Implicit Rate-Constrained Optimization of Non-decomposable Objectives

Abhishek Kumar [1]   **Harikrishna Narasimhan** [1]   **Andrew Cotter** [1]

Abstract

We consider a popular family of constrained optimization problems arising in machine learning that involve optimizing a non-decomposable evaluation metric with a certain thresholded form, while constraining another metric of interest. Examples of such problems include optimizing the false negative rate at a fixed false positive rate, optimizing precision at a fixed recall, optimizing the area under the precision-recall or ROC curves, etc. Our key idea is to formulate a rate-constrained optimization that expresses the threshold parameter as a function of the model parameters via the Implicit Function theorem. We show how the resulting optimization problem can be solved using standard gradient based methods. Experiments on benchmark datasets demonstrate the effectiveness of our proposed method over existing state-of-the-art approaches for these problems.

1. Introduction

In many modern machine learning applications, the performance of a model is evaluated using metrics that are complex and nuanced. For example, in retrieval systems, it is common to evaluate a scoring model based the area under the precision-recall curve, or the ROC curve, or on its precision at a certain recall value (Eban et al., 2017). Similarly, in many medical diagnostic applications, a model is required to yield low false positive rates while restricting its false negatives rate to be within an allowed limit (Rao et al., 2008), while in machine learning fairness applications one might be interested in imposing the "80% rule", which requires a positive prediction rate of at least 80% on the minority class (Biddle, 2006; Zafar et al., 2017).

The above problems cannot be directly solved by minimizing a standard classification loss. In fact, prior work has found that doing so can result in inferior model performance (Joachims, 2005; Koyejo et al., 2014; Kar et al., 2014; Eban

---

[1]Google Research. Correspondence to: Abhishek Kumar <abhishk@google.com>.

et al., 2017; Cotter et al., 2019a). Moreover, many of the metrics that we are interested in have a non-decomposable structure, *i.e.*, they cannot be expressed directly in terms of an average over individual data points, making them hard to optimize using standard optimization tools. Much prior work has sought to address this problem, resulting in a range of methods targeting different classes of non-decomposable metrics (Yue et al., 2007; Narasimhan & Agarwal, 2013b; Kar et al., 2015; Yan et al., 2018).

In this paper, we consider a popular family of non-decomposable objectives that have a certain thresholded form. This includes metrics like the false negative rate (FNR) at a certain fixed false positive rate (FPR), precision at a fixed recall, precision@K, AUC-PR, and AUC-ROC, as well as more recent threshold-based fairness metrics (Hardt et al., 2016). The task of optimizing these metrics can naturally be written as a constrained optimization problem, wherein one seeks to optimize a quantity such as the model's precision or false positive rate at one or more thresholds, subject to the model satisfying a set of rate constraints at those thresholds. The dominant approach for solving such rate-constrained problems has been to relax the constraints with surrogate losses, and to formulate an equivalent Lagrangian-based primal-dual problem (Eban et al., 2017). Follow-up work has improved upon this approach by using the surrogate relaxations only for the primal updates, but not the dual (Cotter et al., 2019b; Narasimhan et al., 2019a).

Our proposed optimization strategy departs significantly from the prior Lagrangian-based methods, and avoids explicitly solving the constrained optimization problem. Instead, we express the threshold variables in the optimization problem as an *implicit* function of the model parameters, and thus re-formulate the problem as an unconstrained optimization over the model parameters. By appealing to the Implicit Function Theorem (Tu, 2011), we show how to compute the gradients for the resulting unconstrained objective, despite not knowing the form of the implicit function, and then use them to perform standard gradient-based optimization (see Section 3). Although the Implicit Function Theorem makes a local statement about the existence of the implicit function (*i.e.*, valid in a small neighborhood around current model parameters), we can still effectively use the theorem to make local gradient updates towards optimizing the objective.

We experiment with two image classification datasets and several UCI datasets, and show that our proposed method often performs significantly better than the state-of-the-art constrained optimization solvers in optimizing popular metrics such as FNR at fixed FPR, and the "partial" areas under the ROC and Precision-Recall curves evaluated at a selection range of FPR/recall values (see Section 5). We find our approach to be particularly effective when used to target extreme values of FPR or recall. We also discuss how our formulation can be extended to apply to more complex learning problems, such as query-based ranking, where standard constrained optimization techniques are known to have notable drawbacks (see Section 6).

2. Problem Formulation

We describe our formulation in a binary classification setting, with input space X and binary labels $\{0, 1\}$. Later, we will discuss how to extend our setup to multi-class classification problems. Our goal is to learn a scoring model $s^\theta : X \to \mathbb{R}$, parameterized by $\theta \in \mathbb{R}^p$, whose scores can be thresholded to make a binary prediction. We denote a scoring model thresholded at t by $s_t^\theta(x) = \mathbf{1}(s^\theta(x) > t) \in \{0, 1\}$. We will use $\text{TP}(s_t^\theta)$, $\text{FP}(s_t^\theta)$ and $\text{FN}(s_t^\theta)$ to denote the true positives, false positive and false negatives respectively for the thresholded classifier.

We are interested in solving constrained optimization problems of the form:

$$\min_{\theta \in \mathbb{R}^p} f(\theta, \lambda) \quad \text{s.t.} \quad g(\theta, \lambda) = \mathbf{0} \qquad (1)$$

where the objective $f : \mathbb{R}^p \times \mathbb{R}^m \to \mathbb{R}$ maps the model parameters $\theta \in \mathbb{R}^p$ and a set of m thresholds $\lambda \in \mathbb{R}^m$ to a real value, and the constraints $g : \mathbb{R}^p \times \mathbb{R}^m \to \mathbb{R}^m$ map (θ, λ) to m real numbers. We further assume θ stays in the feasible region, i.e., $\forall \theta, \exists \lambda$ s.t. $g(\theta, \lambda) = \mathbf{0}$. We will further discuss how several commonly used constrained optimization problems of interest in machine learning satisfy this assumption of feasibility. We provide some popular examples of evaluation metrics below.

Example 1 (Precision at fixed recall). *To maximize the model's precision at the threshold $\lambda \in \mathbb{R}$ at which its recall is β, we will have:*

$$
\begin{aligned}
f(\theta, \lambda) &= -\text{precision}(s_\lambda^\theta) = -\frac{\text{TP}(s_\lambda^\theta)}{\text{TP}(s_\lambda^\theta) + \text{FP}(s_\lambda^\theta)}; \\
g(\theta, \lambda) &= \text{recall}(s_\lambda^\theta) - \beta = \frac{\text{TP}(s_\lambda^\theta)}{\text{TP}(s_\lambda^\theta) + \text{FN}(s_\lambda^\theta)} - \beta.
\end{aligned}
$$

Example 2 (FNR at fixed FPR). *To minimize the model's false negative rate at the threshold $\lambda \in \mathbb{R}$ at which its false positive rate is $\beta \in [0, 1]$, we will have:*

$$
\begin{aligned}
f(\theta, \lambda) &= \text{FNR}(s_\lambda^\theta) = \frac{\text{FN}(s_\lambda^\theta)}{\text{TN}(s_\lambda^\theta) + \text{FP}(s_\lambda^\theta)}; \\
g(\theta, \lambda) &= \text{FPR}(s_\lambda^\theta) = \frac{\text{FP}(s_\lambda^\theta)}{\text{TP}(s_\lambda^\theta) + \text{FN}(s_\lambda^\theta)} - \beta.
\end{aligned}
$$

