{\boldsymbol{r_1}, \boldsymbol{r_2} \cdots \boldsymbol{r_M}} \quad \sum_{i=1}^{M} f(\boldsymbol{g}(\boldsymbol{x_i}, \boldsymbol{r_i})); \quad (6)$$

$$s.t. \quad \sum_{i=1}^{M} \boldsymbol{r_i} \le \boldsymbol{R}; \quad (7)$$

$$\boldsymbol{r_i} \in \mathbb{R}_i, \quad (8)$$

where $\boldsymbol{R}$ is the total available resource for all users, and $\mathbb{R}_i$ is the allocatable resource space for user i. Note that $\mathbb{R}_i$ can be continuous, discrete, or hybrid. For example, user i's

allocatable bandwidth is bounded by the his/her network condition and data requirement. We also note that we can include artificial bounds here so that the model is reasonably valid within the bound, in order to address the issue of diminishing causal relationship, as discussed in Sec. 3.3.

This objective function is non-convex and complex because of the nature of neural networks. It is difficult, if not impossible, to solve this problem efficiently by gradient-based numerical methods. Therefore, we propose an algorithm that solves this problem approximately in Sec. 4.2. The idea is to decompose the Lagrangian of the optimization problem into individual sub-problems. The Lagrangian multiplier serves as a signal that coordinates users to approach to the global optimal solution.

4.2 Dual Algorithm

The dual algorithm is based on dual decomposition. First, the dual algorithm determines a proper Lagrangian multiplier by solving the dual problem approximately. Then, individual users use the Lagrangian multiplier to maximize their own utilities in a distributed manner.

Since $\boldsymbol{x_i}$ is given for each user i, for the ease of notation, denote $f(\boldsymbol{g}(\boldsymbol{x_i}, \boldsymbol{r_i}))$ by $f_i(\boldsymbol{r_i})$. The Lagrangian of **(P-0)** is

$$L(\boldsymbol{r_1}, \boldsymbol{r_2}, \cdots, \boldsymbol{r_M}, \boldsymbol{\lambda}) = \sum_{i=1}^{M} f_i(\boldsymbol{r_i}) + \boldsymbol{\lambda}^T (\sum_{i=1}^{M} \boldsymbol{r_i} - \boldsymbol{R}), \quad (9)$$

where $\boldsymbol{\lambda} \in \mathbb{R}^{K \times 1}$ is the Lagrangian Multiplier (LM). The dual problem of **(P-0)** is

$$\textbf{(D-0)} \quad \max_{\boldsymbol{\lambda}} D(\boldsymbol{\lambda}) = \sum_{i=1}^{M} u_i(\boldsymbol{\lambda}) - \boldsymbol{\lambda}^T \boldsymbol{R}; \quad (10)$$

$$s.t. \quad \boldsymbol{\lambda} \geq 0, \quad (11)$$

where

$$\textbf{(D-}i\textbf{)} \quad u_i(\boldsymbol{\lambda}) = \min_{\boldsymbol{r_i}} f_i(\boldsymbol{r_i}) + \boldsymbol{\lambda}^T \boldsymbol{r_i}; \quad (12)$$

$$s.t. \quad \boldsymbol{r_i} \in \mathbb{R}_i. \quad (13)$$

Typically, $\boldsymbol{\lambda}$ is interpreted as the prices of the K types of resources, and $\boldsymbol{\lambda}^T \boldsymbol{r}_i$ is the cost of user i's resource consumption $\boldsymbol{r}_i$. As shown in **(D-i)**, given $\boldsymbol{\lambda}$, each user i minimizes its own positive likelihood plus the cost of resource consumption separately. Intuitively, when the cost of resource k increases, a user tends to reduce the consumption of resource k. We discuss how to obtain the optimal solution for problem **(D-i)**, denoted by $\boldsymbol{r_i}^*(\boldsymbol{\lambda})$, in Sec. 4.3. Note that **(D-i)** may have more than one optimal solutions.

Denote one of the optimal LM-s by $\boldsymbol{\lambda}^*$, i.e.,

$$\boldsymbol{\lambda}^* = \arg\max_{\boldsymbol{\lambda}} \min_{\boldsymbol{r_1}, \boldsymbol{r_2}, \cdots, \boldsymbol{r_M}} L(\boldsymbol{r_1}, \boldsymbol{r_2}, \cdots, \boldsymbol{r_M}, \boldsymbol{\lambda}). \quad (14)$$

Dual algorithm first finds the optimal LM $\boldsymbol{\lambda}^*$, and then allocates resources to users based on $\boldsymbol{\lambda}^*$, subject to the resource constraints. When the system has only one type of resource, binary search is used to find $\boldsymbol{\lambda}^*$ efficiently, because the resource consumed is a monotonically non-increasing function of the price $\boldsymbol{\lambda}$.

When there are multiple types of resources, we use a subgradient method to update $\boldsymbol{\lambda}$, as shown in Algorithm 1. For a given $\boldsymbol{\lambda}(t)$, we obtain the solution $\boldsymbol{r_i^*}(\boldsymbol{\lambda}(t))$ for problem **(D-i)**, and denote $\boldsymbol{r}_{need}$ as total resources consumed by users. Note that $\boldsymbol{R} - \sum_{i=1}^{M} \boldsymbol{r_i^*}(\boldsymbol{\lambda}(t))$ is one of the subgradients, we update the LM as follows (in Lines 3 to 4):

$$\boldsymbol{\lambda}(t+1) = \left[\boldsymbol{\lambda}(t) - a(t)(\boldsymbol{R} - \sum_{i=1}^{M} \boldsymbol{r_i^*}(\boldsymbol{\lambda}(t))) \right]^+, \quad (15)$$

where the step size $a(t)$ should satisfy the following conditions [12],

$$a(t) \to 0, \quad \text{as} \quad t \to \infty \quad \text{and} \quad \sum_{t=1}^{\infty} a(t) = \infty. \quad (16)$$

For instance, we can choose $a(t)$ as β/t for some positive constant β. After achieving the optimal LM $\boldsymbol{\lambda}^*$, resources are allocated in an arbitrary order, as shown in Algorithm 1 (Lines 7 to 10).

