Adversarial Balancing for Causal Inference

Michal Ozery-Flato[*]
IBM Research

Pierre Thodoroff[*†]
McGill University

Tal El-Hay
IBM Research

Abstract

Biases in observational data pose a major challenge to estimation methods for the effect of treatments. An important technique that accounts for these biases is reweighting samples to minimize the discrepancy between treatment groups. Inverse probability weighting, a popular weighting technique, models the conditional treatment probability given covariates. However, it is overly sensitive to model misspecification and suffers from large estimation variance. Recent methods attempt to alleviate these limitations by finding weights that minimize a selected discrepancy measure between the reweighted populations. We present a new reweighting approach that uses classification error as a measure of similarity between datasets. Our proposed framework uses bi-level optimization to alternately train a discriminator to minimize classification error, and a balancing weights generator to maximize this error. This approach borrows principles from generative adversarial networks (GANs) that aim to exploit the power of classifiers for discrepancy measure estimation. We tested our approach on several benchmarks. The results of our experiments demonstrate the effectiveness and robustness of this approach in estimating causal effects under different data generating settings.

1 Introduction

Causal inference deals with estimating the potential outcomes under certain treatments or interventions. The gold standard for causal inference studies is randomized controlled trials (RCTs), in which treatment and control groups come from the same data distribution, due to a randomized treatment assignment process. However, RCTs are often costly, sometimes impractical to implement, and may raise ethical questions. An appealing alternative is to use the abundance of available observed data. However, in such data there is no control over the treatment assignment mechanism, which is often unknown to the researcher. Moreover, treatment populations are likely to differ from the population out of which they are drawn. This difference, or bias, may hinder the inference of the potential outcomes for the treatment in the larger population.

The range of algorithms for estimating potential outcomes from observational data can be roughly categorized into three approaches. The first method, referred here as balancing weights, attempts to change the weights of the unit samples such that the resulting distributions of observed covariates in the compared datasets become statistically similar. The second approach employs outcome models to extrapolate unobserved outcomes. The third approach integrates balancing weights with outcome models. Therefore, balancing weights can be used to estimate the average treatment effects, either directly by computing weighted averages of observed outcomes, or in conjunction with outcome modeling approaches. Other research areas have extensively studied the problem of finding balancing weights, albeit under different names, such as *domain adaptation* [3, 15] and *covariate shift* [9, 30].

Inverse propensity weighting (IPW) [26] is a historical and widely-used balancing method that models the conditional treatment probability given pre-treatment covariates. If the model is correctly

[*]equal contribution
[†]This work was done while the author was an intern in IBM Research-Haifa

Preprint. Work in progress.

specified, then the computed weights are balancing [13]. However, a misspecified model may generate weights that fail to balance the biases, potentially leading to erroneous estimations. In recent years, various methods were developed to generate weights that directly minimize the different objectives used to measure the discrepancy between compared populations (e.g., [11, 7, 14, 33, 1, 18, 32]). The key factors to determining the success of such methods are the way they represent bias between weighted samples, the objective functions that are derived from this measure, hyper-parameters that are involved in the objective, and the means to optimize it.

We propose a framework that adapts the objective function of generative adversarial networks (GANs) to the task of generating balancing weights. This framework is motivated by the immense success of GANs in producing simulated data that highly resembles real world samples. Similar to GANs, our framework is based on a two-player game that involves a discriminative model and a data generator. The key difference is that our data generator produces "new" data by reweighting a given dataset. Our objective is to find weights that maximize the minimum loss of the discriminator. Simply put, the goal is to generate weights for a given dataset that will make it indistinguishable, in the eyes of the discriminator, from another given dataset. Using this framework allows us to harness the complete arsenal of classification methods to the task of inferring balancing weights. Under this framework, we propose a novel algorithm for generating balancing weights that iteratively updates the weights for a given dataset using a standard classifier. We evaluated the performance of this algorithm on a range of published causal-inference benchmarks, and assessed the ability to select an appropriate classifier for the input datasets in a standard cross-validation routine.