Example 3 (Precision at k). *To maximize the model's precision at the threshold $\lambda \in \mathbb{R}$ at which it achieves a coverage of k, we can set:*

$$
\begin{aligned}
f(\theta, \lambda) &= -\text{precision}(s_\lambda^\theta); \\
g(\theta, \lambda) &= \text{TP}(s_\lambda^\theta) + \text{FP}(s_\lambda^\theta) - k.
\end{aligned}
$$

Example 4 (AUC-PR). *To maximize the area under the Precision-Recall curve, following Eban et al. (2017), we use a Riemann approximation to the area: we divide the recall range into m equally-spaced values $\beta_1, \ldots, \beta_m \in [0, 1]$, and evaluate the average precision that the model achieves when thresholded to match each of the target recalls. This can be written as a constrained optimization problem with m thresholds $\lambda_1, \ldots, \lambda_m \in \mathbb{R}$, and with the objective and constraints set to:*

$$
\begin{aligned}
f(\theta, \lambda) &= -\frac{1}{m} \sum_{i=1}^{m} \text{precision}(s_{\lambda_i}^\theta) \\
g_i(\theta, \lambda) &= \text{recall}(s_{\lambda_i}^\theta) - \beta_i, \forall i \in [m].
\end{aligned}
$$

One can similarly compute the "partial" area under the PR curve in any given range of recall (precision) targets by thresholding only at those particular recall (precision) values. This is particularly useful for excluding low recalls (precisions), since one is generally uninterested in the performance of the model at such thresholds.

Example 5 (AUC-ROC). *To maximize the (partial) area under the Receiver Operator Characteristic (ROC) curve, we can again apply a Riemann approximation: we divide the FPR range into m values $\beta_1, \ldots, \beta_m$, and compute the average TPR at m thresholds $\lambda_1, \ldots, \lambda_m \in \mathbb{R}$, chosen to satisfy the FPR targets:*

$$
\begin{aligned}
f(\theta, \lambda) &= -\frac{1}{m} \sum_{i=1}^{m} \text{TPR}(s_{\lambda_i}^\theta) \\
g_i(\theta, \lambda) &= \text{FPR}(s_{\lambda_i}^\theta) - \beta_i, \forall i \in [m].
\end{aligned}
$$

Example 6 (Fairness criterion). *In a typical group fairness application (Hardt et al., 2016), each example belongs to one of m protected groups, and the goal is to constrain the model to have equitable performance across all groups. One way to enforce this requirement is to introduce a separate threshold λ_i for examples from each group, and to tune them to satisfy the fairness constraints. For example, the popular* demographic parity *constraint for two (disjoint)*

groups, which requires equal positive prediction rates for both groups, can be encoded as:

$$g(\theta, \lambda) = \frac{1}{m_1}\left(\mathrm{TP}_1(s^\theta_{\lambda_1}) + \mathrm{FP}_1(s^\theta_{\lambda_1})\right)$$
$$- \frac{1}{m_2}\left(\mathrm{TP}_2(s^\theta_{\lambda_2}) + \mathrm{FP}_2(s^\theta_{\lambda_2})\right),$$

where $\mathrm{TP}_1, \mathrm{TP}_2$ *are the true positives on examples belonging group* 1 *and* 2 *respectively,* $\mathrm{FP}_1, \mathrm{FP}_2$ *are the false positives for the two groups, and* m_1 *and* m_2 *are the number of examples in the two groups.*

Feasibility of constraints by tuning λ. Assuming that the model does not map two different training examples to same outputs, it is easy to see that for fixed model parameters θ, any of the aforementioned rate constraints (*e.g.*, false positive rate, precision, recall, *etc.*) can be satisfied to any feasible value by tuning *only* the thresholds $\lambda_1, \ldots, \lambda_m$.

All the rate based optimization objectives and constraints discussed above are non-smooth. To make the problem amenable to gradient based optimization, following earlier work (Eban et al., 2017; Cotter et al., 2019b; Narasimhan et al., 2019a) we replace f and g with smooth differentiable surrogates $\tilde{f}$ and $\tilde{g}$, and relax (1) into:

$$\min_{\theta \in \mathbb{R}^p} \tilde{f}(\theta, \lambda) \quad \text{s.t.} \quad \tilde{g}(\theta, \lambda) = \mathbf{0}. \quad (2)$$

Surrogate losses. We use sigmoid and softplus functions, denoted as $\sigma(\cdot)$, as surrogates in our experiments. Specifically, we replace the innermost indicators with smooth surrogate σ. For example, if the objective f is $\mathrm{FNR} = (\frac{\mathrm{FN}}{\text{total positives}})$ and p_i is the prediction (logit) for ith example, then we replace $\mathrm{FN} = \sum_{i:y_i=1} \mathbb{I}_{p_i<\lambda}$ with $\widehat{\mathrm{FN}} = \sum_{i:y_i=1} \sigma_\tau(-(p_i - \lambda))$, yielding the surrogate $\tilde{f} = \frac{\widehat{\mathrm{FN}}}{\text{total positives}}$ (where $\sigma_\tau(x) = \sigma(\tau x)$ denotes a temperature scaled sigmoid or softplus function). We use similar surrogates for ratio based objectives such as precision and recall. For example, if f is precision ($\frac{\mathrm{TP}}{\text{predicted positives}}$) then we replace $\mathrm{TP} = \sum_{i:y_i=1} \mathbb{I}_{p_i>\lambda}$ with $\widetilde{\mathrm{TP}} = \sum_{i:y_i=1} \sigma_\tau(p_i - \lambda)$ and predicted-positives $= \sum_i \mathbb{I}_{p_i>\lambda}$ with $\widetilde{\mathrm{PP}} = \sum_i \sigma_\tau(p_i - \lambda)$, yielding the surrogate $\tilde{f} = \frac{\widetilde{\mathrm{TP}}}{\widetilde{\mathrm{PP}}}$.

3. Optimization with Implicit Thresholds

The canonical approach to solving the constrained optimization problem in (2) is to formulate a Lagrangian for the problem, and then perform gradient updates to maximize the Lagrangian over the mulitipliers and minimize it over θ and λ. Our key idea is to avoid explicitly solving the constrained problem by instead formulating an equivalent *unconstrained* problem in which we express the thresholds λ as an *implicit* function of the model parameters θ (within a neighborhood around θ).

To this end, we make use of the Implicit Function Theorem (Tu, 2011). Specifically, suppose the point $(\theta^0, \lambda^0) \in \mathbb{R}^{p+m}$ satisfies the constraint, *i.e.*, $\tilde{g}(\theta^0, \lambda^0) = 0$. Then we can express the thresholds as $\lambda^0 = \tilde{h}(\theta^0)$ in a neighborhood around θ^0, for some implicit function $\tilde{h}$:

Theorem 1 (**Implicit Function Theorem** (Tu, 2011)). *Let U be an open subset in $\mathbb{R}^p \times \mathbb{R}^m$ and $\tilde{g} : U \to \mathbb{R}^m$ a C^1 map. Write (θ, λ) for a point in U, with $\theta \in \mathbb{R}^p, \lambda \in \mathbb{R}^m$. At a point $(\theta^0, \lambda^0) \in U$ where $\tilde{g}(\theta^0, \lambda^0) = 0$ and the determinant $\det[\partial \tilde{g}^i/\partial \lambda^j(\theta^0, \lambda^0)]$ is nonzero, there exists a neighborhood $\Theta \times \Lambda$ of (θ^0, λ^0) in U and a unique C^1 function $\tilde{h} : \Theta \to \Lambda$ such that in $\Theta \times \Lambda \subset U \subset \mathbb{R}^p \times \mathbb{R}^m$,*

$$\tilde{g}(\theta, \lambda) = 0 \quad \Longleftrightarrow \quad \lambda = \tilde{h}(\theta)$$

Using this theorem, we can write the implicit threshold as $\lambda = \tilde{h}(\theta)$ within a neighborhood around θ^0, which enables us to turn problem (2) into the equivalent unconstrained problem of minimizing $\tilde{f}(\theta, \tilde{h}(\theta))$.