Algorithm 1: Dual Algorithm

Input : positive likelihood function $f_1(\cdot), f_2(\cdot), \cdots, f_M(\cdot)$, $\boldsymbol{\lambda}(0)$, available resource $\boldsymbol{R}$, and convergence threshold Δ.
Output: $\boldsymbol{r_1}^p, \boldsymbol{r_2}^p, \cdots, \boldsymbol{r_M}^p$

1 **while** $\|\boldsymbol{\lambda}(t+1) - \boldsymbol{\lambda}(t)\| \geq \Delta$ **do**
2 $\quad \boldsymbol{r_i}^* = \arg\min_{\boldsymbol{r_i}} f_i(\boldsymbol{r_i}) + \boldsymbol{\lambda}(t)\boldsymbol{r_i}$ s.t. $\boldsymbol{r_i} \in \mathbb{R}_i$ (Sec. 4.3);
3 $\quad \boldsymbol{r}_{need} = \sum_{i=1}^{M} \boldsymbol{r_i}^*$;
4 $\quad \boldsymbol{\lambda}(t+1) = \left[\boldsymbol{\lambda}(t) - a(t)(\boldsymbol{R} - \boldsymbol{r}_{need}) \right]^+$;
5 **end**
6 $\boldsymbol{\lambda}^* = \boldsymbol{\lambda}(t+1)$;
7 **for** *user* i **do**
8 $\quad \boldsymbol{r_i}^p = \arg\min_{\boldsymbol{r_i}} f_i(\boldsymbol{r_i}) + \boldsymbol{\lambda}^* \boldsymbol{r_i}$ (See Sec. 4.3)
$\quad$ s.t. $\boldsymbol{r_i} \in \mathbb{R}_i$ and $\boldsymbol{r_i} \leq \boldsymbol{R}$;
9 $\quad$ Update available resource: $\boldsymbol{R} = \boldsymbol{R} - \boldsymbol{r_i}^p$;
10 **end**

Since **(P-0)** is non-convex, the gap between the primal problem **(P-0)** and the dual problem **(D-0)** is not necessarily zero. Therefore, the optimal solution for the dual problem may not be the optimal solution for the primal problem. However, recent advances [21, 22, 23] on optimization with separable objectives show the duality gap is bounded as the number of users increases.

4.3 Resource Allocation for Individual Users

When **(D-i)** is convex, we can use the gradient-based method to find the optimal resource allocation for each individual user efficiently. If the utility function is non-convex, but simple, we can develop heuristics. In this paper, we consider the case where the utility function is both non-convex and complex, but the number of resource types is limited. In this case, we can use exhaustive search to obtain an approximate solution.

Considering the feasible solution set $\mathbb{R}_i$ of user i, assume for resource k, the allocated resource is bounded by $R_{i,k}^{max}$. If the step size for this type of resource is Δ_k, the number of searches performed will be $t_k = \lceil R_{i,k}^{max} / \Delta_k \rceil$. Therefore, considering K types of resources, the overall number of searches is $\prod_{k=1}^{K} t_k$. The best among the $\prod_{k=1}^{K} t_k$ candidates is chosen as the resource allocation solution for user i.

Denote the partial derivative of $\boldsymbol{g}(\boldsymbol{x}, \boldsymbol{r})$ on variable $\boldsymbol{r}$ by $g'_{d,k}$-s, where k is the resource index and d is the feature index. We have the following lemma to decide a proper step size for the exhaustive search.

LEMMA 1. *When the step size of the exhaustive search is chosen as*

$$\Delta_k = \frac{\Delta}{K(\frac{HD}{16}) \max_{1 \leq h \leq H, 1 \leq d \leq D} |w_{0,h} w_{h,d}| \max_{1 \leq d \leq D} |g'_{d,k}| + \lambda_k)}, \quad (17)$$

for resource k, the best solution found by the exhaustive search is at least Δ-suboptimal.

PROOF. Suppose $\boldsymbol{\theta} : \mathbb{R}^{n\times 1} \rightarrow \mathbb{R}^{m\times 1}$, and let $\bigtriangledown_{\boldsymbol{x}}\boldsymbol{\theta}(\boldsymbol{x})$ denote the derivative of $\boldsymbol{\theta}$ at $\boldsymbol{x}$, which is a $m \times n$ matrix. According to the chain rule, we have

$$\bigtriangledown_{\boldsymbol{r}} f(\boldsymbol{g}(\boldsymbol{x},\boldsymbol{r})) = \bigtriangledown_{\boldsymbol{x}'=\boldsymbol{g}(\boldsymbol{x},\boldsymbol{r})} f(\boldsymbol{x}') \bigtriangledown_{\boldsymbol{r}} \boldsymbol{g}(\boldsymbol{x},\boldsymbol{r}). \quad (18)$$

Since $\bigtriangledown_{\boldsymbol{x}} f(\boldsymbol{x}) = \eta'(\boldsymbol{w}_0, [\eta(\boldsymbol{w}_1,\boldsymbol{x}), \eta(\boldsymbol{w}_2,\boldsymbol{x}), ..., \eta(\boldsymbol{w}_H,\boldsymbol{x}), 1]^T)$ $[\eta'(\boldsymbol{w}_1,\boldsymbol{x}), \eta'(\boldsymbol{w}_2,\boldsymbol{x}), ..., \eta'(\boldsymbol{w}_H,\boldsymbol{x})][w_{0,1}\boldsymbol{w}_1, w_{0,2}\boldsymbol{w}_2, ..., w_{0,H}\boldsymbol{w}_H]^T$, and $|\eta'(*)| < \frac{1}{4}$, we have

$$\bigtriangledown_{\boldsymbol{r}} f(\boldsymbol{g}(\boldsymbol{x},\boldsymbol{r})) \leq \frac{HD}{16} \max_{1\leq h\leq H, 1\leq d\leq D} |w_{0,h} w_{h,d}|$$
$$[\max_{1\leq d\leq D} |g'_{d,1}|, \max_{1\leq d\leq D} |g'_{d,2}|, \cdots, \max_{1\leq d\leq D} |g'_{d,K}|]. \quad (19)$$

