Online Algorithms with Better-Than-Random Predictions

Cooper Sigrist
UMASS Amherst
Amherst, Massachusetts, United States

Shahin Khamali
York University
Toronto, Canada

Bo Sun
Chinese University of Hong Kong
Hong Kong, China

Mohammed Hajiesmaili
UMASS Amherst
Amherst, Massachusetts, United States

ABSTRACT

As Machine Learning methods have become more reliable, there has been growing interest in designing algorithms that leverage predictions about the input. A number of previous works have designed algorithms that benefit from good predictions and are robust against adversarial error in prediction. Considering adversarial predictions is, however, inherently pessimistic and unrealistic in many cases. In this work, we show a method to design algorithms around a weak assumption of better-than-random (BTR) predictions and show that these algorithms can perform much better in practice while only ever performing as poorly as if we had no prediction in the worst case. We believe this prediction model to be a more realistic and practical methodology.

KEYWORDS

Online Algorithms, Algorithms with Advice, Predictions

ACM Reference Format:
Cooper Sigrist, Shahin Khamali, Bo Sun, and Mohammed Hajiesmaili. 2023. Online Algorithms with Better-Than-Random Predictions. In *Proceedings of ACM Conference (Conference'17).* ACM, New York, NY, USA, 2 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION

Multiple works have been able to use predictions to great effect in algorithm design [3]. They have found a particular home in online algorithms for their natural ability to help predict the unknown future of such problems. Inherent in the use of such predictions are two major components:

(1) Decision of what to predict (and when) [1][2]
(2) Assumptions about the fallibility of the prediction.

In many cases, the latter can inform the former and we see that differences in this assumption can lead not only to different results but also fundamentally different approaches to algorithm design. In this work, we will explore one such assumption about the second component with an emphasis on the practicality of the prediction model in real-world applications.

Conference'17, July 2017, Washington, DC, USA

ACM ISBN 978-x-xxxx-xxxx-x/YY/MM…$15.00
https://doi.org/10.1145/nnnnnnn.nnnnnnn

1.1 Previous Works

Before explaining our prediction model, we briefly review some of the existing models.

- **Advice model:** The advice model is the earliest and most theoretical model, which assumes that the predictions are perfect and reliable. This assumption is obviously not practical but is useful in finding lower bounds on the amount of information needed for certain performance ratios [3, 4, 11].
- **Predictions with unbounded error:** Under this model, the predictions may have any error and are considered to be adversarial in the worst case. Designing algorithms under this model will minimize the impact of adversarial predictions (Robustness) but is extremely pessimistic and will fail to use predictions as well as possible in cases where we can reliably trust them (Consistency), this [6, 10].
- **Predictions with bounded error:** This model assumes that the error (generally the L1 distance) of the prediction is bounded by some value, H. The competitive ratio is then computed as a function of H. It is, however, difficult to justify this parameter H and designing algorithms around this assumption still requires minimizing the impact of adversarial cases if generalizing for all values of H. [8]
- **Randomly Infused Advice:** Under this model, the prediction is considered perfect, but with some probability, say p, each bit of the prediction may be set randomly. Therefore, in the worst case, the prediction becomes a completely random sequence rather than an adversarial one, and this is more realistic in practice. This model does, however, gives extra power to the algorithms in terms of random bits, which can be helpful on their own. In particular, this allows designing predictions with error correction bits, which gives impractical lower bounds [5].[7]

Attempts to incorporate prediction models into real applications have enhanced the underlying assumptions to become more realistic over time. Yet, the existing models are either difficult to implement in practice or do not make the best use of the general reliability of predictions.

2 BETTER-THAN-RANDOM PREDICTIONS

In our prediction model, we build over previous works and take one step further toward practicality. Like the randomly infused advice, our predictor model performs randomly in the worst case (and not adversarially), and like many existing results, we use the L1 distance

to quantify the prediction error. Together, these assumptions yield our notion of Better-Than-Random (BTR) Predictions.

We consider a prediction that is made about a value in a bounded range (m, M). Let y denote the true value. A prediction $\hat{y}$ is called a BRT prediction if it satisfies

$$|y - \hat{y}| \leq \mathbb{E}[|y - Y|], \tag{1}$$

where $Y \in (m, M)$ is a random prediction that will be specified based on problems (e.g., $Y = U(m, M)$ is a uniform random variable in the cow-path problem).

This assumption is very weak and is expected to hold in practical application. A machine-learned method that regularly violates this assumption would likely be identified and discarded before attempts to use it in any practical way.

Performance metrics. Since we will be focusing on online algorithms, we evaluate the performance of algorithms by a prediction-augmented competitive ratio (CR). Let Ω denote the set of instances that satisfy the BTR prediction (1). Then for a minimization problem, the prediction-augmented CR is defined as

$$\mathsf{CR}_b = \max_{I \in \Omega} \frac{\mathsf{ALG}(I)}{\mathsf{OPT}(I)}, \tag{2}$$

which is the worst-case ratio between the cost of the algorithm (i.e., $\mathsf{ALG}(I)$) and that of an optimal offline algorithm (i.e., $\mathsf{OPT}(I)$) across all instances in Ω. Note that Ω is a subset of the instances of original online problems. Thus, CR_b is no larger than the conventional worst-case optimized CR. This work is to investigate (i) how to leverage BTR predictions in algorithmic design and (ii) when BTR predictions are useful in terms of improving CR_b.

