Discriminative Learning of Prediction Intervals

Nir Rosenfeld
Harvard University
and Microsoft Research

Yishay Mansour
Tel Aviv University

Elad Yom-Tov
Microsoft Research

Abstract

In this work we consider the task of constructing prediction intervals in an inductive batch setting. We present a discriminative learning framework which optimizes the *expected* error rate under a budget constraint on the interval sizes. Most current methods for constructing prediction intervals offer guarantees for a single new test point. Applying these methods to multiple test points results in a high computational overhead and degraded statistical properties. By focusing on expected errors, our method allows for variability in the per-example conditional error rates. As we demonstrate both analytically and empirically, this flexibility can increase the overall accuracy, or alternatively, reduce the average interval size.

While the problem we consider is of a regressive flavor, the loss we use is combinatorial. This allows us to provide PAC-style, finite-sample guarantees. Computationally, we show that our original objective is NP-hard, and suggest a tractable convex surrogate. We conclude with a series of experimental evaluations.

1 Introduction

Constructing an interval which contains some point of interest with high probability is a fundamental task in statistics. In contrast to point predictions, intervals provide some measure of confidence, and can be used to reason about the reliability of a prediction. In this paper we focus on *prediction intervals* (PIs). For some predetermined *significance level* α, the classic task of PI estimation is to construct an interval $[\ell, u]$ which will contain a point $y \in \mathbb{R}$ sampled from some distribution D with probability of at least $1 - \alpha$. PIs are hence similar in spirit to *confidence intervals*, but are designed to contain future sampled points rather than population parameters such as the mean or variance.

In this paper we consider non-parametric PI estimation in a regressional setting. Let $D = D_{XY}$ be an unknown joint distribution over examples $x \in \mathcal{X}$ and labels $y \in \mathcal{Y} = \mathbb{R}$, where X, Y denote random variables and x, y their instantiations. In the classic PI task, we are given a training set $S = \{(x_i, y_i)\}_{i=1}^m$ of m pairs sampled i.i.d. from D_{XY}, and an additional test example x_{m+1} sampled from the marginal D_X. Then, for a given confidence level α, the goal is to construct an interval $[\ell, u] \in \mathbb{R}^2$ which will contain the true label $y_{m+1} \sim D_{Y|X=x}$ with probability of at least $1 - \alpha$. The classic approach to this task is to first estimate the conditional $D_{Y|X=x_{m+1}}$, and then infer the smallest interval which covers $1 - \alpha$ of the probability mass. In the case of a Gaussian conditional distribution, this leads to simple closed form solution for computing PIs, and provides asymptotic guarantees when the data is indeed Gaussian (see also Sec. 2). There are many variations on this parametric two-step approach [10, 11], as well as other direct and non-parametric methods [24, 9, 23].

Methods such as the above are designed for the classic setting which concerns only a single new test example. In many realistic cases, however, the task is to predict intervals for a *set* of future test points, given only the training data. This setting is known as *inductive batch learning* and has been extensively studied in machine learning. As we later discuss in detail, applying single test-point methods to the batch setting is rarely straightforward, and often comes at the cost of a significant computational overhead and/or the loss of statistical guarantees [14, 15, 3]. This is also true for the recently popular methods based on conformal prediction designed in the online learning setting [2, 13].

Our goal in this paper is therefore to provide a general framework for PI estimation in the batch setting.

The approach we present models PI estimation as a discriminative learning task, where the goal is to learn an interval predictor with high expected accuracy. To this end, we design a learning objective which takes into account the natural tradeoff between accuracy and interval size. Our analysis provides computational guarantees on training such predictors, as well as statistical guarantees on their generalization. This means that, with enough samples, an interval predictor which achieves an error of α on the training set will have roughly the same error on the test set.

The method we present differs from the standard single-point approach in several important ways. First, the learning objective we consider relaxes the classic α-confidence requirement to hold *in expectation*. Focusing on expected errors allows our method to predict intervals with different error rates for different examples. Second, a model-free discriminative framework separates the task (predicting accurate intervals) from the model (Gaussian, linear, etc.). To this end, we first identify an appropriate combinatorial loss function, and then suggest a tractable convex surrogate. This allows for trading off guarantees on optimization (in the linear case) with potentially higher accuracy (via more expressive predictor classes). Third, our approach reverses the classic roles of accuracy and interval size: instead of fixing an error rate of α and predicting tight intervals, we seek to minimize the error under a fixed *budget* constraint on the mean interval size. Intuitively, allowing for variability in individual interval sizes enables the overall accuracy to be boosted by "sacrificing" some points for the sake of others. Overall, the formulation we present is geared towards maximizing expected accuracy, rather than providing (asymptotic) worst-case guarantees.

For the analysis, we partition the significance level α into a confidence term δ (over $S \sim D_{XY}^m$) and an accuracy term ϵ (over $y \sim D_{Y|X}$), thus promoting PAC-style results. Fixing a base class $\mathcal{B}$ of real functions, we show how the VC dimension of interval predictors based on $\mathcal{B}$ can be expressed in terms of the VC dimension of thresholds over $\mathcal{B}$. This covers several well-established classes of binary classifiers. Our results show that the VC of intervals grows only linearly in that of the threshold class. While classic methods typically offer only algorithm-specific asymptotic guarantees, our framework provides finite-sample results that depend only on the function class.

The paper is structured as follows. In Sec. 2 we review related work on prediction intervals. We then compare the single-point and batch settings in Sec. 3, and present our discriminative method in Sec. 4. Sec. 5 contains a theoretical analysis of both the statistical complexity (Sec. 5.1) and computational hardness (Sec. 5.2) of our approach. Finally, Sec. 6 contains a detailed experimental evaluation of our method on a collection of benchmark datasets. We conclude with a discussion of our results in Sec. 7.