Assume the optimal solution is $\boldsymbol{r}^*$, then the exhaustive search covers at least one $\boldsymbol{r}'$ that satisfies

$$|\boldsymbol{r}^* - \boldsymbol{r}'| \leq [\Delta_1, \Delta_2, \cdots, \Delta_K]^T. \quad (20)$$

Therefore, we have

$$|f(g(\boldsymbol{x},\boldsymbol{r}^*)) + \boldsymbol{\lambda}^T\boldsymbol{r}^* - f(g(\boldsymbol{x},\boldsymbol{r}')) - \boldsymbol{\lambda}^T\boldsymbol{r}'|$$
$$\leq \max_{\boldsymbol{r}} |\bigtriangledown_{\boldsymbol{r}} f(\boldsymbol{g}(\boldsymbol{x},\boldsymbol{r})) + \boldsymbol{\lambda}^T||\boldsymbol{r}^* - \boldsymbol{r}'| \leq \Delta, \quad (21)$$

i.e., $\boldsymbol{r}'$ is a Δ-suboptimal solution. □

Note that when $\mathbb{R}_i$ is discrete, we can try all candidates and select the best.

4.4 Goal Algorithm

The dual algorithm may require a large number of iterations to find the optimal LM. Here, we propose a Goal algorithm with much lower complexity, inspired by the intuition of the dual algorithm, i.e., users close to the boundary area are allocated with resources with higher priority.

The dual algorithm decomposes the resource allocation among users by introducing the prices of resources. In this part, we propose the Goal algorithm with low complexity to decompose the multi-user optimization problem. First of all, we define a goal $v \in (0, 1)$ for the users. Basically, the best v can be found by testing the candidate numbers in the interval $(0, 1)$ with a small step size. For the users whose positive probability is larger than v, we calculate the minimal total resource needed for them to achieve the v. Note that the less total resource needed, the more efficient it is to allocate resources to this user. Therefore, the resources are allocated in a non-decreasing order of required resources. This algorithm has much lower complexity since no iteration is needed. However, the weakness is that all types of resources are treated equally, which does not reflect the different scarcities of resources.

We will compare the dual algorithm, the Goal algorithm, as well as the baseline (discussed in Sec. 3.3), in Sec. 6.

5. CLASSIFIER OPTIMIZATION

In the proposed framework, the "Evaluation & Data Sampling" component plays two important roles. First, it allows us to validate or invalidate the causal relationship hypothesized in Sec. 4. Specifically, similar to the idea of randomized test, if we see improved user experience after resource allocation, we validate the causal relationship between the network performance metrics and user experience. Second,

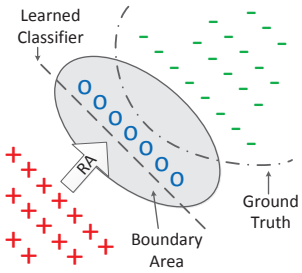


Figure 2: The necessity of classifier optimization.

it allows us to further collect data samples and to adjust the constructed classifier. Specifically, the resource allocation scheme inevitably modifies the distributions of user performance metrics, and new samples may appear in regions with few or no existing samples. Thus, the original constructed classifier needs to be adjusted.

Consider the following illustrative example, where positives ("+") and negatives ("-") are distributed in a 2D space in Fig. 2. Given this dataset, the best classifier learned from the existing positives and negatives could be a line in the middle. This classifier separates positives and negatives nicely and would be considered optimal under certain accuracy metrics. Based on this classifier, resources are allocated, where a subset of positives are moved to the boundary area, shown as "o". All these moved points are negative according to the linear classifier. However, the ground truth of these points turns out to be positive and the resource allocation reduces no positives. This is because the original dataset has no instance in the boundary area, and thus a lot of prediction errors occur in this area. Therefore, the feedback from resource allocation is necessary to obtain a better classifier that pays more attention to the targeted area of the resource allocation.

In this section, we study the issue of classifier optimization. We first discuss the desirable properties of a good classifier and then propose an iterative method to obtain a good classifier.

5.1 Optimal Classifier for Resource Allocation

We note that the goal of the classifier optimization here is different from that of the traditional classifier. Traditionally, the goal of the classifier is to maximize accuracy in predicting user labels. In contrast, the goal of the classifier here is to **best guide resource allocation** with respect to the ground truth. In particular, it needs to better quantify the relationship in the targeted region: where users are distributed as the result of resource allocation.

Specifically, denote the ground truth by a function $G(\boldsymbol{x})$, representing the positive probability for a given feature vector $\boldsymbol{x}$. The set of classifiers that can be expressed by a certain machine learning method (e.g. logistic regression, neural networks) is denoted by $\mathbb{F}$. Then, the task is to find the optimal f^* within $\mathbb{F}$ such that

$$\textbf{(P-1)} \quad f^* = \arg\min_{f\in\mathbb{F}} \sum_{i=1,2,...,M} G(\boldsymbol{g}(\boldsymbol{x_i}, \boldsymbol{r_i^*}(f))), \quad (22)$$

where $[\boldsymbol{r_1^*}(f), \boldsymbol{r_2^*}(f)...\boldsymbol{r_M^*}(f)]$ is an optimal solution of **(P-0)**. If we know the ground truth $G(\boldsymbol{x})$, we can find the

optimal f^* by optimizing **(P-1)**. However, $G(\cdot)$ is unknown and thus we need to approximate it based on the sampled data.