3.1. Characterization of the implicit function

A differentiable function $\tilde{h}(\theta) \in \mathbb{R}^m$ that provides us with the m thresholds at which $\tilde{f}(\theta, \tilde{h}(\theta)) = \mathbf{0}$ may not always exist, and even if it does, it may be available in closed-form only in some highly simplified settings.[1] Nonetheless, under some assumptions, we can show that when $\tilde{h}$ exists, the resulting composite function $\tilde{f}(\theta, \tilde{h}(\theta))$ is convex in θ.

Proposition 1. *Let $m = 1$. Suppose the objective $\tilde{f}(\theta, \lambda)$ is jointly convex in (θ, λ) and is strictly increasing in $\lambda \in \mathbb{R}$, and the constraint $\tilde{g}(\theta, \lambda)$ is jointly convex in (θ, λ) and is strictly descreasing in $\lambda \in \mathbb{R}$. Suppose there exists a C^1 function $\tilde{h} : \mathbb{R}^p \to \mathbb{R}$ such that $\tilde{g}(\theta, \tilde{h}(\theta)) = 0, \forall \theta$. Then $\tilde{h}$ is convex in θ. Consequently, the composite objective $\tilde{f}(\theta, \tilde{h}(\theta))$ is convex in θ.*

The proof adapts a result from Wurker (2001) and is given in Appendix A. The assumptions in the proposition hold for simple linear models, for example, when minimizing the FPR while constraining the FNR (see appendix for details).

3.2. Gradient computation

To compute a (local) derivative for $\tilde{f}(\theta, \tilde{h}(\theta))$ w.r.t. θ within the neighborhood of θ^0 in Theorem 1, we use:

$$\nabla_\theta \tilde{f}(\theta, \tilde{h}(\theta)) = \nabla_\theta \tilde{f}(\theta, \lambda) + \frac{\partial \tilde{f}(\theta, \lambda)}{\partial \lambda} \nabla_\theta \tilde{h}(\theta), \quad (3)$$

---

[1]For example, if the distribution of the instances $x \in \mathbb{R}^p$ conditioned on label $y = 1$ is a Gaussian distribution with mean $\mu \in \mathbb{R}^p$ and covariance matrix $\Sigma \in \mathbb{R}^{p \times p}$, and suppose we wish to constrain the false negative rate (FNR) for a linear model parameterized by $\theta \in \mathbb{R}^p$. Then the FNR at any threshold $\lambda \in \mathbb{R}$ is given by $\mathbb{P}_{x \sim \mathcal{D}_+}\left(\theta^\top x \le \lambda\right)$, and the threshold λ at which the FNR is β is given by $\tilde{h}(\theta) = \Phi^{-1}(\beta)$, where Φ is the CDF of a normal distribution with mean $\theta^\top \mu$ and variance $\theta^\top \Sigma \theta$.

Algorithm 1 Implicit Constrained Optimization (ICO)

1: **Hyper-parameters:** OPT, $\tau \in \mathbb{N}$
2: **Intialize:** θ^0, λ^0
3: **For** $t = 1$ to T:
4: $\quad H^t = -\nabla_\theta \tilde{g}(\theta^t, \lambda^t) / \frac{\partial \tilde{g}(\theta^t, \lambda^t)}{\partial \lambda}$
5: $\quad G^t = \nabla_\theta \tilde{f}(\theta^t, \lambda^t) + \frac{\partial \tilde{f}(\theta^t, \lambda^t)}{\partial \lambda} H^t$
6: $\quad \theta^{t+1} = \text{OPT}(\theta^t, G^t)$ // optimizer step
7: $\quad$ **If** $t \bmod \tau = 0$:   // correction step
8: $\quad\quad$ Set λ^{t+1} s.t. $g(\theta^{t+1}, \lambda^{t+1}) = \mathbf{0}$
9: $\quad$ **Else**:   // gradient based update for λ
10: $\quad\quad \lambda^{t+1} = \lambda^t + \langle \nabla_\theta \tilde{h}(\theta^t), \theta^{t+1} - \theta^t \rangle$
11: $\quad$ **End If**
12: **End For**
13: **Return** θ

where for simplicity we show the derivative for a scalar λ. We will further need the derivative of the implicit function $\tilde{h}(\cdot)$. Since $\tilde{g}(\theta, \tilde{h}(\theta)) = 0$ in this neighborhood, we have

$$\nabla_\theta \tilde{g}(\theta, \lambda) + \frac{\partial \tilde{g}(\theta, \lambda)}{\partial \lambda} \nabla_\theta \tilde{h}(\theta) = 0$$
$$\Longrightarrow \nabla_\theta \tilde{h}(\theta) = -\frac{\nabla_\theta \tilde{g}(\theta, \lambda)}{\frac{\partial \tilde{g}(\theta, \lambda)}{\partial \lambda}} \tag{4}$$

This gives us the derivative of the implicit function which can be plugged into Eq. (3) to get the final gradients for model parameters θ.

3.3. Updating thresholds

Having performed the gradient update on θ with:

$$\theta^{t+1} = \theta^t - \eta \nabla_\theta \tilde{f}(\theta^t, \tilde{h}(\theta^t)),$$

where $\eta > 0$ is a step-size parameter, what remains is to update the threshold. Again appealing to the Implicit Function Theorem, in the neighborhood around the current iterate θ^t, we can approximate the new threshold as:

$$\begin{aligned} \lambda^{t+1} &= \tilde{h}(\theta^{t+1}) = \tilde{h}(\theta^t + \Delta\theta) \\ &\approx \tilde{h}(\theta^t) + \langle \nabla_\theta \tilde{h}(\theta^t), \Delta\theta \rangle \\ &= \lambda^t + \langle \nabla_\theta \tilde{h}(\theta^t), \Delta\theta \rangle. \end{aligned}$$

As this is an approximation, we employ a *correction step* after every τ minibatch iterations that sets the threshold to satisfy the constraint exactly based on k accumulated minibatches. Note that for all the metrics described in Examples 3–5, this correction step can be performed efficiently using a straight-forward line search.

3.4. Practical improvements

Algorithm 1 outlines our overall approach. In our experiments, we found it effective to use the unrelaxed (and non-

smooth) rates, instead of surrogates, for the correction step, i.e. we set $\lambda^{t+1} = h(\theta^{t+1})$, where h computes the threshold at which the unrelaxed $g(\theta^{t+1}, \lambda^{t+1}) = \mathbf{0}$.

Regularization. In some of our experiments, particularly with smaller UCI datasets, we also impose a *regularizer* on the model parameters that penalizes $\|d\tilde{g}(\theta, \lambda)/d\lambda\|^2$ w.r.t. θ. This encourages the optimization to prefer model parameters for which the constraint function varies smoothly as a function of the threshold. We expect that this will help the model to generalize better on unseen examples.