2 Related Work

The simplest approach for constructing PIs includes two steps. In the first step, as in standard regression, a point-predictor $f : \mathcal{X} \to \mathbb{R}$ is trained. Then, intervals $[\ell, u]$ are constructed around the predictions $\hat{y} = f(x)$. In the parametric setting, the conditional distribution $D_{\mathcal{Y}|\mathcal{X}}$ is assumed to have some parametric form (for a comprehensive review, see [6]). Given a new example x, the interval boundaries ℓ and u are then set to contain $1 - \alpha$ of the probability mass. Many parametric forms of $D_{\mathcal{Y}|\mathcal{X}}$ lead to closed form solutions for $[\ell, u]$. For example, assuming $y \sim \mathcal{N}(\mu(x), 1)$ gives:

$$[\ell, u] = \hat{\mu}(x) \pm \Phi^{-1}(\alpha/2)\sqrt{\frac{m+1}{m}} \qquad (1)$$

where $\Phi^{-1}(\cdot)$ is the inverse normal cdf, and the conditional mean estimates $\hat{y} = \hat{\mu}(x)$ can be learned via simple regression [20].

In Eq. (1), the interval boundaries are in effect set to be the $\alpha/2$ and $1 - \alpha/2$ quantiles of the (assumed) conditional distribution. This suggests that the above two-stage process can be replaced with a direct estimation of the conditional quantiles $q_\tau(x)$. Quantile regression can then be used to predict PIs [9], circumventing the need to explicitly state a distributional assumption. This can either be done by minimizing a tilted version of the absolute loss over a parametric hypothesis class [10, 11], or by using non-parametric empirical quantile estimates using bootstrapping [24] or leave-one-out sampling [23].

Methods such as the above are often justified by asymptotic guarantees. These, however, do not always lead to good performance in practice. Some methods offer corrections for finite-sample biases [19, 17], but are often based on further assumptions, and general sample complexity results are typically hard to obtain.

In a set of comprehensive papers on *conformal prediction*, the authors expand the classic single test-point setting to an online setting where a set of new examples are given in sequence [5, 21]. The goal is then to minimize regret, namely predict intervals which will be good in hindsight compared to the best fixed alternative. Based on this, recent work [12] introduces flexible extensions to the basic conformal method, as well as additional finite-sample and distribution-free asymptotic guarantees. While several works on conformal prediction discuss the batch setting [2, 14, 15], the implicit goal in these is still to achieve a per-example

error of at most α. This typically induces a large computational overhead in the inductive setting, since every new point effectively requires fitting a new model. The split-conformal method in [13] fits a single model, but at the cost of larger intervals [3]. Our method, in contrast, focuses on an inductive batch setting where the goal is to minimize the expected error, and a predictor is trained only once on a fixed training set.

Several works consider a learning setting that is similar to ours, most of which are motivated by specific applications. In [8], the authors design a multi-criteria loss and train a neural network to explicitly predict the interval boundaries. Several extensions of this approach have been suggested as well [25, 7]. In contrast to our method, the above methods balance accuracy and interval size heuristically, use a discontinuous (and non-convex) objective, offer no statistical guarantees, and are applied only to small datasets.

3 Single-point vs. Batch

Before presenting our method, it will be useful to formally discuss the subtle but crucial distinctions between the single test-point and inductive batch settings. Let $f : \mathcal{X} \to \mathbb{R} \times \mathbb{R}$ be an interval predictor. In the single test-point setting (as well as in the online learning setting used in conformal prediction [21]), the requirement is to guarantee that:

$$\mathbb{P}_{D^{m+1}} \left[y_{m+1} \in f(x_{m+1}) \right] \geq 1 - \alpha \tag{2}$$

where D^{m+1} denotes the joint probability over both S and (x_{m+1}, y_{m+1}). In this setting, and in order for Eq. (2) to hold, the algorithm for constructing f is typically given access to both S *and* x_{m+1}. In contrast, in the batch setting, the algorithm is allowed access to S alone. To highlight this distinction we use the notations $f(\cdot \mid S, x_{m+1})$ and $f(\cdot \mid S)$ accordingly.

Methods which are designed to satisfy Eq. (2) can in principle be applied to the batch setting. This, however, often comes at either a computational or a statistical cost. One popular and straightforward approach is to compute a new predictor $f_x = f(\cdot \mid S, x)$ for every new test point x. For example, this idea lies at the core of conformal prediction, and is used to provide regret-type guarantees for the online learning settings. While feasible, such an approach carries a significant computational overhead. To circumvent this, other approaches partition S into effective 'train' and 'test' subsets [2, 13, 3]. This allows them to work with a single predictor, but can result in the downgrading or loss of statistical guarantees.

One way to restore some guarantees in this setting is to tighten the condition in Eq. (2), and require that the probability of success conditionally holds for *all*[1] examples [15], namely:

$$\mathbb{P}_{D^m, D_{Y|X}} \left[y \in f(x \mid S) \mid X = x \right] \geq 1 - \alpha \quad \forall x \in \mathcal{X} \tag{3}$$

Here, the probability is over the joint distribution of sample sets S and conditional new labels y. This means that for *every* $x \in \mathcal{X}$, the predicted intervals should contain the labels with probability of at least $1 - \alpha$, and is in effect what methods such as quantile regression [10, 9] aim for.

Due to its worst-case nature, the performance of predictors constructed to satisfy Eq. (3) is dominated by the difficult instances. Here we argue that such worst-case requirements can be overly demanding. In accordance, we propose an objective which modifies Eq. (3) in two important ways. First, we relax the worst-case requirement over $x \in \mathcal{X}$ to hold *in expectation* over the marginal D_X. Second, in the analysis (Sec. 5), we partition α into a confidence term δ over the training set and an accuracy term ϵ over new examples. Overall, our aim is to guarantee that, with probability of at least $1 - \delta$ over $S \sim D_{XY}^m$, we have:

$$\mathbb{P}_{D_{XY}} \left[y \in f(x \mid S) \right] \geq 1 - \epsilon \tag{4}$$

Intuitively, this means that when the training set is representative of the distribution, we'd like f to generalize well to new examples. In the next section, we show how this can be achieved by minimizing an appropriate loss function.