We are facing a problem of improving a classifier given certain historical data and opportunities to sample more data. Techniques from active learning can be leveraged to solve this problem. However, existing active learning algorithms are mainly accuracy-centric, i.e., targeting at improving the prediction accuracy. In contrast, user experience-guided resource allocation aims to improve the user experience and is usually more sensitive to the model accuracy of the targeted area of the resource allocation. Emphasis on the targeted area will be helpful for improving user experience. Moreover, a learning model tries to minimize the total loss function over all training samples, and thus the distribution of training data will affect the trained model. In traditional active learning, either streaming-based or pool-based learning, the accuracy is measured under a certain "natural" distribution that does not change. In comparison, our resource allocation schemes change the distribution, and thus, the "best" model becomes a moving target and much more difficult to obtain.

This problem also inherently requires an exploration v.s. exploitation tradeoff, where we need to balance between optimizing the user experience based on the learned classifier (exploitation) and improving the classifier by sampling more data (exploration). However, the feature space is typically huge and exploring all this space to obtain an accurate classifier will result in a large cost as noted in [17].

In practice, the problem is usually most sensitive to the targeted area of the resource allocation. Thus, a classifier that pays more attention to the points in the targeted area may be sufficient for improving the resource allocation performance. This principle motivates the design of the following iterative classifier optimization mechanism.

5.2 Iterative Classifier Optimization

Algorithm 2: Iterative Classifier Optimization

Input : Initial classifier f^0, iteration number T
Output: The best classifier f^*

1 $t = 0$;
2 **while** $t \leq T$ **do**
3 | Allocate resources based on classifier f^t using the dual algorithm;
4 | Construct a new training data by sampling the points with allocated resource;
5 | Train an updated classifier f^{t+1} based on the new training data;
6 | $t = t + 1$;
7 **end**
8 Choose f^* from $[f^0, f^1, f^2, ..., f^T]$, based on performance evaluation.

We propose an iterative algorithm to improve the classifier by taking into account the new samples after resource allocation. As shown in Algorithm 2, we adjust the classifier by utilizing the points after resource allocation. There are different ways of utilizing the historical data, with different emphases on the original data and the data after resource allocation. Specifically, the sampling method in Line 4 defines a sampling rate $p(\tau, r_{flag})$ for each historical instance. $p(\tau, r_{flag})$ is a function of two parameters: the iteration index τ and the instance type flag r_{flag} ($r_{flag} = 1$ for an instance with allocated resource, $r_{flag} = 0$ for others). Intuitively, $p(\tau, r_{flag})$ should give more emphasis on the most recent instances and instances with allocated resource. In the following, we list four sampling heuristics to achieve this goal to different degrees:

(1) Sample all the historical data, i.e, $p(\tau, r_{flag}) = 1$ for any τ and any r_{flag};

(2) Sample data only from the last iteration, i.e, $p(t-1, r_{flag}) = 1$ for any r_{flag};

(3) Sample all historical data with allocated resource (moved data) only, i.e, $p(\tau, 1) = 1$ for any τ;

(4) Sample all historical data with allocated resource (moved data) and the last iteration, i.e, $p(\tau, r_{flag}) = 1$, if $\tau = t - 1$ or $r_{flag} = 1$.

In (1), the data from all past iterations are used as training data; In (2), only the data in the last iteration are considered as training data; Both (3) and (4) contain all historically moved data, and they differ in whether the non-moved data in the last iteration are included or not. The obtained $T+1$ classifiers may not get improved every single iteration based on evaluation, and there could be oscillations. However, the algorithm keeps tracing the performance of classifiers for all iterations and chooses the best from the $T + 1$ candidates. Depending on different applications and different distributions of the data, some sampling methods may outperform the others, as shown in Sec. 6.1.

Note that instead of fixing the iteration number T, we can also use some other criteria as the stop condition. For example, stop searching when the performance is no longer increasing for the last T iterations.

6. EXPERIMENTS AND RESULTS

In this section, we test the performances of the designed algorithms based on datasets with ground truth. Experiments in our work are different from traditional machine learning experiments, where training data are used to learn a model and test data are used to evaluate the performance. This is because in this framework, we need to get the labels for the instances with allocated resource, and these instances probably are not contained by the original training or test data.[1] Therefore, we need ground truth to label new instances.

Using the experiments, we hope to answer two questions: 1) Based on the initial classifier, how efficient is the resource optimization? 2) Based on the ground truth, how efficient is the classifier optimization?

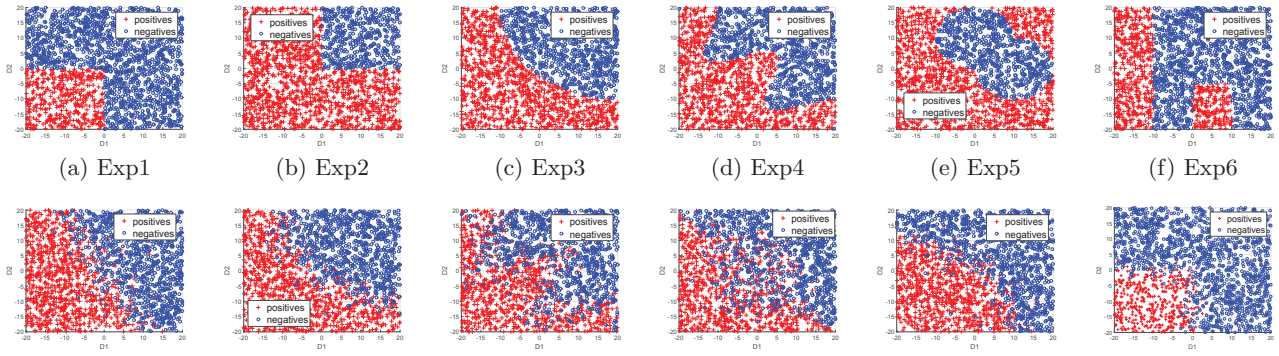
6.1 Synthetic Data

In this part, we consider 2000 points distributed evenly in a two-dimensional square $[-20, 20] \times [-20, 20]$. There are 12 cases with ground truth shown in Table 1. In experiments 1–6, we assume the ground truth is deterministic, i.e., given a location $[x_1, x_2]$, its probability to be positive is either 0 or 1. In experiments 7–12, the ground truth is probabilistic.