Optimizing objective with multiple constraints. For optimizing objectives involving multiple constraints (and hence multiple thresholds $\lambda = [\lambda_1, \ldots, \lambda_m] \in \mathbb{R}^m$), such as (partial) PR-AUC or ROC-AUC, a naïve implementation would need multiple gradient computations w.r.t. θ as follows:

$$\nabla_\theta \tilde{f}(\theta, h_1(\theta), \ldots, h_m(\theta_m)) = \nabla_\theta \tilde{f}(\theta, \lambda) - $$
$$\sum_i \frac{\partial \tilde{f}(\theta, \lambda)}{\partial \lambda_i} \frac{\nabla_\theta \tilde{g}(\theta, \lambda_i)}{\frac{\partial \tilde{g}(\theta, \lambda_i)}{\partial \lambda_i}} \tag{5}$$

It needs computation of $\nabla_\theta \tilde{g}(\theta, \lambda_i)$ for all m constraints. We avoid this by first computing the partial derivatives of $\tilde{f}(\theta, \lambda)$ and $\tilde{g}(\theta, \lambda_i)$ w.r.t. λ_i which are much cheaper to compute, and then treating their ratios as constants r_i (akin to using *stop-gradient*). Denoting $h(\theta) = \{h_1(\theta), \ldots, h_m(\theta_m)\}$, we then rewrite the gradient computation as

$$\nabla_\theta \tilde{f}(\theta, h(\theta)) = \nabla_\theta \tilde{f}(\theta, \lambda) - \nabla_\theta \sum_i r_i \tilde{g}(\theta, \lambda_i), \tag{6}$$

which again reduces to just two gradient computations. We also disable the gradient based updates for thresholds to avoid computing separate $\nabla_\theta \tilde{g}(\theta, \lambda_i)$ for all i, and only rely on the correction step of Algorithm 1 every τ minibatches.

4. Related Work

The problem of training models to optimize a given non-decomposable metrics has received much attention in the literature. Early methods on this topic focused on constructing convex surrogates that closely approximate the metric of interest (Joachims, 2005; Yue et al., 2007; Narasimhan & Agarwal, 2013a; Kar et al., 2014; Mohapatra et al., 2014; Narasimhan et al., 2015a), often using structured support vector machines (Tsochantaridis et al., 2005). One of the drawbacks of these approaches is that they are not directly amenable to handling constraints on multiple rates, and may sometimes result in a loose approximation to the metric (Kar et al., 2015). More recent methods seek to directly optimize a given rate metric subject to constraints on multiple rate metrics (Goh et al., 2016; Eban et al., 2017; Narasimhan, 2018; Cotter et al., 2019a;b; Narasimhan et al., 2019a), and

come with scalable gradient-based solvers. There has also been work on construction of structured surrogates (Fathony & Kolter, 2020; Bao & Sugiyama, 2020).

There is also a distinction between methods which handle classification metrics such as the F-measure (Koyejo et al., 2014; Narasimhan et al., 2014; Yan et al., 2018), where often tuning a threshold on a pre-trained class probability model results in a consistent estimator, and those that handle scoring metrics such as the precision-recall and AUC metrics we consider in this paper (Eban et al., 2017), where the focus is on learning a scoring model that performs well at one or more operating thresholds. Other techniques focus on optimizing specialized evaluation metrics that, for example, emphasize good top-k performance in ranking and classification tasks (Agarwal, 2011; Boyd et al., 2012; Fan et al., 2017; Lapin et al., 2017; Hiranandani et al., 2020).

Our approach is most closely related to the method of Eban et al. (2017), who encode the given metric as constraints on classification rates, relax the rates with differentiable surrogates, and perform gradient updates to minimize over the model parameters, and maximize over the Lagrangian multipliers for the constraints. The recent work of Cotter et al. (2019b) and Narasimhan et al. (2019a) improves upon their method by observing that the use of surrogates is *only* required when minimizing over the model parameters, while the maximization over the Lagrange multipliers can be performed with the original unrelaxed rates. The resulting min-max problem can be interpreted as a non-zero-sum game, for which the authors provide efficient gradient-based algorithms to find an equilibrium. This selective use of surrogates is incorporated into our proposal: we use surrogates only when we need to compute gradients for the objective $\tilde{f}(\theta, \tilde{h}(\theta))$, whereas, as we mentioned in Section 3.4, we use the original unrelaxed rates while computing a correction for the threshold $\lambda = h(\theta)$.

Unlike these earlier papers, our method avoids explicitly solving a constrained optimization problem, and instead expresses the threshold as an implicit function of the model parameters. This proposal is similar in flavor to the approach taken by Mackey et al. (2018), who like us formulate an unconstrained objective, but do so by expressing the threshold as a quantile of the model scores.

Finally, the growing literature on fairness in machine learning has opened the door for many new applications for constrained optimization (Hardt et al., 2016; Agarwal et al., 2018), introducing many group-based fairness metrics that can be easily handled using the proposed approach.

5. Experiments

We evaluate our approach on five UCI classifications tasks, and two image classification tasks. A summary of the

Table 1. Summary of datasets.

Datasets	#Examples	#Features
CelebA	202,599	$32 \times 32 \times 3$
BigEarthNet	590,326	$40 \times 40 \times 3$
Letter	19,999	16
IJCNN1	49990	22
Adult	48,842	122
Spambase	4,601	57
Com. & Crime	1,994	145

datasets used in the main text is provided in Table 1. We present additional experimental results in Appendices B, C and D.

Baselines. We compare the proposed ICO approach with the state-of-the-art tools of Cotter et al. (2019a) and Narasimhan et al. (2019a) for constrained optimization, open-sourced as a part of the TensorFlow Constrained Optimization Library (TFCO)[2]. The prior technique of Eban et al. (2017) can be viewed as a special case of the functionality provided by this library. We also compare to the baseline approach of optimizing a standard cross-entropy (CE) loss.

Experimental protocol. For our experiments with Image datasets, we use a 6-layer neural network with 5 convolutional layers with 128, 256, 256, 512 and 512 filters respectively. We use ReLU activation functions and batch normalization layers in the network. We use a separate validation split for model selection in all our experiments. For all three methods (CE, TFCO, and ICO), we use the evaluation metric of interest on the validation set for model selection (*e.g.*, FNR, ROC-AUC, PR-AUC). This makes the CE baseline further strong in our experiment. We do 5 random trials for each experiment and report the average value of the metric. Other details such as standard deviations for the random trials are reported in the Appendix.

5.1. Minimizing FNR at FPR β

First, we consider the task of minimizing false negative rate (FNR) at a given false positive rate (FPR) of β. This setting is particularly relevant for security sensitive applications where one wants to operate at a desired false positive rate. We experiment with the publicly available CelebA dataset (Liu et al., 2015), which contains 202,599 celebrity face images of size $32 \times 32 \times 3$. CelebA has 40 annotated binary attributes for every face image, of which we randomly choose 8 for our experiments. We use the standard train,

---

[2]https://github.com/google-research/tensorflow_constrained_optimization

Table 2. Minimizing false negative rate (FNR) at a given false positive rate (FPR) for **CelebA**. The mean FNR (in %) are reported over five random trials for cross-entropy/ TFCO/ ICO, respectively. Proposed ICO outperforms both CE and TFCO by a considerable margin. We report results on more attributes along with the std. errors in Appendix C. *Lower* values are better.