4 Method

Our approach considers the task of generating PIs from a model-free, discriminative learning perspective. Given a training set $S = \{(x_i, y_i)\}_{i=1}^m$ sampled i.i.d. from D_{XY}, our goal is to learn an interval predictor f with a low expected loss:

$$\mathcal{L}(f) = \mathbb{E}_D \left[L\left(y, f(x)\right) \right] \tag{5}$$

The loss function $L\left(y, f(x)\right)$ should quantify how well the predicted interval $f(x)$ fits the label y. Since a good interval is one which contains the true label, a natural choice is the following *interval 0/1-loss*:

$$L\left(y, [\ell, u]\right) = L\left(y, \ell, u\right) = \begin{cases} 0 & \text{if } y \in [\ell, u] \\ 1 & \text{otherwise} \end{cases} \tag{6}$$

Under this loss, Eq. (5) can be rewritten as:

$$\begin{aligned} \mathcal{L}(f) &= \mathbb{E}_X \mathbb{E}_{Y|X} \left[\mathbb{1}_{\{y \in f(x)\}} \mid X = x \right] \\ &= \mathbb{E}_X \Big[\mathbb{P}_{Y|X} \left[y \notin f(x) \mid X = x \right] \Big] \end{aligned} \tag{7}$$

[1] The guarantees in [15] are for *almost* all $x \in \mathcal{X}$.

Hence, a good mapping is one which produces good intervals *in expectation*. Specifically, a mapping f with $\mathcal{L}(f) = \alpha$ guarantees that predicted intervals will contain the true labels with probability $1-\alpha$, on average.

When comparing the above discriminative criterion with the standard PI criterion in Eq. (3), it becomes clear that they differ only in their requirement over x. Specifically, while Eq. (3) requires that labels be covered with probability $1-\alpha$ for *all* $x \in \mathcal{X}$, Eq. (7) requires this only in expectation. The discriminative objective in Eq. (7) can therefore be seen as a relaxation of the classic criterion, one which allows for variability in the probability of covering the true label. This gives our approach some flexibility, which as we show, can boost predictive performance.

4.1 Learning Objective

When considering how to train a predictor over S, it might at first be tempting to solve the following ERM:

$$\min_{f \in \mathcal{F}} \frac{1}{m} \sum_i L\left(y_i, \hat{\ell}_i, \hat{u}_i\right) \tag{8}$$

where $[\hat{\ell}, \hat{u}] = f(x)$ is the predicted interval, and $\mathcal{F}$ is a class of interval predictors. However, it quickly becomes apparent that without constraining the size of the predicted intervals, the loss can be arbitrarily low for *any* input. This is because, for any reasonable $\mathcal{F}$, intervals can be made large enough to always contain the true labels. This emphasizes a fundamental trade-off between error and interval size, and motivates the addition of a global *budget* constraint:

$$\begin{aligned} \min_{f \in \mathcal{F}} \quad & \frac{1}{m} \sum_i L\left(y_i, \hat{\ell}_i, \hat{u}_i\right) \\ \text{s.t.} \quad & \hat{\ell}_i \le \hat{u}_i \qquad \forall\, i = 1, \ldots, m \\ & \frac{1}{m} \sum_i \hat{u}_i - \hat{\ell}_i \le B \end{aligned} \tag{9}$$

Thus, for a given budget B, our hypothesis class is effectively restricted to include only predictors f which generate intervals with an average size of at most B. Of these, our objective is to choose one with minimal error. Note that the $\hat{\ell} \le \hat{u}$ constraints are always feasible when $\mathcal{F}$ includes functions with a bias term.

The idea of fixing a budget and minimizing error in some sense reverses their roles when compared to that of classic PI methods. In these, the first step is to predetermine and fix the amount of tolerated error α. Then, for a new point x, the interval is set to the tightest lower and upper bounds which contain $1-\alpha$ of the (estimated) conditional density of Y given X.[2]

[2]Typically under additional constraints, such as symmetric or one-sided error tails.

Compared to this approach, the roles of the objective and constraints in Eq. (9) are reversed.

Still, for cases where a fixed-error approach is more suitable, a "reversed" variant of Eq. (9) can be solved:

$$\begin{aligned} \min_{f \in \mathcal{F}} \quad & \sum_i \hat{u}_i - \hat{\ell}_i \\ \text{s.t.} \quad & \hat{\ell}_i \le \hat{u}_i \qquad \forall\, i = 1, \ldots, m \\ & \frac{1}{m} \sum_i L\left(y_i, \hat{\ell}_i, \hat{u}_i\right) \le \alpha \end{aligned} \tag{10}$$

When $L(\cdot)$ is convex (such as in the surrogate we consider in next section), Eqs. (9) and (10) are in fact dual, in the sense that for every error α there exists a budget B such that the solutions of Eq. (9) and Eq. (10) coincide. In this case, although Eq. (10) is tractable, it is not clear what generalization guarantees apply. In practice, a solution for the fixed-error setting can be obtained by perform a line search over B while solving Eq. (9), and using cross-validation to guarantee an expected error of approximately α.

4.2 Convex Surrogate

Since the labels and predictions in our setting are real, the problem we consider may appear to be one of regression. Nonetheless, it is in essence a classification problem, and the combinatorial nature of the 0/1 loss in Eq. (19) makes it NP-hard to solve (see Sec. 5.2). Due to this, we turn to learning with tractable surrogates. In what follows we suggest a convex surrogate to the 0/1 loss, and discuss its properties.

Our surrogate loss is inspired by the *ε-insensitive loss* used in Support Vector Regression Machines [26, 22]:

$$\begin{aligned} \tilde{L}(y, \hat{y}; \varepsilon) &= \max\{0, |y - \hat{y}| - \varepsilon\} \\ &= \begin{cases} 0 & \text{if } |y - \hat{y}| \le \varepsilon \\ |y - \hat{y}| & \text{otherwise} \end{cases} \end{aligned} \tag{11}$$

Here, a point-prediction $\hat{y} \in \mathbb{R}$ incurs no penalty if it is within distance ε of the true label y; otherwise, the penalty is linear. For the special case of $\varepsilon = 0$, the ε-insensitive reduces to the standard mean absolute error loss. For $\varepsilon = 1$, the loss $\tilde{L}$ can be thought of as a symmetrized variant of the popular *hinge loss* used in SVMs, in which both under- and over-estimates of y are penalized. This motivates the idea that, just as the hinge loss serves as a proxy to the binary 0/1-loss, the ε-insensitive loss can be used as a surrogate to the interval 0/1-loss.

Recall that our learning goal is to produce a low-error interval predictor under a budget constraint on the average interval size (Eq. (9)). For a given budget B, one way to achieve this is to set $\varepsilon = B$ and learn a

point-predictor $\tilde{f} : \mathcal{X} \to \mathbb{R}$ with $\tilde{L}$. Then, predicted intervals are constructed via $\hat{\ell} = \tilde{f}(x) - \Delta/2$ and $\hat{u} = \tilde{f}(x) + \Delta/2$, for $\Delta = \varepsilon$. Note that for every x, the interval size is exactly $\Delta = \hat{u} - \hat{\ell} = B$. Under this approach, the training loss and test-time predictions are calibrated: a predicted interval is penalized only if it does not contain the true label.