[1]Even though they are contained in the original dataset, labeling new instances based on the past data may not reflect the ground truth.

Table 1: Ground truth in experiments.

Exp No.	$G(\boldsymbol{x})$
E1	$G(\boldsymbol{x}) = 1$ if $x_1 < 0$ and $x_2 < 0$
E2	$G(\boldsymbol{x}) = 1$ if $x_1 < 0$ or $x_2 < 0$
E3	$G(\boldsymbol{x}) = 1$ if $\|[x_1 - 20, x_2 - 20]\| > 30$
E4	$G(\boldsymbol{x}) = 1$ if $(x_1 + x_2) < 2\cos(x_1 - x_2)$
E5	$G(\boldsymbol{x}) = 1$ if $\|[x_1 - 20, x_2 - 20]\| > 30$ and $\|[(x_1 - 20, x_2 - 20]\| > 30$
E6	$G(\boldsymbol{x}) = 1$ if **1)** $x_1 < -10$, or **2)** $x_1 \geq -20$ and $x_2 < 0$, or **3)** $x_1 \geq 0$ and $x_2 < -5$
E7	$G(\boldsymbol{x}) = 1/(1 + \exp((x_1 + x_2/2)/2))$
E8	$G(\boldsymbol{x}) = 1/(1 + \exp(-\|[x_1 - 20, x_2 - 20]\|^2/100 + 9))$
E9	$G(\boldsymbol{x}) = 1/(1 + \exp(2(x_1 + x_2 - \cos((x_1 - x_2)/5))))$
E10	$G(\boldsymbol{x}) = 1/(1 + \exp((x_1^3 + x_2^3)/1000))$
E11	$G(\boldsymbol{x}) = 1/(1 + \exp(\|[x_1 + 20, x_2 + 20]\|^2/100 - 9))$
E12	$G(\boldsymbol{x}) = 1/(1 + \exp(\max(x_1, x_2)))$



(a) Exp1 (b) Exp2 (c) Exp3 (d) Exp4 (e) Exp5 (f) Exp6

(g) Exp7 (h) Exp8 (i) Exp9 (j) Exp10 (k) Exp11 (l) Exp12

Figure 3: Positives and negatives in synthetic datasets.

We also illustrate the distributions of positives and negatives in Fig. 3.

We assume there are two types of resources that can be utilized to change feature values (and thus their locations in the 2D space). We assume

$$\boldsymbol{g}(\boldsymbol{x_i}, \boldsymbol{r_i}) = \boldsymbol{x_i} + \boldsymbol{r_i}, \tag{23}$$

i.e., one unit of resource 1 and 2 increases feature 1 (x-axis) and 2 (y-axis) by one unit, respectively. The trained neural network classifiers have 5 neurons in the hidden layer.

The resource allocation results are given in Table 2. For each experiment, we check three different combinations of resources, as shown by columns "R1" and "R2". Columns "Dual(C)", "Goal (C)" and "Baseline (C)" show the reduced positives of the resource allocation algorithms as well as the baseline according to the initially learned classifier. Since the classifier is not the ground truth, the results can be over-optimistic or under-optimistic. Columns "Dual(G)", "Goal (G)" and "Baseline (G)" show the reduced positives of the resource allocation algorithms as well as the baseline according to the ground truth. The last four columns "S1", "S2", "S3" and "S4" give the results of the dual algorithm based on the optimized classifiers using the four sampling methods presented in Sec. 5.2. The one that achieves the best performance is made bold.

Comparing columns "Dual(C)", "Goal (C)" and "Baseline (C)", we can see the dual algorithm outperforms the baseline by almost one order of magnitude, especially when the available resources are scarce (100, 100). In certain cases, the Goal algorithm can have a similar performance with the dual algorithm. However, there are also cases where the dual algorithm outperforms the Goal algorithm by more than 20%. Note that these three columns are based on the initial classifier, which is not the ground truth. Therefore, the results only show the efficiency of the optimization algorithm.

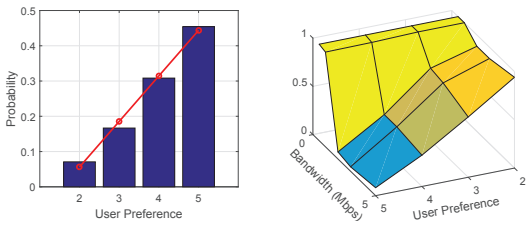
Columns "Dual(G)", "Goal (G)" and "Baseline (G)" are based on ground truth. The dual algorithm also outperforms the baseline by almost one order of magnitude. In a few experiments, the dual algorithm is outperformed by the Goal algorithm. This is due to the mismatch between the initial classifier and the ground truth. When the initial classifier is biased, optimization based on it could lead to poor performance. As shown by "E6.1", the dual algorithm according to the initial classifier reduces 58.88 positives, but actually only reduces 10 positives with respect to the ground truth.

Comparing column "Dual(G)" with columns "S1", "S2", "S3" and "S4", we can see the benefits of classifier optimization. In "E4.1", "E4.2", "E6.1" and "E9.2", the optimized classifier reduces at least 2x positives compared with the initial classifier. In "E3.1", "E4.3", "E6.2", "E8.1", "E9.3" and "E10.1", more than 50% gain is achieved by the optimized classifier. Among the four sampling methods, "S1" generally performs well, although occasionally achieves slightly smaller gain compared to the best of the four.

When the boundaries between positives v.s. negatives are non-linear and complex, it is difficult for the initial classifier to represent the boundaries well. Thus, the evaluated

Table 2: Reduced positives in experiments.