2 Problem formulation - balancing for causal inference

Consider a population where each individual received a single treatment from a finite set of treatments $\mathcal{A}$. The received treatment and the resulting outcome for every individual are indicated by the covariates A and Y, respectively. For every treatment $a \in \mathcal{A}$, Y^a denotes the potential outcome for the treatment. This outcome is observed only when $A = a$, that is, $Y = \sum_a Y^a \cdot \mathbb{1}_{A=a}$. Let X denote the vector of observed pre-treatment covariates used to characterize the individuals, and denote by p its distribution in the population. The expected outcome of a treatment $a \in \mathcal{A}$ in the population is:

$$\mathbb{E}[Y^a] \equiv \int_x \int_y y \cdot p(Y^a = y|X = x)p(X = x)dydx \equiv \mathbb{E}_{X\sim p(X)}[\mathbb{E}_{Y^a\sim p(Y^a|X)}[Y^a|X]] . \tag{1}$$

The goal of many observational studies is to estimate $\mathbb{E}[Y^a]$ from a finite data sample $\mathcal{D} = \{(x_i, a_i, y_i)\}_{i=1}^n$. Conceptually, a straightforward estimation of $\mathbb{E}[Y^a]$ could be made by sampling Y^a from a population in which $X \sim p(X)$. However, Y^a is observed only in the subpopulation that actually received treatment a, and in which $X \sim p(X|A = a) \not\equiv p(X)$. To overcome this hurdle, we employ the standard assumptions of *strong ignorability*: $Y^a \perp\!\!\!\perp A|X$, and *positivity*: $0 < p(A = a|X = x) < 1$, $\forall a \in \mathcal{A}$ [27]. Strong ignorability, often stated as "no hidden confounders", means that the observed covariates contain all the information that may affect treatment assignment. These assumptions allow rewriting Equation 1:

$$\mathbb{E}[Y^a] = \mathbb{E}_{X\sim p(X)}[\mathbb{E}_{Y\sim p(Y|X,A=a)}[Y|X, A = a]] . \tag{2}$$

Equation 2 suggests that $\mathbb{E}[Y^a]$ can be estimated by a sample from the subpopulation corresponding to $A = a$ under the condition that the sample is distributed according $p(X)$ rather than the actual distribution $p(X|A = a)$. A common approach to handle this sampling challenge is to assign a weight $\omega^a(x)$ to each individual characterized by x in the subpopulation under treatment a such that $p(X = x|A = a)\omega^a(x) = p(X = x)$. The weights that satisfy this condition are $\omega^a(x) = \frac{p(A=a)}{p(A=a|X=x)}$ [13], and therefore

$$\mathbb{E}[Y^a] = \mathbb{E}_{x\sim p(X|A=a)}[\mathbb{E}[Y|X = x, A = a]\omega^a(x)] . \tag{3}$$

We refer to weights as *balancing* if they balance the biases between the distributions $p(X|A = a)$ and $p(X)$, making them more similar to each other. Given a finite sample $\mathcal{D}$, the ultimate goal of all balancing methods is to produce weights w_i that approximate $\omega^a(x_i)$. IPW estimates w_i by learning a model for $p(A = a|X = x)$ from $\mathcal{D}$ and using the inverse of the estimated probabilities, which may be unstable. Therefore, more recent methods attempt to infer the weights directly. Following Equation3, given w_i we use the following estimation for the expected potential outcome:

$$\widehat{\mathbb{E}[Y^a]} = \sum_{i:A_i=a} w_i y_i \,. \tag{4}$$

3 Background on adversarial framework for learning generative models

The adversarial framework, which was introduced by Goodfellow et al. [6], aims to learn a generative model of an unknown distribution p using a class of discriminators that gauge the similarity between data distributions. This framework can be described as a game in which a generator simulates data and a discriminator tries to distinguish samples of simulated data from true data samples. The generator employs generative models with an input random variable Z from a predefined distribution p_Z and a deterministic mapping $g(z)$ to the data space. Simulated data is generated by sampling data from z and transforming them through g. At the end of each round of the game, the generator observes the predictions of the discriminator and updates the model for $g(z)$. The discriminator models the probability $d(x)$ that a sample is from $g(z)$ and not from the true data distribution by attempting to minimize the following loss:

$$L(g, d) = \mathbb{E}_{X\sim p}[l(d(X), 0)] + \mathbb{E}_{Z\sim p_Z}[l(d(g(Z)), 1)] = \mathbb{E}_{X\sim p}[l(d(X), 0)] + \mathbb{E}_{X\sim p_g}[l(d(X), 1)] \tag{5}$$

where p_g is the distribution implicitly induced by $g(Z)$ and l is the loss function. The generator attempts to maximize the expected loss, and its objective is to find

$$g^* = \arg\max_g \left(\min_d L(g, d) \right) \tag{6}$$

Examples for loss functions are the Log-loss $l(d(x), c) = (1 - c) \cdot \log d(x) + c \cdot \log(1 - d(x))$, which was used in [6]; and the 0-1 loss, $l_{0,1}(d(x), c) = \mathbb{1}\left[\mathbb{1}\left[d(x) > \frac{1}{2}\right] \neq c\right]$, which was used in [10, 25] for likelihood-free inference and in [23] for classifier two-sample tests. In the next section we adapt the adversarial framework and its key principle of maximizing the discrimination loss to the task of generating balancing weights.

4 Adversarial balancing

In this section we present our adversarial framework for generating balancing weights, and a novel algorithm that applies it. Similar to GAN, our goal is to generate a sample that resembles data coming from a true distribution $p(X)$. However, while in the original GAN framework the generated sample is simulated by applying a transformation on unlimited random data, our balancing framework is constrained to reweight a finite sample from $p(X|a)$.

4.1 Discrepancy objective

Given a finite sample from the distribution $p(X|a)$, our goal is to find weights that maximize the probability of a discriminator to fail to distinguish the reweighted sample from an unweighted sample from $p(X)$. We adapt the adversarial objective for this purpose as follows. Suppose $\widehat{\omega^a}(x)$ is a non-negative function that reweights samples from $p(X|a)$, resulting in a new distribution $q_a(x) \equiv \widehat{\omega^a}(x)p(x|a)$. For $q_a(x)$ to be a valid density function, $\omega^a(x)$ should satisfy the constraint $\mathbb{E}_{p(x|a)}[\omega^a(x)] = 1$. Plugging q_a into p_g into the loss defined in Equation 5 we obtain

$$L(q_a, d) = \mathbb{E}_{X\sim p(X)}[l(d(X), 0)] + \mathbb{E}_{X\sim q_a(X)}[l(d(X), 1)]$$

Since $p(x|a)$ is a density function, we can rewrite the second term as follows:

$$L(\widehat{\omega^a}, d) = \mathbb{E}_{X\sim p(X)}[l(d(X), 0)] + \mathbb{E}_{X\sim p(X|a)}[\widehat{w^a}(X) \cdot l(d(X), 1)] \tag{7}$$

We can confine the representation of $\widehat{\omega^a}$ to a family of models and optimize the loss with respect to this family. However, for the estimation problem defined in Equation 4, it suffices to infer point estimates of ω^a for the given sample from $p(X|A = a)$. We take this approach without making any parametric assumptions on the functional form of ω^a. Denoting these estimates by $w_i \equiv \widehat{\omega^a}(x_i)$, the loss function becomes

$$L_n(\mathbf{w}, d; a) = \frac{1}{n} \sum_{i=1,\dots,n} l(d(x_i), 0) + \frac{1}{n_a} \sum_{i:A_i=a} w_i l(d(x_i), 1)\,, \tag{8}$$

where $\mathbf{w} \equiv [w_i]_{i:A_i=a}$ are the weights corresponding to the individuals with $A = a$, and n_a as their number. Note this expression is the empirical loss of a learning algorithm for the defined prediction task. Also, Equation 8 gives similar importance to classification errors in the sample from $p(X)$ and the reweighted subsample corresponding to $A = a$. Since $\mathbb{E}_{p(x|a)}[\omega^a(x)] = 1$ we require that the generated weights satisfy the normalization property $\frac{1}{n_a} \sum_{i:A_i=a} w_i = 1$.