Clearly, fixing all interval sizes Δ to B is sufficient for satisfying the budget constraint. This, however, is not necessary, as the constraint requires only that the intervals be of size B *on average*. Hence, interval sizes can vary, and solutions with varying interval size can in principle give lower error.

The idea of varying interval sizes lies at the base of our approach. To allow for variation, we *parametrize* the interval size Δ, and jointly learn a point predictor $\hat{y} = f(x; w)$ and an interval-size predictor $\hat{\Delta} = g(x; v)$. For learning, we use a parametrized ε-insensitive loss, with a per-example insensitivity scale $\varepsilon(x) = g(x; v)$. The learning objective can be written as follows:

$$\begin{aligned} \min_{w,v} \quad & \frac{1}{m} \sum_i \tilde{L}\left(y_i, \hat{y}_i\,;\, \hat{\Delta}_i/2\right) \\ \text{s.t.} \quad & \hat{\Delta}_i \geq 0 \qquad \forall i = 1, \ldots, m \\ & \frac{1}{m} \sum_i \hat{\Delta}_i \leq B \end{aligned} \tag{12}$$

where in practice f and g are additionally regularized. For a liner parameterization, namely $f(x; w) = \langle w, x \rangle + b$ and $g(x; v) = \langle v, x \rangle + a$ for $w, v \in \mathbb{R}^d$ and $b, a \in \mathbb{R}$, the loss and constraints in Eq. (12) become convex in both w and v. Finding the global optimum can therefore be done efficiently using standard convex solvers. Note that while $\hat{y}$ lies at the center of $\hat{\Delta}$, the quantiles which correspond to the interval endpoints $\hat{\ell}, \hat{u}$ are not necessarily symmetric, and can vary across examples. This is in contrast to most methods which use fixed symmetric-tail quantiles.

Note that a possible alternative parametrization is to directly model the interval boundaries via $\hat{\ell} = f_\ell(x; w_\ell)$ and $\hat{u} = f_u(x; w_u)$. In the supplementary material we show that for linear predictors both parameterizations are equivalent (up to, perhaps, regularization).

The objective in Eq. (12) allows for variability in the size of predicted intervals. As in other approaches, this can be used to account for conditional heteroskedasticity. But, more importantly, a differential allocation of the budget allows for variability in the *conditional probability of errors*, namely:

$$\mathbb{P}_{Y|X}[y \notin [\hat{y} - \hat{\Delta}/2, \hat{y} + \hat{\Delta}/2] \mid X = x] \tag{13}$$

Intuitively, this allows our method to boost the overall accuracy by "sacrificing" the accuracy of some points (by reducing the size of their interval) in favor of others (by allocating them more of the budget).

Without restricting the form of $\hat{\Delta}$, solving Eq. (12) would most likely result in overfitting. Intuitively, in this scenario, intervals would either be allocated just enough budget to cover the labels exactly, or allocated no budget at all. Clearly, highly is undesired as it can lead to unstable predictions. It therefore remains to show that learning parametrized interval predictors via Eq. (12) can lead to good generalization. In the next section, we analyze the sample complexity of learning in our setting , and show that the VC dimension of a class of interval predictors is linear in the VC dimension of a corresponding class of threshold functions.

5 Theoretical Analysis

In this section we consider three theoretical aspects of our approach: the asymptotic properties of the interval constraints in Eq. (9), the sample complexity of our approach, and the computational hardness of ERM over the 0/1-interval loss in Eq. (19).

Constraints: Eq. (9) requires that the average interval size does not exceed the budget B, and that each predicted interval is consistent (that is, $\hat{\ell}_i \leq \hat{u}_i$). For the budget constraint, let D be a random variable specifying the size of predicted intervals,[3] with instances denoted by d. The distance between the average and expected interval sizes $|\frac{1}{m}\sum_{i=1}^m d_i - \mathbb{E}[D]|$ can therefore be bounded using standard concentration bounds. For consistency, note that if functions in $\mathcal{F}$ include a bias term, then the constraint can always be satisfied. This means that given an empirically consistent predictor f, its expected consistency can be estimated using sample complexity bounds for the realizable case. In practice, inconsistent predicted intervals can simply be replaced by point predictions. Alternatively, both constrains can be replaced by a single convex constraint of the form $\frac{1}{m}\sum_i \max\{0, \hat{u}_i - \hat{\ell}_i\} \leq B$.

Sample complexity: As noted in Sec. 1, classic PI methods consider a single confidence term α defined the joint distribution of training set *and* a new example. For our analysis, we separately consider the training set and new examples. We use PAC theory to determine, for every $\epsilon, \delta \in [0, 1]$, the minimal number of examples m that guarantee an expected error of at most ϵ (over the test distribution) with probability of at least $1 - \delta$ (over the train set). To this end, in the following Sec. 5.1 we analyze the VC dimension of

[3] The source of randomness for D are the samples (x_i, y_i) in S, which determine both the optimal f and the prediction outcomes $f(x_i) = [\hat{\ell}_i, \hat{u}_i]$.

learning a class of interval predictors $\mathcal{F}$.

Computational hardness: Many of the discontinuous losses in machine learning lead to combinatorial optimization problems which are hard to optimize. In Sec. 5.2 we prove that this is also the case for the interval 0/1 loss in Eq. (19). To this end, we show a reduction from the NP-hard problem MAX FS, which motivates the use of the convex proxy loss in Eq. (12).

5.1 VC Dimension

In this section we analyze the VC dimension of batch interval prediction. The difficulty in analysis lies in the fact that the function classes we consider are not binary, as both labels and predictions in our setting are real. Albeit the regressive nature of the learning setup, the loss is binary, and as a result, the complexity of learning is combinatorial. To overcome this difficulty, the main modeling point here is that we consider *both* x *and* y (and not just x) as the input to an hypothesis. This allows us to express the VC dimension of a class of interval predictors in terms of the VC dimension of some base class of threshold functions.