Exp No.	R1	R2	Dual(C)	Goal (C)	Baseline (C)	Dual(G)	Goal (G)	Baseline (G)	S1	S2	S3	S4
E1.1	100	100	82.88	79.16	6.01	92	89	6	**118**	101	116	117
E1.2	500	500	279.44	277.91	49.46	283	281	49	**285**	274	282	284
E1.3	1000	1000	376.52	370.95	101.05	386	390	110	405	386	403	**406**
E2.1	100	100	118.86	121.45	1.65	121	124	4	**126**	112	124	123
E2.2	500	500	281.99	281.52	13.38	289	289	15	**306**	304	**306**	305
E2.3	1000	1000	330.57	306.89	23.74	376	346	24	432	425	433	**436**
E3.1	100	100	73.47	66.89	3.16	67	63	6	**117**	94	113	112
E3.2	500	500	239.07	217.7	23.2	243	199	27	**304**	230	298	278
E3.3	1000	1000	381.03	343.92	63.39	375	311	52	**432**	336	**432**	425
E4.1	100	100	16.69	13.85	4.32	31	25	7	**131**	95	111	106
E4.2	500	500	90.73	64.64	20.51	133	84	28	**353**	299	340	**353**
E4.3	1000	1000	163.69	139.32	41.1	276	167	42	**520**	425	499	518
E5.1	100	100	54.5	49.48	2.03	69	55	4	93	80	73	**96**
E5.2	500	500	173.62	171.01	6.01	202	201	4	**251**	204	243	245
E5.3	1000	1000	307.75	300.92	21.81	249	236	11	**364**	292	339	344
E6.1	100	100	58.88	57.48	2.33	10	8	1	**102**	81	82	93
E6.2	500	500	184	147.23	35.55	212	191	45	**306**	249	241	279
E6.3	1000	1000	328.42	275.79	95.05	338	295	103	**427**	340	414	415
E7.1	100	100	34.24	28.96	3.78	16.89	17.34	4.31	17.15	**17.91**	17.43	17.61
E7.2	500	500	145.55	126.31	22.68	67.42	75.94	19.8	83.24	**83.26**	80.81	80.13
E7.3	1000	1000	165.67	146.57	43.91	141.1	150.62	44.84	154.12	**155.89**	153.82	154.05
E8.1	100	100	48.4	38.07	3.46	16.9	14.26	3.98	**26.35**	24.81	24.17	25.72
E8.2	500	500	119.23	101.05	20.47	113.18	94.3	19.88	**119.3**	113.38	113.75	117.68
E8.3	1000	1000	233.92	201.29	47.35	207.42	172.09	46.59	211.83	210.94	207.35	**212.69**
E9.1	100	100	10.86	8.45	3.52	20.22	14.1	4.28	**24.52**	23.77	22.77	23.53
E9.2	500	500	119.38	83.68	17.18	10.99	6.34	19.85	**117.23**	103.13	107.42	110.76
E9.3	1000	1000	129.01	100.06	46.17	126.82	90.44	41.07	**211.46**	191.22	195	205.48
E10.1	100	100	22.24	16.81	4.5	19.16	11.66	3.65	**31.33**	29.08	24.31	28.52
E10.2	500	500	90.19	88.37	21.06	79.67	72.38	27.71	**112.36**	103.24	107.89	104.3
E10	1000	1000	111.45	93.72	46.7	128.77	109.79	62.68	**188.94**	159.6	178.62	176.77
E11.1	100	100	32.78	30.37	4.05	25.33	24.2	3.64	**26.33**	25.92	25.47	26.24
E11.2	500	500	135.55	123.34	23.87	108.17	103.11	20	**118.89**	113.12	114.14	116.71
E11.3	1000	1000	227.41	201.03	49.21	173.87	149.37	39.03	212.14	209.52	**212.22**	208.8
E12.1	100	100	52.14	46.56	5.05	39.3	38.58	4.47	43.21	42.85	**43.48**	43.04
E12.2	500	500	164.79	152.28	31.81	146.56	141.85	25.24	**169.63**	164.4	167.75	167.91
E12.3	1000	1000	280.08	249.63	62.74	205.03	188.39	51.81	272.84	265.38	270.2	**279.11**



(a) Distribution of user preferences. (b) Ground Truth of the video dataset.

Figure 4: Statistics about video dataset.

performance according to the initial classifier could be very different from the evaluated performance according to the ground truth, e.g, "E4", "E5", "E6" and "E9". In this case, the gain of classifier optimization is usually larger.

6.2 Real-World Dataset

From properly conducted controlled experiments, we can study the causal relation between features and corresponding labels. In the following, we introduce two datasets collected from controlled experiments. Both datasets are available online. We first construct ground truth using statistical analysis on the collected data. Then the ground truth is used to label new instances. We know in a real system, the ground truth is typically not available. However, this method enables us to evaluate the proposed mechanisms.

6.2.1 Video MOS Dataset

The first dataset comes from experiments conducted in [13]. The goal of this experiment is to study the effect of network features on user experience for video services. The dataset contains 8 original videos and 8×5 streamed videos. The streamed videos are the original videos transmitted over a limited network bandwidth. Each video is viewed and subjectively scored by 30 viewers. Therefore the dataset includes 240 views of the original videos and 1200 views of the streamed videos. Each view is associated with a MOS (Mean Opinion Score) given by the subject. MOS, a score from 1 to 5, is a subject measurement of user experience in many applications. For each view, it has two features: the bandwidth used to transmit the video (x_1) and the viewer's preference (x_2). The bandwidth has 5 discrete values: 0.5, 1, 2, 3 and 5 Mb/s. A viewer's preference is the score the viewer gives to the original video. The preferences are related to the video contents and video quality but not any network features. In the dataset, all preferences are larger than 1, and the histogram is shown in Fig. 4(a). With a linear regression, we have the following estimation of the pdf of user preferences,

$$p(x_2) = -0.2021 + 0.1292x_2, \quad \text{for } 2 \le x_2 \le 5. \tag{24}$$

We assume that when MOS ≤ 3, the viewer is unsatisfied with the viewed video and thus a positive instance. Otherwise, the viewer is satisfied and thus a negative instance. In Fig. 4(b), we show the effect of bandwidth and user preference on positive ratios. When the bandwidth is less than 2.5 Mbps, the users are unsatisfied with probability 1. When the bandwidth is greater than 2.5 Mbps, whether a user is satisfied or not depends on the user preference, and the relation is almost linear. Therefore, the ground truth we generate