FPR	High-cheekbones	Heavy-makeup	Wearing-lipstick	Smiling	Black-hair	Blond-hair
1%	53.5/ 49.0/ **46.9**	57.0/ 57.0/ **49.6**	44.0/ 42.6/ **37.5**	37.4/ 35.9/ **33.7**	69.3/ 64.4/ **63.2**	40.4/ 38.6/ **36.8**
2%	44.8/ 40.9/ **39.8**	45.6/ 41.2/ **38.9**	32.7/ 30.4/ **26.7**	29.4/ 27.8/ **26.1**	56.4/ 52.0/ **50.5**	28.9/ 25.6/ **24.2**
5%	32.9/ 30.1/ **28.5**	28.2/ 25.4/ **23.1**	16.3/ 14.9/ **13.1**	18.7/ 17.0/ **16.9**	36.7/ 32.4/ **32.5**	13.4/ 11.6/ **10.8**
10%	22.9/ 20.4/ **19.7**	15.1/ 13.6/ **12.4**	6.6/ 5.9/ **4.7**	11.7/ 10.7/ **10.2**	23.0/ 19.2/ **18.6**	6.5/ 4.9/ **4.7**

Table 3. Maximizing area under the ROC curve for **CelebA**, in a given FPR range $[0, \beta]$ for $\beta \in \{1\%, 2\%, 5\%, 10\%, 20\%\}$. The mean ROC-AUC are reported over five random trials for cross-entropy/ TFCO/ ICO, respectively. Last column shows the mean partial AUC over all 8 attributes. We report results on more attributes along with the std. errors in Appendix C. *Higher* values are better.

β	High-cheekbones	Heavy-makeup	Wearing-lipstick	Smiling	Black-hair	Mean
1%	66.1/ 62.9/ **69.8**	65.0/ **68.5**/ 66.7	70.2/ **74.8**/ 72.3	75.4/ **78.0**/ 75.6	60.5/ **61.4**/ 61.2	64.7/ 65.9/ **66.1**
2%	70.8/ **74.9**/ 73.2	68.8/ **73.4**/ 71.6	75.4/ **79.3**/ 78.4	78.4/ **81.5**/ 79.8	64.5/ **66.5**/ 66.1	68.6/ **70.8**/ 70.5
5%	75.9/ 73.8/ **78.5**	75.5/ **79.5**/ 78.3	82.2/ 84.7/ 84.6	83.5/ 65.0/ **84.8**	70.9/ 73.2/ **73.8**	74.7/ 72.8/ **76.8**
10%	80.1/ 74.1/ **82.7**	81.5/ **85.0**/ 84.4	87.7/ 89.3/ **89.7**	86.7/ 73.8/ **88.8**	78.0/ 79.9/ **80.2**	79.7/ 77.9/ **81.8**
20%	84.2/ 72.4/ **86.8**	88.2/ 89.9/ 89.8	91.8/ 93.1/ **93.9**	90.9/ 76.6/ **92.1**	84.3/ 86.0/ 86.1	84.9/ 82.4/ **86.8**

validation, and test splits[3] for CelebA, and train a binary classifier for each attribute.

We use TFCO and ICO for optimizing FNR at four different FPR targets: 1%, 2%, 5% and 10%. For the cross-entropy baseline, we use Adam optimizer (Kingma & Ba, 2014) with a learning rate of 0.001. For TFCO, we use Adam for both the primal and dual updates; the primal learning rate is set to 0.001, while the dual learning rate is chosen from $\{0.1, 0.01\}$ using the validation sample. We use the cross-entropy surrogate (softplus function for binary case) provided by the TFCO library to approximate the rates. For ICO, we again use Adam with a learning rate of 0.001. We approximate the rates for ICO with temperature-scaled sigmoid surrogates, and choose the temperature parameter from the range $\{0.001, 0.01, 0.1, 1.0\}$ using the validation sample. We do not apply gradient regularization, and perform the correction step described in Section 3.4 once every 1000 mini-batch updates using data from next 10 mini-batches. All optimizers use a batch size of 512.

We present in Table 2 the test evaluation metrics. The proposed ICO method performs the best for all six prediction tasks and for all four FPR targets, with TFCO coming in second. Interestingly, the gap between ICO and the other methods is the larger for smaller FPR targets. This suggests that ICO is most advantageous when applied to metrics that are harder to optimize. Unsurprisingly, cross-entropy optimization often yields higher FNR values at the specified

Table 4. Maximizing area under the ROC curve for **BigEarthNet**, in a given FPR range $[0, \beta]$ for $\beta \in \{5\%, 10\%, 20\%\}$. The mean ROC-AUC are reported over five random trials for cross-entropy/ TFCO/ ICO, respectively. We report results on more attributes along with the std. errors in Appendix D. *Higher* values are better.

Labels	β		
	5%	10%	20%
BLF	66.2/ 66.4/ **69.9**	71.0/ 71.7/ **73.9**	75.4/ 76.2/ **77.9**
CC	62.2/ 62.7/ **63.6**	67.4/ 66.8/ **68.3**	73.8/ **76.0**/ 74.9
CF	71.9/ 71.5/ **74.7**	78.6/ 80.0/ **80.7**	84.8/ **86.6**/ 86.0
DUF	69.8/ 71.8/ **73.9**	75.1/ 77.2/ **78.0**	78.8/ 81.4/ **81.8**
ANV	58.8/ 58.8/ **60.4**	62.7/ 63.9/ **64.4**	67.8/ 69.1/ 69.0
Mean	65.8/ 66.2/ **68.5**	71.0/ 71.9/ **73.1**	76.1/ 77.9/ 77.9

FPR targets than the other methods, indicating that methods which directly optimize performance at the desired FPR do end up performing better at that target. Results on other attributes are reported in Appendix C. We also report timing comparisons of ICO and TFCO in Appendix B.

5.2. Maximizing ROC-AUC in FPR range $[0, \beta]$

Next, we consider the task of maximizing the (partial) area under the ROC curve in a select range $[0, \beta]$ of FPRs. This metric is used in medical diagnostic tasks and biometric screening (Rao et al., 2008; Ricamato & Tortorella, 2011), where optimizing performance in a relevant FPR range may
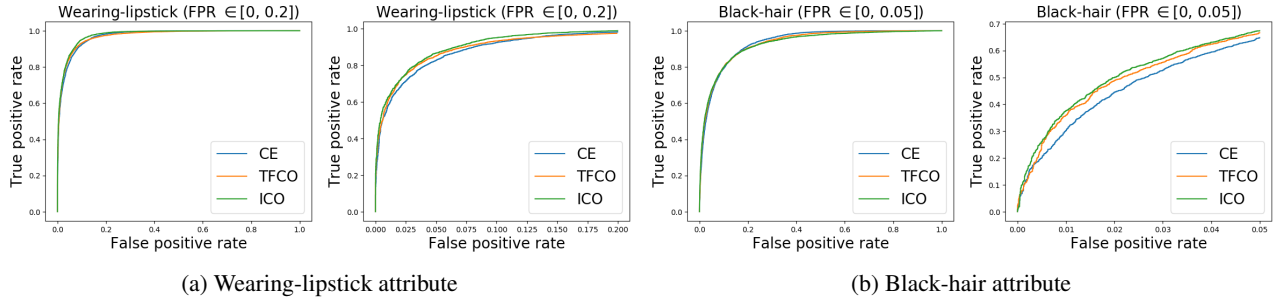
(a) Wearing-lipstick attribute