Let $\mathcal{B} = \{b : \mathcal{X} \to \mathbb{R}\}$ be a *base* class of real functions (e.g., linear functions). Denote by $\mathcal{F}(\mathcal{B})$ the class of interval predictors whose interval boundaries are set by functions in $\mathcal{B}$, namely function of the form $f_{b_\ell,b_u}(x) = [b_\ell(x), b_u(x)]$, where $b_\ell, b_u \in \mathcal{B}$. Our goal here is to bound the VC dimension of $\mathcal{F}(\mathcal{B})$. We show that this can be done by first considering the VC dimension of threshold functions over the base class $\mathcal{B}$.

In binary classification, a conventional way of using real functions $b \in \mathcal{B}$ for classification is to consider the corresponding classes of threshold functions:

$$\mathcal{H}_{\leq}(\mathcal{B}) = \{\mathbb{1}_{\{b(x)\leq\theta\}} \,:\, b \in \mathcal{B}\}$$
$$\mathcal{H}_{\geq}(\mathcal{B}) = \{\mathbb{1}_{\{b(x)\geq\theta\}} \,:\, b \in \mathcal{B}\}$$

For example, when $\mathcal{B}$ is the set of linear functions, both $\mathcal{H}_{\leq}(\mathcal{B})$ and $\mathcal{H}_{\geq}(\mathcal{B})$ are equivalent to the standard class of halfspace classifiers, whose VC dimension is $d+1$.

The VC dimension of a class $\mathcal{H}$ of binary classifiers (such as $\mathcal{H}_{\leq}(\mathcal{B})$ and $\mathcal{H}_{\geq}(\mathcal{B})$) regards the number of possible label assignments of functions $h \in \mathcal{H}$ to a set of m points $\{x_i\}_{i=1}^m$. The interval class $\mathcal{F}(\mathcal{B})$, however, is not binary, as functions $f \in \mathcal{F}(\mathcal{B})$ map examples to real intervals. Nonetheless, the true object of interest in terms of sample complexity is rather the number of assignments of the *loss function* over a given class. This means that, for a given sample set $S = \{(x_i, y_i)\}_{i=1}^m$ of size m, we will analyze the number of possible assignments of the interval 0/1 loss in Eq. (19) when learning with $\mathcal{F}(\mathcal{B})$. Formally, with a slight abuse of notation, we analyze the VC dimension of the binary class:

$$\mathcal{L}(\mathcal{F}) = \{L\left(y, f(x)\right) \,:\, f \in \mathcal{F}\} \tag{14}$$

It should be noted that analyzing the complexity of a class under a loss function is in fact the true (though sometime implicit) goal in standard binary classification as well. We prove the following result in the supplementary material:

Lemma 1. *The VC dimension of $\mathcal{H}$ equals the VC dimension of $\mathcal{L}_{0/1}(\mathcal{H})$.*

where $\mathcal{L}_{0/1}$ is appropriately defined over the standard binary 0/1 loss guven by $L_{0/1}\left(y, \hat{y}\right) = \mathbb{1}_{\{y\neq\hat{y}\}}$.

The main difference in the analysis of these classes lies in the input and output of the functions they include. For binary classifiers, the inputs are examples x, and the outputs (which we'd like to shatter) are labels y. In contrast, inputs to functions in $\mathcal{L}(\mathcal{F})$ and $\mathcal{L}_{0/1}(\mathcal{H})$ are *pairs* (x, y) of an example and a label, while outputs are binary assignments of the loss function.

The next theorem shows that the VC dimension of $\mathcal{F}(B)$ under $\mathcal{L}\left(\cdot\right)$ can be linearly bounded by the VC dimension of a corresponding binary threshold class. For simplicity, we focus on the case where $\mathcal{H}(\mathcal{B}) = \mathcal{H}_{\leq}(\mathcal{B}) = \mathcal{H}_{\geq}(\mathcal{B})$, which holds under mild assumptions. The general case is straightforward.

Theorem 2. *Let $\mathcal{B} = \{b : \mathcal{X} \to \mathbb{R}\}$ be a base class. If the VC dimension of the threshold class $\mathcal{H}(\mathcal{B})$ is k, then the VC dimension of the interval class $\mathcal{F}(\mathcal{B})$ over the interval 0/1-loss in Eq.* (19) *is at most $10k$.*

Proof. For some $f_{b_\ell,b_u} \in \mathcal{F}(\mathcal{B})$, we have:

$$L\left(y, f_{b_\ell,b_u}(x)\right) = \begin{cases} 0 & \text{if } b_\ell(x) \leq y \wedge b_u(x) \geq y \\ 1 & \text{otherwise} \end{cases}$$
$$= 1 - \mathbb{1}_{\{b_\ell(x)\leq y\}} \cdot \mathbb{1}_{\{b_u(x)\geq y\}} \tag{15}$$

Consider a set $S = \{(x_i, y_i)\}_{i=1}^m$ of size m that $\mathcal{L}(\mathcal{F})$ shatters. On the one hand, the m pairs have 2^m assignments under $\mathcal{L}(\mathcal{F})$. On the other hand, due to Eq. (15), the number of assignments is at most that of $\mathcal{H}_{\leq}(\mathcal{B})$ times that of $\mathcal{H}_{\geq}(\mathcal{B})$, namely:

$$\Pi_m\left(\mathcal{L}(\mathcal{F}(\mathcal{B}))\right) = 2^m \leq \Pi_m\left(\mathcal{H}_{\leq}(\mathcal{B})\right) \cdot \Pi_m\left(\mathcal{H}_{\geq}(\mathcal{B})\right)$$

where $\Pi_m(\cdot)$ denotes the maximal number of assignments of an hypothesis class over m points. Using the Sauer-Shelah Lemma and our assumptions on the VC dimension of $\mathcal{H}(\mathcal{B})$, for $k \leq m/3$ we have:

$$2^m \leq \left(\sum_{i=0}^{k} \binom{m}{i}\right)^2 \leq \left(\frac{em}{k}\right)^{2k} \tag{16}$$

Solving for the maximal m satisfying Eq. (16) gives $m \leq 10k$. Hence, $\mathrm{VC}(\mathcal{F}(\mathcal{B}))$ under $\mathcal{L}$ is at most $10k$.