3 TOY EXAMPLE: COW-PATH PROBLEM

We used our prediction model to design and test algorithms for a few problems. Here, we briefly review the algorithm and results on a restricted variant of the one-dimensional search problem, which we call the bounded cow-path problem. Though this is a toy problem, it demonstrates how algorithms must be designed under our prediction model.

3.1 Bounded Cow Path

The problem is defined as follows: There exists a "cow" at the origin, and an unknown target value $y \in (-M, M)$ for a known value M. Starting at 0, the cow moves back and forth in search of the target value; it finds the target if it reaches the position y on the line segment $(-M, M)$. The objective is to move the cow in a way to minimize the total distance it travels until it finds the target. The optimal offline cost is the magnitude of y. The optimal, deterministic, online algorithm with no prediction for this problem is a doubling strategy that alternately moves powers of two on either side. This approach has a competitive ratio of 9 [9].

3.2 Algorithm with BTR Predictions

We begin by computing $\mathbb{E}[|y - U(-M, M)|]$, giving us the following:

$$\begin{aligned}\mathbb{E}[|y - U(-M, M)|] &= \left(\frac{M - y}{2M} \cdot \frac{M - y}{2}\right) + \left(\frac{M + y}{2M} \cdot \frac{M + y}{2}\right) \\ &= \frac{y^2}{2M} + \frac{M}{2}.\end{aligned}$$

Given the BTR prediction $\hat{y}$, we can estimate the possible ranges of the true position

$$y \in \begin{cases} (-M, M - \sqrt{-2M\hat{y}}) & \hat{y} \in (-M, 0] \\ (-M + \sqrt{2M\hat{y}}, M) & \hat{y} \in (0, M) \end{cases}. \tag{3}$$

From this, we can realize that a prediction with good enough magnitude, specifically $|\hat{y}| \geq \frac{M}{2}$, will tell us the direction from the origin where the target is located, and hence we can solve the problem optimally by moving the cow on the right direction. On the other hand, when $|\hat{y}| < \frac{M}{2}$, we have a reduced range of possible positions for the target. This allows us to choose a starting side of the doubling algorithm to minimize the likelihood of a very high cost ratio. This will decrease the expected cost ratio, but won't improve on the competitive ratio in the worst case.

3.3 Experimentation

To test our algorithm, we used uniformly random cow positions and predictions generated either from a uniform distribution over the range allowed by the assumption or a normal distribution with a mean equal to y, and standard deviation equal to the random error bound H divided by 3. We ran this test over different magnitudes of M up to 10^8 and averaged over 10,000 tests for each order of magnitude. We found that the average performance ratio of our algorithm was around 3.3 for both methods of generating predictions. This was consistent over every order of magnitude and significantly better compared to the competitive ratio of 5.2 of the purely online algorithm (without prediction). This is to be expected, but we believe that this is evidence that designing an algorithm around this assumption can give better empirical performance without fear of adversarial cases causing issues, at least in these toy problems.

REFERENCES

[1] Keerti Anand, Rong Ge, Amit Kumar, and Debmalya Panigrahi. 2022. Online algorithms with multiple predictions. In *International Conference on Machine Learning*. PMLR, 582–598.

[2] Keerti Anand, Rong Ge, and Debmalya Panigrahi. 2020. Customizing ML Predictions for Online Algorithms. *ArXiv* abs/2205.08715 (2020).

[3] Joan Boyar, Lene M Favrholdt, Christian Kudahl, Kim S Larsen, and Jesper W Mikkelsen. 2017. Online algorithms with advice: A survey. *ACM Computing Surveys (CSUR)* 50, 2 (2017), 1–34.

[4] Joan Boyar, Shahin Kamali, Kim S Larsen, and Alejandro López-Ortiz. 2016. Online bin packing with advice. *Algorithmica* 74 (2016), 507–527.

[5] Yuval Emek, Yuval Gil, Maciej Pacut, and Stefan Schmid. 2023. Online Algorithms with Randomly Infused Advice. *arXiv preprint arXiv:2302.05366* (2023).

[6] Google and Mountain View. 2018. Improving Online Algorithms via ML Predictions.

[7] Anupam Gupta, Debmalya Panigrahi, Bernardo Subercaseaux, and Kevin Sun. 2022. Augmenting Online Algorithms with varepsilon Accurate Predictions. *Advances in Neural Information Processing Systems* 35 (2022), 2115–2127.

[8] Sungjin Im, Ravi Kumar, Mahshid Montazer Qaem, and Manish Purohit. 2021. Online knapsack with frequency predictions. *Advances in Neural Information Processing Systems* 34 (2021), 2733–2743.

[9] Ming-Yang Kao, John H. Reif, and Stephen R. Tate. 1996. Searching in an unknown environment: an optimal randomized algorithm for the cow-path problem. *Inf. Comput.* 131 (1996), 63–79.

[10] Thodoris Lykouris and Sergei Vassilvitskii. 2018. Competitive caching with machine learned advice. *J. ACM* 68 (2018), 24:1–24:25.

[11] Dhruv Rohatgi. 2019. Near-Optimal Bounds for Online Caching with Machine Learned Advice. *ArXiv* abs/1910.12172 (2019).