(b) Black-hair attribute

Figure 1. ROC curves for CelebA: **(a)** For attribute *Wearing-lipstick* and optimizing for partial area under the ROC curve with FPR $\in [0, 0.2]$, **(b):** For attribute *Black-hair* and optimizing for partial area under the ROC curve with FPR $\in [0, 0.05]$. Left figures show full ROC curves while the right figures show the (zoomed-in) ROC curves in the respective target FPR ranges.
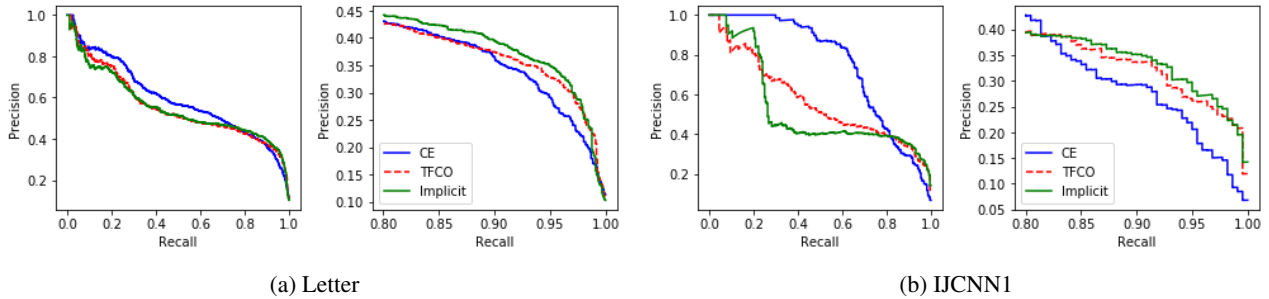

(a) Letter

(b) IJCNN1

Figure 2. Precision-Recall curves on the test set. Both TFCO and the proposed method seek to optimize PR-AUC in the recall range $[0.95, 1]$. The left plots for each dataset show the entire curve, while the right plots zoom in on the right-end of the curve.

prove critical. We also compare with a pairwise loss baseline (Narasimhan & Agarwal, 2013b) which optimizes the objective $\frac{1}{N^+|S^-|}\sum_{i:y_i=1}\sum_{j\in S^-}\tilde{f}(s^\theta(x_i) - s^\theta(x_j))$, where $s^\theta(x)$ denotes the score (*e.g.*, logits) for example x, N^+ is the number of positive examples in the minibatch, S^- is the subset of negative examples whose scores lie in the top β fraction of all negative examples, and $\tilde{f}$ is the surrogate used for 0-1 loss (either softplus or sigmoid with a temperature hyperparameter as we used for the proposed method). We use the pairwise-loss, TFCO, and the proposed method to optimize this metric for five different value of β: 1%, 2%, 5%, 10% and 20%. The results for the pairwise loss baseline are reported in the supplementary material. We experiment with CelebA and BigEarthNet (Sumbul et al., 2019) image datasets. BigEarthNet contains 590,326 Sentinel-2 image patches of size $120 \times 120 \times 3$ in RGB, which we downsize to $40 \times 40 \times 3$, and 43 annotated binary labels, of which we choose 5 for our experiments. These lables are Broad-Leaved Forest (BLF), Complex Cultivation patterns (CC), Coniferous Forest (CF), Discontinuous Urban Fabric (DUF) and Agricultural with Natural Vegetation land (ANV). We split the dataset randomly into 70% for training, 15% for validation and 15% testing.

For both datasets, we train the same convolutional neural network model as the previous experiment. Both TFCO and our method divide the specified FPR range $[0, \beta]$ into

10 equally-spaced values, and optimize the average true positive rate (TPR) at those targets. We replicate the same parameter configurations used in the previous experiment, except that the update frequency for ICO is performed either once in every 100 updates or 1000 updates, based on which of the two choices yields the highest validation ROC-AUC. We do not use gradient regularization in this experiment.

We present in Table 3 the test evaluation metrics for different methods on CelebA, where we applied TFCO and ICO to optimize the partial ROC-AUC metric for five different values of β: 1%, 2%, 5%, 10% and 20%. We apply the standard McClish correction (McClish, 1989) to rescale the area between 0 and 100. On at least half the classification tasks, the proposed ICO method performs the best, with TFCO coming in second. On average across all six image attributes, ICO is considerably better than TFCO on three of the five values of FPR targets β. We also show the ROC plots for a few specific cases in Figure 1. We present the results for BigEarthNet in Table 4, where we experiment with three values of β: 5%, 10%, 20%. For all five labels, the proposed ICO is seen to perform better than the baselines for the smaller false-positive ranges, i.e. for smaller β, thus demonstrating its effectiveness in optimizing performance in the initial portion of the ROC curve for this dataset. We also report timing comparisons of ICO and TFCO in Appendix B, observing that ICO can converge faster than TFCO in

Table 5. Maximizing (partial) PR-AUC in the recall range [0.95, 1] on **UCI** datasets. Proposed ICO performs better than the other methods on datasets with severe class imbalance. *Higher* values are better.

	%Positives	Cross-entropy	TFCO	ICO
Letter	4%	15.13 ± 0.86	20.49 ± 0.43	$\mathbf{23.04 \pm 0.77}$
IJCNN1	9.7%	21.18 ± 0.33	26.14 ± 0.57	$\mathbf{27.28 \pm 0.58}$
Adult	24%	39.74 ± 0.29	40.21 ± 0.38	$\mathbf{40.34 \pm 0.41}$
Spam	39%	71.51 ± 1.73	73.08 ± 1.70	$\mathbf{73.48 \pm 1.80}$
Com. & Crime	30%	47.00 ± 0.94	47.04 ± 0.97	47.03 ± 1.08

terms of wall-clock time.

5.3. Maximizing PR-AUC in Recall Range $[\beta, 1]$

In our final set of experiments, we consider the task of maximizing the (partial) area under the Precision-Recall curve in a select range of recall values $[\beta, 1]$. This metric is relevant in retrieval applications, where the quality of the system is often evaluated at multiple recall targets. We experiment with the five smaller datasets in Table 1 obtained from the UCI repository (Frank & Asuncion, 2010). For the Letter dataset, we treat the most frequent letter as the positive example, and the rest as negative. For the Communities & Crime dataset, we seek to predict if a community in the US has a crime rate above the 70th percentile (Kearns et al., 2018). We trained a linear model in each case. We split the datasets into train, validation and test datasets in the ratios 50%:25%:25%.

Both TFCO and ICO divide the specified recall range $[\beta, 1]$ into five equally-spaced values, and optimize the average precision at those targets. We use Adam for the cross-entropy baseline and for TFCO, and Adagrad for the proposed ICO. For cross-entropy optimization, we tune the learning rate from the range $\{10^{-3}, 10^{-2}, 10^{-1}, 1\}$, picking the one with maximum PR-AUC metric on the validation sample. For TFCO, we tune the learning rate and dual scale parameters from the range $\{0.01, 0.1, 1.0\}$ and $\{0.1, 1.0, 10.0\}$ respectively. We approximated the rates for TFCO using the cross-entropy surrogate loss provided by the library. For the proposed ICO, we use a fixed learning rate of 0.1, and approximate the rates with a temperature-scaled sigmoid surrogates, with the temperature parameter for the surrogate chosen from $\{0.5, 1.0, 5.0\}$. We also apply the gradient regularizer described in Section 3.4, with the regularization strength parameter chosen from the range $\{0, 0.05, 0.1\}$. We perform the correction step once every 10 updates. All optimizers perform full gradient updates.