□

Corollary 2.1. *Let $\mathcal{H}(\mathcal{B})$ be a class of binary classifiers over the base class $\mathcal{B}$ with VC-dimension k, and let $\mathcal{F}(\mathcal{B})$ be the corresponding interval class. Then, for every $\epsilon, \delta \in [0,1]$, if $\mathcal{H}(\mathcal{B})$ is (ϵ,δ)-PAC-learnable over the* binary *0/1-loss with $m(k)$ samples, then F(B) is (ϵ,δ)-PAC-learnable over the* interval *0/1-loss with $m(10k)$ samples. Hence, for any $m' \geq m(10k)$, we get:*

$$\mathbb{P}_{S\sim D_{XY}^{m'}}\left[\mathbb{P}_{D_{XY}}\left[y \notin f(x \mid S)\right] \leq \epsilon\right] \geq 1-\delta \qquad (17)$$

5.2 NP-hardness

In this section we establish the computational hardness of minimizing the empirical risk over the interval 0/1-loss as in Eq. (9). We focus on the linear case where $x \in \mathbb{R}^d$ and the interval class is:

$$\mathcal{F}_{\text{lin}} = \left\{f(x) = [b_\ell(x), b_u(x)] \,:\, b_\ell, b_u \in \mathcal{B}_{\text{lin}}\right\},$$
$$\mathcal{B}_{\text{lin}} = \left\{b(x) = \langle w, x\rangle + c \,:\, w \in \mathbb{R}^d,\, c \in \mathbb{R}\right\}$$

Theorem 3. *Minimzing Eq. (9) over $\mathcal{F}_{\text{lin}}$ is NP-hard.*

Proof. We show a reduction from the NP-hard problem Max-FS [18]. In this problem, given a (not necessarily feasible) linear system $Az = d$ of M equations over N variables, the goal is to find the maximal feasible subsystem.[4] Given an instance of Max-FS, namely $A \in \mathbb{R}^{M\times N}$ and $d \in \mathbb{R}^M$, we will construct a training set $S = \{(x_i, y_i)\}_{i=1}^m$ and budget B with corresponding optimal solutions. W.l.o.g. we will assume that rows in A are distinct and that $M \geq 2$.

Our construction is as follows. Let $m = 2M+1$. For every $i = 1,\ldots,M$, set $x_i = (A_{i1},\ldots,A_{iN})$ and $y_i = d_i$, for $i = M+1,\ldots,m$ set $x_i = (0,\ldots,0)$ and $y_i = 0$, and set $B = 0$. Let $f^*(x) = [b_\ell(x), b_u(x)]$ be a minimizer of Eq. (9), with $b_\ell(x) = \langle w_\ell, x\rangle + c_\ell$ and $b_u(x) = \langle b_u, x\rangle + c_u$. First, note that f^* must have $c_\ell = c_u = 0$. This is because a solution with no bias terms is correct on at least $M+1 > m/2$ examples, while any other solution can be correct on at most $M \leq m/2$ examples. Second, in order to satisfy the budget and interval constraints, it must also hold that $w_\ell = w_u = w$. This means that for $i = 1,\ldots,M$, we have $f^*(x_i) = y_i$ (and therefore $L\left(y_i, f^*(x_i)\right) = 0$) iff the i^{th} equation in $Az = d$ is satisfied by $z = w$, concluding our proof. □

6 Experiments

In this section we evaluate the performance of our method on a collection of benchmark datasets. We compare our method to several baselines on data from the UCI repository [16]. We used all dense-featured regression datasets with between 1,000 and 10,000 samples, giving a total of 8 datasets. All of our experiments use a 80:20 train-test split. For each considered setting we set all meta-parameters via 5-fold cross-validation on the training set over a tight grid. We report average results over 100 random data splits. For datasets with less than 25 features, we augment each example with pairwise interaction terms.

At its core, our method offers a general means for exploiting the variability in interval size for reducing error. While applicable to virtually any base class of real functions, our evaluation focuses on linear functions. This is because linear functions can be plugged into almost any method, provide computational guarantees for our method as well as the other baselines, and in many cases works well in practice. By using a linear base class in all methods, we allow for a fair comparison, where results express the statistical power (rather than computational traits) of the different methods.

In accordance with the above, we compare our method (IntPred) to several baselines, some of which are designed for the fixed-error setting (and take as input a significance lever α), and some, likes ours, which are designed for the fixed-budget setting (and take a budget B as input). Our fixed-error baselines include quantile regression (QuantReg) with symmetric tails, and batch-mode conformal prediction with the absolute loss (AbsConf) and squared loss (SqrConf). We use the efficient split-conformal inference method suggested in [13, 12], since other conformal methods are computationally infeasible for the datasets we consider. Our fixed-budget baselines include regressing with the absolute loss (AbsReg) and squared loss (LinReg) and adding a fixed symmetric interval of size B, as well as the fixed-budget method discussed in Sec. 4.2, which minimizes the ε-insensitive loss with $\varepsilon = B$ (SVR). We use L_2 regularization in methods where this is possible.

Our main evaluation criterion is the test error, namely the probability that a predicted interval does not contain the true test label. Since errors can be reduced simply by enlarging the intervals, for a fair comparison we evaluated all methods over a fixed set of average interval sizes $\{B_i\}$, and compared for each B_i separately. This is easily achieved for the fixed-budget methods (as they take budget inputs), but not neces-

[4] The popular version of Max-FS considers the system $Az \leq d$, but is also NP-hard for the relations $\{<, =, \neq\}$ [1].

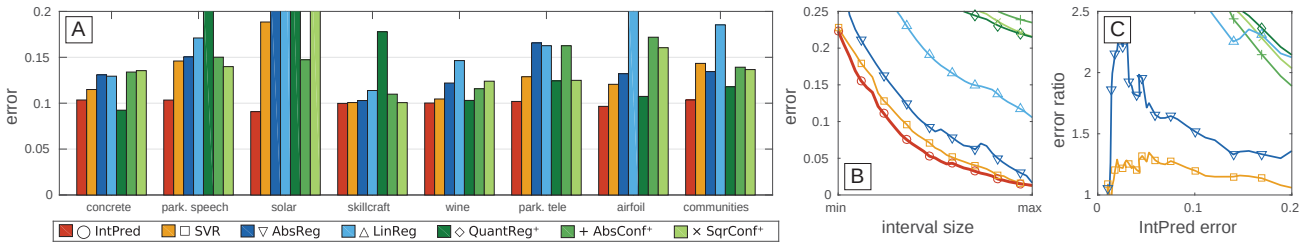

Figure 1: (A) Accuracy across datasets, for budgets B under which the test error of IntPred is roughly 0.1. (B) Test error per interval size, averaged over datasets. (C) Ratio of test error and the test error of IntPred.