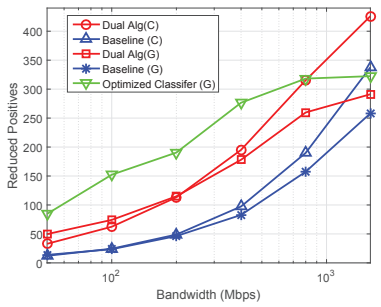


Figure 5: Resource vs Reduced Positives (Video MOS).

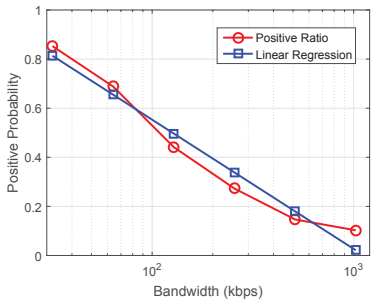


Figure 6: Ground Truth (Web Browsing).

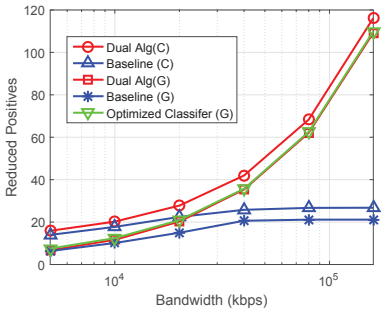


Figure 7: Resource vs Reduced Positives (Web Browsing).

from the dataset as

$$G(\boldsymbol{x}) = \max\left(\mathbf{1}(x_1 < 2.5),\ 1.5036 - 0.2853x_2\right), \qquad (25)$$

where $\mathbf{1}(\cdot)$ is the indicator function.

We consider 2000 users in the system, with each user watching one video. The user preference is generated according to the distribution given by Eq. (24), and the bandwidth received is assumed to be Gaussian distributed with mean 2.5 Mbps and standard deviation 0.5 Mbps. The classifier we consider is a neural network with 5 neurons in the hidden layer.

The resource allocated is (additional) bandwidth, and one unit of bandwidth increases the value of feature 1 by one unit. Resource allocation results are shown in Fig. 5. We compare the dual algorithm with the baseline, under the initial classifier ("C") and ground truth ("G"), as well as with the best classifier from the four sampling methods. Each sampling method outputs an optimized classifier, and the best among the four is chosen. In Fig. 5, we change the available bandwidth from 50 to 1600 Mbps. Blue curves show the performance of the baseline according to the initial classifier (w/ $\triangle$) and the ground truth (w/ $*$). Red curves show the performance of the dual algorithm according to the initial classifier (w/ $\circ$)and the ground truth (w/ $\square$). The dual algorithm achieves at least 2x performance gain compared with the baseline when the available bandwidth resource is less than 400 Mbps. When the resource allocation is based on the optimized classifier (green curve, w/ $\triangledown$), the reduced number of positives is further improved by 50%. When the resource is over 800 Mbps, the gap between the actual performance (based on the ground truth) and the predicted performance (based on the initial classifier) becomes much larger. This demonstrates the necessity of classifier optimization.

6.2.2 *Web Browsing Dataset*

In dataset [15], the authors conducted controlled experiments to study the relation between bandwidth and user experience for web browsing. The dataset contains 572 MOS values for users who surfed four types of web sites. After removing the instances with insufficient data, we have 418 valid instances. We assume when the MOS ≤ 3, the user is unsatisfied, i.e., a positive instance.

From the dataset, we find that the positive ratio goes down linearly with the logarithm of bandwidth, as shown in Fig. 6. Therefore we assume the ground truth in this dataset is

$$G(x) = \min\left(\max(1.6058 - 0.2286x, 0), 1\right); \qquad (26)$$

We generate 10,000 users whose bandwidth is Gaussian distributed with mean 512 kbps and standard deviation 128 kbps. The neural network classifiers we considered have two neurons in the hidden layer.

Additional bandwidth is the available resource we can allocate to users, and one unit of bandwidth increases the feature by one unit. Resource allocation results are shown in Fig. 7. When the available resource is larger than 10,000 kbps, the gain increases exponentially. The reason is that the positive probability is relatively low in this dataset, and the learned classifier can only pick up a few positives with high true positive rate. Since the baseline only allocates bandwidth to predicted positives, which are a small portion of real positives, the reduced positives stop increasing when the resource is sufficient. The dual algorithm takes the learned classifier as the objective function for resource optimization, and therefore it does not suffer from this issue. It is also shown by the figure that the initial classifier is over-optimistic compared with the ground truth. In this dataset, the gain of the optimized classifier over the initial classifier is limited.

7. DISCUSSIONS

The proposed algorithms can handle cases with a large number of features. However, if there are many types of resources to allocate, it will be expensive to use exhaustive search to find individual optimum. Therefore, our algorithms are suitable to the scenarios where the number of resource types is small, which is usually the case in computer networks.

Most existing resource allocation mechanisms rely on domain knowledge to decide resource allocation. The proposed framework provides an alternative way of resource allocation, and can be used to augment existing mechanisms. If user experience is the ultimate goal, collecting user experience data is a necessary step. Sometimes, user experience data comes naturally (e.g., user call complains [3]), and other times, one needs to collect user experience data (e.g., video MOS scores). Once such data collection exists, one is able to build a model offline and adjust the model periodically. The classifier optimization algorithm needs to train multiple classifiers iteratively based on the feedback of resource allocation. In computer networks, the feedback of re-

source allocation may often be obtained quickly. Therefore, the classifier optimization could obtain a good classifier in a relatively short amount of time, which makes the closed-loop framework practical.