We first evaluate the performance of different methods at very high recall values. For this, we run both TFCO and ICO to optimize average precision in the recall range $[0.95, 1]$. Table 5 presents the test PR-AUC metric values in the range $[0.95, 1]$ for the different methods. We find that on

the Letter and IJCNN1 datasets, which have severe class imbalance, the proposed approach performs significantly better than the two baselines. On these datasets, cross-entropy optimization performs poorly. On the datasets where the classes are reasonably balanced, all three baselines perform similarly. On the Communities & Crime dataset, we find our approach yielding better metric values than the other methods on the training sample, but because of the small data size, does not generalize as well to the test set.

In Figure 2, we show the Precision-Recall cuves for the different methods for the the Letter and IJCNN1 datasets. Notice that while cross-entropy optimization yields higher precision for lower recall values, it does not fair well in the recall range that matters $[0.95, 1]$. Clearly, there is notable benefit to directly optimizing for the recall range that we care about instead of using an off-the-shelf loss function. Moreover, the proposed ICO method outperforms TFCO at high recall values. We also apply the methods to optimize PR-AUC in recall ranges $[\beta, 1]$, for varying values of β. In the results shown in Figure 3 for the Letter and IJCNN1 datasets, one can see that the benefit offered by ICO is the most benefit over the two baselines is most notable for higher β values.

6. Discussion and Future Work

We proposed an approach for optimizing popular constrained optimization problems arising in machine learning that involve non-decomposable rate metrics, such as false positive rate, true positive rate, areas under the precision-recall or ROC curves, *etc.*. Our approach deviates significantly from the existing methods based on Lagrange multipliers, and uses Implicit Function Theorem to express the classifier thresholds as a function of model parameters. Our experiments showed considerable improvements on optimizing several common evaluation metrics (such as FNR at a fixed FPR, areas under (partial) precision-recall and ROC curves) over existing state-of-the-art methods (Cotter et al., 2019a; Narasimhan et al., 2019a) that are part of open-source tools. In this work, we primarily focused on the straightforward setting where the classifier thresholds are expressed as a function of *all* model parameters but it is also
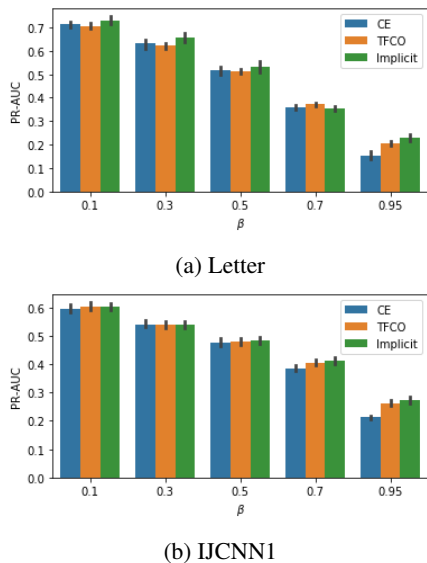
(a) Letter

(b) IJCNN1

Figure 3. Plot of PR-AUC in $[\beta, 1]$ as a function of β on the Letter and IJCNN1 datasets. The proposed method is often advantageous for very high β values. *Higher* values are better.

possible to consider an alternative setting where the thresholds are expressed as a function of only *a subset* of model parameters (*e.g.*, last few layers of the neural network). We leave this as a direction for future work.

We close by highlighting how our proposal can be extended to handle more complex settings and constraints.

6.1. Inequality constraints

All the constrained metrics we described are defined as equality constraints. In our current implementation, we handle inequality constraints by searching for thresholds that satisfy the original non-smooth inequality constraints (during the correction step in Algorithm 1 and finally at the end of training). However, we can easily extend our proposal to handle inequality constraints of the form $\tilde{g}(\theta, \lambda) \leq \mathbf{0}$ in a more principled manner. In this case, one can introduce m auxiliary slack variables $\xi_1, \ldots, \xi_m \in \mathbb{R}_+$, and rewrite the inequality-constrained problem as one with equality constraints:

$$\min_{\theta \in \mathbb{R}^p, \xi \in \mathbb{R}_+^m} \tilde{f}(\theta, \lambda) \quad \text{s.t. } \tilde{g}(\theta, \lambda) + \xi = \mathbf{0}. \quad (7)$$

We can now apply Algorithm 1 to solve the re-written problem by treating the model parameters and auxiliary variables $(\theta, \xi) \in \mathbb{R}^p \times \mathbb{R}_+^m$ together as the optimization variables, with an additional projection step in the gradient descent procedure to ensure non-negativity of the ξ_is. We did not experiment with this version of the algorithm for the sake of implementation simplicity.

6.2. Multi-class metrics

In our experiments, we handled binary classification tasks. The extension to multi-class metrics requires some effort as in this case we are allowed to predict only one among m labels. As with the binary classification setting, we work with a model $s^\theta : X \to \mathbb{R}^m$ that outputs m scores, and we will maintain one parameter λ_i for each class $i \in [m]$. The parameters λ_i's would then be used to post-shift the model via a *weighted* or *shifted* argmax to predict the final class:

$$s_\lambda^\theta(x) \in \arg\max_{i \in [m]} (s_i^\theta(x) - \lambda_i).$$

Computing the parameters λs so that the resulting classifier satisfies the specified constraints is not straightforward, but can still be performed efficiently with, *e.g.*, the methods in Narasimhan et al. (2015b). While it isn't clear if a feasible λ exists for general rate constraints, it certainly does for constraints like "coverage" (Cotter et al., 2019a), which require that the model makes a certain percentage of predictions from each class.

6.3. Ranking metrics

Perhaps, the most interesting extension of our approach is to query-based ranking problems (Schütze et al., 2008), where each example contains a query and a list of documents, and the goal is to rank the documents based on the relevance to the query. Popular ranking metrics such as Precision@K or Recall@K seek to measure performance in the top ranked documents (Lapin et al., 2017). Unfortunately, writing these metrics out as an explicit constrained optimization problem would require one constraint per query, with the number of constraints growing with the size of the training set. Consequently, standard constrained optimization approaches, when applied to optimize these metrics, would need to maintain one Lagrange multiplier for each query, making it impractical to use them with large datasets. Recently, (Narasimhan et al., 2019b) propose solving such heavily-constrained problems with lower-dimensional representations of Lagrange multipliers. In contrast, our method offers an alternate route which does not require explicitly handling the large number of constraints, through implicit modeling of the per-query thresholds. We look forward to future work comparing our implicit thresholding approach with the state-of-the-art methods for these ranking metrics (e.g. Kar et al. (2015); Lapin et al. (2017)).

References

Agarwal, A., Beygelzimer, A., Dudik, M., Langford, J., and Wallach, H. A reductions approach to fair classification. In *International Conference on Machine Learning*, pp. 60–69, 2018.

Agarwal, S. The infinite push: A new support vector ranking

algorithm that directly optimizes accuracy at the absolute top of the list. In *Proceedings of the 2011 SIAM International Conference on Data Mining*, pp. 839–850. SIAM, 2011.

Bao, H. and Sugiyama, M. Calibrated surrogate maximization of linear-fractional utility in binary classification. In *International Conference on Artificial Intelligence and Statistics*, pp. 2337–2347. PMLR, 2020.