sarily so the fixed-error methods, which do not offer a direct way to control interval sizes. To include these in the comparison, we first evaluate them over a tight set of confidence levels $\alpha_j \in [0, 1]$. Then, we interpolate errors from the interval size outputs, and compute approximate errors for the budgets B_i. For interpolation, we used 3rd-order monotonically-increasing concave splines with 5 knots over 30 points. This gave average R^2 values of 0.996 for QuantReg, 0.855 for AbsConf, and 0.901 for SqrConf. Furthermore, due to the difference in the scale of label values and variance across datasets, we present results for normalized budgets, where a normalized $B = 1$ corresponds (approximately) to an error of 0.1.

Our results are presented in Fig. 1. In the spirit of classic PI methods, we focus on the high-confidence regime with errors in the range $[0, 0.2]$.[5] Fig. 1 (A) presents an evaluation across all datasets. For each dataset, we show results for the B_i under which the test accuracy of our method was roughly 0.1. As can be seen, IntPred outperforms all other baselines for all but one dataset (where it ranks a close second). Across all evaluations, IntPred is statistically significantly better than the other methods (Friedman test [4], $P < 10^{-10}$). While other methods perform well over some datasets, their performance is in general inconsistent. This is demonstrated in Fig. 1 (B) which displays errors averaged over all datasets for a range of (normalized) budgets. As the plot shows, the volatility of some baselines (especially of the fixed-error methods) results in high average errors.

Recall that, in effect, SVR optimizes over the same interval loss as IntPred, but is constrained to fixed-size interval predictions. Hence, comparing IntPred to SVR quantifies the added value (in terms of error) of allowing for variability in interval size. Fig. 1 (C) therefore presents a direct comparison between the average error of IntPred and that of other methods, where each point on the plot corresponds to a different budget B_i. The relative performance of SVR demonstrates that interval-size flexibility reduces the error by roughly 20% across most of the focal error region.

7 Discussion

In this paper we presented a method for constructing prediction intervals in an inductive batch setting. Our framework views PI estimation as a discriminative learning task, where the goal is to minimize the probability of error under a budget constraint. The algorithmic, computational, and statistical results we presented were made possible due to our suggested modification of the classic PI objective: focusing on expected errors, allowing for error variability, and separating accuracy from confidence.

While the statistical properties we presented are general, the computational guarantees we discussed hold mostly for the convex surrogate loss in Eq. (12) over linear functions. In practice, however, our method can be applied to any function class, as well as other potential non-conex loss surrogates. It will therefore be intriguing to explore the empirical performance of our method when used with non-linear predictors (such as neural networks) over non-convex losses (such as an L^p-based loss with $p < 1$).

The experimental evaluation we presented demonstrates the empirical potential of allowing for variable errors across examples. However, our method achieves better overall accuracy at the price of explainability, as our method does not offer any guarantee on the confidence associated with each individual example. For example, a predicted interval can be small either because the confidence is high (and there is no reason to waist budget), or because it is low (and allocating budget is justified in terms of the loss). In interesting direction to explore is therefor to augment prediction with an estimate of confidence, namely the probability

[5] Both approaches present some difficulty when targeting very low test errors. For the fixed-budget methods, a higher budget leads to lower train errors. Hence, budgets which give a train error of 0 may be prone to overfitting. For most fixed-error methods, as there are typically no finite-sample guarantees on test error, even setting $\alpha = 1$ does not guarantee an arbitrarily low test error.

of the true label falling within the interval. We leave this for future research.

References

[1] Amaldi, E., and Kann, V. The complexity and approximability of finding maximum feasible subsystems of linear relations. *Theoretical computer science 147*, 1-2 (1995), 181–210.

[2] Burnaev, E., and Vovk, V. Efficiency of conformalized ridge regression. In *Conference on Learning Theory* (2014), pp. 605–622.

[3] Chen, W., Wang, Z., Ha, W., and Barber, R. F. Trimmed conformal prediction for high-dimensional models. *arXiv preprint arXiv:1611.09933* (2016).

[4] Demšar, J. Statistical comparisons of classifiers over multiple data sets. *Journal of Machine learning research 7*, Jan (2006), 1–30.

[5] Gammerman, A., and Vovk, V. Hedging predictions in machine learning: The second computer journal lecture. *The Computer Journal 50*, 2 (2007), 151–163.

[6] Geisser, S. *Predictive inference*, vol. 55. CRC press, 1993.

[7] Hosen, M. A., Khosravi, A., Nahavandi, S., and Creighton, D. Improving the quality of prediction intervals through optimal aggregation. *IEEE Transactions on Industrial Electronics 62*, 7 (2015), 4420–4429.

[8] Khosravi, A., Nahavandi, S., Creighton, D., and Atiya, A. F. Lower upper bound estimation method for construction of neural network-based prediction intervals. *IEEE Transactions on Neural Networks 22*, 3 (2011), 337–346.

[9] Koenker, R. *Quantile regression*. No. 38. Cambridge university press, 2005.

[10] Koenker, R., and Bassett Jr, G. Regression quantiles. *Econometrica: journal of the Econometric Society* (1978), 33–50.

[11] Koenker, R., and Hallock, K. Quantile regression: An introduction. *Journal of Economic Perspectives 15*, 4 (2001), 43–56.

[12] Lei, J., G'Sell, M., Rinaldo, A., Tibshirani, R. J., and Wasserman, L. Distribution-free predictive inference for regression. *Journal of the American Statistical Association*, just-accepted (2017).

[13] Lei, J., Rinaldo, A., and Wasserman, L. A conformal prediction approach to explore functional data. *Annals of Mathematics and Artificial Intelligence 74*, 1-2 (2015), 29–43.

[14] Lei, J., Robins, J., and Wasserman, L. Distribution-free prediction sets. *Journal of the American Statistical Association 108*, 501 (2013), 278–287.

[15] Lei, J., and Wasserman, L. Distribution-free prediction bands for non-parametric regression. *Journal of the Royal Statistical Society: Series B (Statistical Methodology) 76*, 1 (2014), 71–96.