When adjusting the classifier, we only use heuristic algorithms to sample the historical data, including the original data and the data with allocated resource. However, there is still a lack of knowledge about the areas that are neither covered by original data set nor the targeted data set. This lack of information may make the learned classifier inaccurate and result in suboptimal resource allocation. Therefore, from a long term perspective, we may need to temporally apply a suboptimal resource allocation scheme to explore those areas. This will require an appropriate exploration-exploitation tradeoff and will be investigated in the future.

There is a tradeoff between maximizing efficiency and guaranteeing fairness. As our algorithms tend to focus on the users whose experience can be improved with the least amount of resources, the fairness among users could deteriorate. In the future, we will consider the balance between efficiency and fairness.

8. CONCLUSIONS

In this paper, we propose a closed loop approach to the data-driven resource allocation problem where the objective is to minimize the number of positives based on neural networks. We consider a novel framework, where the classifier learned by machine learning is utilized as the objective function in resource allocation, and the classifier is further optimized based on post-allocation evaluation and feedback. We design a dual algorithm to coordinate the resource allocation among users, and a classifier optimization algorithm to find the most proper neural network classifier. Experiments using synthetic data and real data from computer networks show our algorithms can reduce the expected number of unsatisfied users by up to 2x compared with the baseline, while the classifier optimization further improves the performance by 50%.

9. ACKNOWLEDGMENTS

This work was partially supported by NSF through grants CNS-1547461; CNS-1457060; CCF-1423542.

10. REFERENCES

[1] A. Balachandran, V. Aggarwal, E. Halepovic, J. Pang, S. Seshan, S. Venkataraman, and H. Yan. Modeling web quality-of-experience on cellular networks. In *MobiCom*, pages 213–224. ACM, 2014.

[2] A. Balachandran, V. Sekar, A. Akella, S. Seshan, I. Stoica, and H. Zhang. Developing a predictive model of quality of experience for internet video. In *ACM SIGCOMM Computer Communication Review*, volume 43, pages 339–350. ACM, 2013.

[3] Y. Bao, X. Liu, and A. Pande. Data-guided approach for learning and improving user experience in computer networks. *ACML 2015*.

[4] M. Bayati, M. Braverman, M. Gillam, K. M. Mack, G. Ruiz, M. S. Smith, and E. Horvitz. Data-driven decisions for reducing readmissions for heart failure: General methodology and case study. *PloS one*, 9(10):e109264, 2014.

[5] J. L. Berral, Í. Goiri, R. Nou, F. Julià, J. Guitart, R. Gavaldà, and J. Torres. Towards energy-aware scheduling in data centers using machine learning. In *e-Energy*, pages 215–224. ACM, 2010.

[6] A. J. Chan, A. Pande, E. Baik, and P. Mohapatra. Temporal quality assessment for mobile videos. In *MobiCom*, pages 221–232. ACM, 2012.

[7] M. Chiang, S. Zhang, and P. Hande. Distributed rate allocation for inelastic flows: Optimization frameworks, optimality conditions, and optimal algorithms. In *INFOCOM*. IEEE, 2005.

[8] M. Fazel and M. Chiang. Network utility maximization with nonconcave utilities using sum-of-squares method. In *CDC-ECC'05*. IEEE, 2005.

[9] E. Horvitz and T. Mitchell. From data to knowledge to action: a global enabler for the 21st century. computing community consortium, v. 11. sep., 2010.

[10] S. S. Krishnan and R. K. Sitaraman. Video stream quality impacts viewer behavior: inferring causality using quasi-experimental designs. *Networking, IEEE/ACM Transactions on*, 21(6):2001–2014, 2013.

[11] G. Lee, N. Tolia, P. Ranganathan, and R. H. Katz. Topology-aware resource allocation for data-intensive workloads. In *APSys*, pages 1–6. ACM, 2010.

[12] J.-W. Lee, R. R. Mazumdar, and N. B. Shroff. Non-convex optimization and rate control for multi-class services in the internet. *Networking, IEEE/ACM Transactions on*, 13(4):827–840, 2005.

[13] P. Paudyal, F. Battisti, and M. Carli. A study on the effects of quality of service parameters on perceived video quality. In *EUVIP*, December 2014.

[14] M. Qu, H. Zhu, J. Liu, G. Liu, and H. Xiong. A cost-effective recommender system for taxi drivers. In *Proceedings of the 20th ACM SIGKDD*. ACM, 2014.

[15] R. Schatz and S. Egger. An annotated dataset for web browsing QoE. In *QoMEX*, pages 61–62, Sept 2014.

[16] M. Z. Shafiq, J. Erman, L. Ji, A. X. Liu, J. Pang, and J. Wang. Understanding the impact of network dynamics on mobile video user engagement. In *2014 ACM SIGMETRICS*, pages 367–379. ACM, 2014.

[17] A. Slivkins. Contextual bandits with similarity information. *The Journal of Machine Learning Research*, 15(1):2533–2568, 2014.

[18] G. Song and Y. Li. Utility-based resource allocation and scheduling in OFDM-based wireless broadband networks. *Communications Magazine, IEEE*, 43(12):127–134, 2005.

[19] T. Tulabandhula and C. Rudin. Machine learning with operational costs. *The Journal of Machine Learning Research*, 14(1):1989–2028, 2013.

[20] T. Tulabandhula, C. Rudin, and P. Jaillet. Machine learning and the traveling repairman. *arXiv preprint arXiv:1104.5061*, 2011.

[21] M. Udell and S. Boyd. Maximizing a sum of sigmoids.

[22] M. Udell and S. Boyd. Bounding duality gap for problems with separable objective. *arXiv preprint arXiv:1410.4158*, 2014.

[23] M. Wang. Vanishing price of anarchy in large coordinative nonconvex optimization. Available at http://www.optimization-online.org/DB_HTML/2015/07/5021.html.