Biddle, D. *Adverse impact and test validation: A practitioner's guide to valid and defensible employment testing*. Gower Publishing, Ltd., 2006.

Boyd, S., Cortes, C., Mohri, M., and Radovanovic, A. Accuracy at the top. In *Advances in Neural Information Processing Systems*, 2012.

Cotter, A., Jiang, H., Gupta, M. R., Wang, S., Narayan, T., You, S., and Sridharan, K. Optimization with non-differentiable constraints with applications to fairness, recall, churn, and other goals. *Journal of Machine Learning Research*, 20(172):1–59, 2019a.

Cotter, A., Jiang, H., and Sridharan, K. Two-player games for efficient non-convex constrained optimization. In *Algorithmic Learning Theory*, pp. 300–332. PMLR, 2019b.

Eban, E., Schain, M., Mackey, A., Gordon, A., Rifkin, R., and Elidan, G. Scalable learning of non-decomposable objectives. In *Artificial intelligence and statistics*, pp. 832–840. PMLR, 2017.

Fan, Y., Lyu, S., Ying, Y., and Hu, B.-G. Learning with average top-k loss. *arXiv preprint arXiv:1705.08826*, 2017.

Fathony, R. and Kolter, Z. Ap-perf: Incorporating generic performance metrics in differentiable learning. In *International Conference on Artificial Intelligence and Statistics*, pp. 4130–4140. PMLR, 2020.

Frank, A. and Asuncion, A. UCI machine learning repository. URL: http://archive.ics.uci.edu/ml, 2010.

Goh, G., Cotter, A., Gupta, M., and Friedlander, M. P. Satisfying real-world goals with dataset constraints. In *Advances in Neural Information Processing Systems*, pp. 2415–2423, 2016.

Hardt, M., Price, E., and Srebro, N. Equality of opportunity in supervised learning. In *Advances in neural information processing systems*, pp. 3315–3323, 2016.

Hiranandani, G., Vijitbenjaronk, W., Koyejo, S., and Jain, P. Optimization and analysis of the pap@ k metric for recommender systems. In *International Conference on Machine Learning*, pp. 4260–4270. PMLR, 2020.

Joachims, T. A support vector method for multivariate performance measures. In *Proceedings of the 22nd international conference on Machine learning*, pp. 377–384. ACM, 2005.

Kar, P., Narasimhan, H., and Jain, P. Online and stochastic gradient methods for non-decomposable loss functions. *arXiv preprint arXiv:1410.6776*, 2014.

Kar, P., Narasimhan, H., and Jain, P. Surrogate functions for maximizing precision at the top. In *International Conference on Machine Learning*, pp. 189–198. PMLR, 2015.

Kearns, M., Neel, S., Roth, A., and Wu, Z. Preventing fairness gerrymandering: Auditing and learning for subgroup fairness. In *ICML*, 2018.

Kingma, D. P. and Ba, J. Adam: A method for stochastic optimization. *ICLR*, 2014.

Koyejo, O. O., Natarajan, N., Ravikumar, P. K., and Dhillon, I. S. Consistent binary classification with generalized performance metrics. In *NIPS*, pp. 2744–2752, 2014.

Lapin, M., Hein, M., and Schiele, B. Analysis and optimization of loss functions for multiclass, top-k, and multilabel classification. *IEEE transactions on pattern analysis and machine intelligence*, 40(7):1533–1554, 2017.

Liu, Z., Luo, P., Wang, X., and Tang, X. Deep learning face attributes in the wild. In *Proceedings of the IEEE International Conference on Computer Vision*, pp. 3730–3738, 2015.

Mackey, A., Luo, X., and Eban, E. Constrained classification and ranking via quantiles. *arXiv preprint arXiv:1803.00067*, 2018.

McClish, D. K. Analyzing a portion of the roc curve. *Medical Decision Making*, 9(3):190–195, 1989.

Mohapatra, P., Jawahar, C., and Kumar, M. P. Efficient optimization for average precision SVM. In *NIPS-Advances in Neural Information Processing Systems*, 2014.

Narasimhan, H. Learning with complex loss functions and constraints. In *International Conference on Artificial Intelligence and Statistics*, pp. 1646–1654, 2018.

Narasimhan, H. and Agarwal, S. A structural svm based approach for optimizing partial auc. In *International Conference on Machine Learning*, pp. 516–524. PMLR, 2013a.

Narasimhan, H. and Agarwal, S. Svmpauctight: a new support vector method for optimizing partial auc based on a tight convex upper bound. In *Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 167–175, 2013b.

Narasimhan, H., Vaish, R., and Agarwal, S. On the statistical consistency of plug-in classifiers for non-decomposable performance measures. In *Advances in Neural Information Processing Systems*, pp. 1493–1501, 2014.

Narasimhan, H., Kar, P., and Jain, P. Optimizing non-decomposable performance measures: A tale of two classes. In *International Conference on Machine Learning*, pp. 199–208. PMLR, 2015a.

Narasimhan, H., Ramaswamy, H., Saha, A., and Agarwal, S. Consistent multiclass algorithms for complex performance measures. In *ICML*, pp. 2398–2407, 2015b.

Narasimhan, H., Cotter, A., and Gupta, M. Optimizing generalized rate metrics with three players. In *Advances in Neural Information Processing Systems*, pp. 10747–10758, 2019a.

Narasimhan, H., Cotter, A., Zhou, Y., Wang, S., and Guo, W. Approximate heavily-constrained learning with lagrange multiplier models. In *Advances in Neural Information Processing Systems*, pp. 10747–10758, 2019b.

Rao, R. B., Yakhnenko, O., and Krishnapuram, B. Kdd cup 2008 and the workshop on mining medical data. *ACM SIGKDD Explorations Newsletter*, 10(2):34–38, 2008.

Ricamato, M. T. and Tortorella, F. Partial auc maximization in a linear combination of dichotomizers. *Pattern Recognition*, 44(10-11):2669–2677, 2011.

Schütze, H., Manning, C. D., and Raghavan, P. *Introduction to information retrieval*, volume 39. Cambridge University Press Cambridge, 2008.

Sumbul, G., Charfuelan, M., Demir, B., and Markl, V. Bigearthnet: A large-scale benchmark archive for remote sensing image understanding. In *IGARSS 2019-2019 IEEE International Geoscience and Remote Sensing Symposium*, pp. 5901–5904. IEEE, 2019.

Tsochantaridis, I., Joachims, T., Hofmann, T., Altun, Y., and Singer, Y. Large margin methods for structured and interdependent output variables. *Journal of machine learning research*, 6(9), 2005.

Tu, L. W. An introduction to manifolds. second, 2011.

Wurker, U. Convexity properties of some implicit functions. *Journal of Convex Analysis*, 2001.

Yan, B., Koyejo, S., Zhong, K., and Ravikumar, P. Binary classification with karmic, threshold-quasi-concave metrics. In *International Conference on Machine Learning*, pp. 5531–5540. PMLR, 2018.

Yue, Y., Finley, T., Radlinski, F., and Joachims, T. A support vector method for optimizing average precision. In *Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval*, pp. 271–278, 2007.

Zafar, M. B., Valera, I., Rogriguez, M. G., and Gummadi, K. P. Fairness constraints: Mechanisms for fair classification. In *Artificial Intelligence and Statistics*, pp. 962–970. PMLR, 2017.