[16] Lichman, M. UCI machine learning repository, 2013.

[17] Olive, D. J. Prediction intervals for regression models. *Computational statistics & data analysis 51*, 6 (2007), 3115–3122.

[18] Sankaran, J. K. A note on resolving infeasibility in linear programs by constraint relaxation. *Operations Research Letters 13*, 1 (1993), 19–20.

[19] Schmoyer, R. L. Asymptotically valid prediction intervals for linear models. *Technometrics 34*, 4 (1992), 399–408.

[20] Seber, G. A., and Lee, A. J. *Linear regression analysis*, vol. 936. John Wiley & Sons, 2012.

[21] Shafer, G., and Vovk, V. A tutorial on conformal prediction. *Journal of Machine Learning Research 9*, Mar (2008), 371–421.

[22] Smola, A. J., and Schölkopf, B. A tutorial on support vector regression. *Statistics and computing 14*, 3 (2004), 199–222.

[23] Steinberger, L., and Leeb, H. Leave-one-out prediction intervals in linear regression models with many variables. *arXiv preprint arXiv:1602.05801* (2016).

[24] Stine, R. A. Bootstrap prediction intervals for regression. *Journal of the American Statistical Association 80*, 392 (1985), 1026–1031.

[25] Taormina, R., and Chau, K.-W. Ann-based interval forecasting of streamflow discharges using the lube method and mofips. *Engineering Applications of Artificial Intelligence 45* (2015), 429–440.

[26] Vapnik, V. *The nature of statistical learning theory*. Springer science & business media, 2013.

Discriminative Learning of Prediction Intervals: Supplementary Material

Nir Rosenfeld
Harvard University
and Microsoft Research

Yishay Mansour
Tel Aviv University

Elad Yom-Tov
Microsoft Research

1 Equivalence of Parameterization

There are two natural parameterizations of intervals. The first is to model the interval boundaries via:

$$\hat{\ell} = f_\ell(x; w_\ell), \qquad \hat{u} = f_u(x; w_u)$$

The second is to model the interval center and size:

$$\hat{y} = f(x; w), \qquad \hat{\Delta} = g(x; v)$$

Here we show that for linear predictors, both parameterizations are equivalent. Specifically, we show that for every (w_ℓ, w_u) there exist some $(\tilde{w}, \tilde{v})$ whose predictions coincide on all x, and vice versa.

Let $w_\ell, w_u \in \mathbb{R}^d$, and consider some $x \in \mathcal{X}$. Note that:

$$\hat{y} = \frac{1}{2}(w_\ell^\top x + w_u^\top x) = \big(\frac{1}{2}(w_\ell + w_u)\big)^\top x = \tilde{w}^\top x$$
$$\hat{\Delta} = w_u^\top x - w_\ell^\top x = (w_u - w_\ell)^\top x = \tilde{v}^\top x$$

Similarly, for $w, v \in \mathbb{R}^d$, we have:

$$\hat{\ell} = w^\top x - \frac{1}{2}v^\top x = (w - \frac{1}{2}v)^\top x = \tilde{w_\ell}^\top x$$
$$\hat{u} = w^\top x + \frac{1}{2}v^\top x = (w + \frac{1}{2}v)^\top x = \tilde{w_u}^\top x$$

We note that the only practical difference here would be when each component is regularized separately (e.g., with a different regularization constant).

2 VC of Functions and Losses

Recall that for a loss function L and a function class $\mathcal{F}$, we define their composition as:

$$\mathcal{L}(\mathcal{F}) = \{L\left(y, f(x)\right) \,:\, f \in \mathcal{F}\} \tag{18}$$

we denote the by $\mathcal{L}_{0/1}(\mathcal{H})$ the composition of the binary 0/1-loss function $L_{0/1}\left(\cdot\right)$ with a binary function class $\mathcal{H}$, where:

$$L_{0/1}\left(y, \hat{y}\right) = \mathbb{1}_{\{y \neq \hat{y}\}} \tag{19}$$

Here we prove the following claim:

Lemma 4. *The VC dimension of $\mathcal{H}$ equals the VC dimension of $\mathcal{L}_{0/1}(\mathcal{H})$.*

Proof. The important observation here is that while both $\mathcal{H}$ and $\mathcal{L}_{0/1}(\mathcal{H})$ include functions with binary outputs, the functions differ in their domain. Specifically, functions in $\mathcal{H}$ map items x to binary outputs $y \in \{0, 1\}$, while functions in $\mathcal{L}_{0/1}(\mathcal{H})$ take as input pairs (x, y) with $y \in \{0, 1\}$ and, via some $h \in \mathcal{H}$, output the loss value $z \in \{0, 1\}$.

Assume the VC dimension of $\mathcal{H}$ is m, then there exist some $x_1, \ldots, x_m$ which shatter $\mathcal{H}$. This means that for every $y_1, \ldots, y_m$, there exists some $h_y \in \mathcal{H}$ for which $h_y(x_i) = y_i$ for every i. Consider the set of pairs $(x_1, 0), \ldots, (x_m, 0)$. For any $z_1, \ldots, z_m$, we have some $h_z \in \mathcal{H}$ for which $h_z(x_i) = z_i$ for every i. This means that any $z_i = 0$ gives $h_z(x_i) = 0$, and hence:

$$L_{0/1}\left(0, h_z(x_i)\right) = 0 = z_i$$

Similarly, for any $z_i = 1$ we have $h_z(x_i) = 1$ and:

$$L_{0/1}\left(0, h_z(x_i)\right) = 1 = z_i$$

Now, assume the VC dimension of $\mathcal{L}_{0/1}(\mathcal{H})$ is m. Then there exist some $(x_1, y_1), \ldots, (x_m, y_m)$ which shatter $\mathcal{L}_{0/1}(\mathcal{H})$. This means that for every $z_1, \ldots, z_m$, there exists some $h_z \in \mathcal{H}$ such that $L_{0/1}\left(y_i, h_z(x_i)\right) = z_i$. Consider the set $x_1, \ldots, x_m$. For any $y_1, \ldots, y_m$, set $z_i = 0$ for all i. Hence, the corresponding h_z is such that $L_{0/1}\left(y_i, h_z(x_i)\right) = 0$ for all i, which means that $h(x_i) = y_i$ as needed. □