The aim of the discriminator is to minimize the loss in Equation 8. Weights that lead to a failure of the discriminator to distinguish between the reweighted and unweighted samples will result in a maximization of the minimal loss. Therefore, we formulate the objective of the adversarial balancing framework as solving the following optimization problem:

$$\mathbf{w}^* = \arg\max_{\mathbf{w}} \left(\min_d L_n(\mathbf{w}, d; a) \right), \text{ such that } \mathbf{w} \in \Delta^{n_a}\,, \tag{9}$$

where $\Delta^{n_a} = \{\mathbf{w} |\, \mathbf{w}[i] \geqslant 0, \|\mathbf{w}\|_1 = n_a\}$ is the n_a-dimensional simplex. This is a general framework for estimating the expected potential outcome for a given treatment; therefore, it can be used to estimate different types of causal effects. For example, the it average treatment effect (ATE) is defined as $\mathbb{E}[Y^1] - \mathbb{E}[Y^0]$. Another example is the *average treatment effect in the treated* (ATT), which is defined as $\mathbb{E}[Y^1|A = 1] - \mathbb{E}[Y^0|A = 1]$. We estimate $\mathbb{E}[Y^0|A = 1]$ by reweighting the data sample from the untreated and using the data from treated individuals as the true data sample for comparison.

4.2 Weight learning algorithm

To search for a solution to the max-min objective in Equation 9, we propose the following iterative process. At each step we train a discriminator to minimize the empirical loss of Equation 8. We then update the weights w_i to increase this loss using a single step of exponentiated gradient descent [21], which maintains the weight normalization constraint.

We describe our algorithm for estimating $\mathbb{E}[Y^a]$ in the general population. Let $\mathcal{D}_a$ be the data sample of the population under treatment a, that is, $\mathcal{D}_a = \{x_i : A_i = a\}$. The data sample from the true distribution is $\mathcal{D}_{pop} = \{x_i\}_{i=1}^n$. Note that by this definition $\mathcal{D}_a \subset \mathcal{D}_{pop}$. However, our framework can be applied to different definitions of $\mathcal{D}_a$ and $\mathcal{D}_{pop}$[3]. Given an initial set of weights $\{w_i\}$ for the unit samples in $\mathcal{D}_a$, we define the augmented labeled dataset $\mathcal{D}$ by assigning a class label 1 and weights w_i to $\mathcal{D}_a$, and a class label 0 to $\mathcal{D}_{pop}$:

$$\mathcal{D} \equiv \left\{(\mathbf{x}_j, 0; w_j = 1) | \mathbf{x}_j \in \mathcal{D}_{pop}\right\} \cup \left\{(\mathbf{x}_i, 1; w_i) | \mathbf{x}_i \in \mathcal{D}_a\right\}. \tag{10}$$

Note that the unit samples in $\mathcal{D}_{pop}$ are unweighted ($w_j = 1$). The discriminator predicts the class label, C, of the samples in $\mathcal{D}$ using a classification algorithm $d(\mathbf{x})$. Recall that the final objective of the adversarial framework is to find $\mathbf{w}$ that maximizes the objective in Equation 9. Following Equation 8, for a fixed classifier d, the generator's loss is linear in $\mathbf{w}$ and $\frac{\partial L_n}{\partial w_i} = l(d(\mathbf{x}_i), 1)$ is constant. To maximize the objective in Equation 9, which refers to *any* classifier from the considered family, we update the weights using a single step of exponentiated gradient ascent:

$$w_i^{t+1} = n_a \frac{w_i^t \exp\left(\alpha \cdot l(d(x_i), 1)\right)}{\sum_j w_j^t \exp\left(\alpha \cdot l(d(x_j), 1)\right)} \tag{11}$$

Algorithm 1 shows the complete details of the adversarial framework for non-parametric generation of balancing weights. Only the weights for $\mathcal{D}_a$ are updated, while weights for the sample units in $\mathcal{D}_{pop}$ are constantly set to 1. In each iteration the sum of weights in $\mathcal{D}_a$ equals n, ensuring the same importance with respect to the discriminator loss. The predictions of the discriminator (Step 6 in Algorithm 1) should preferably be obtained with cross validation, to better approximate the generalization error in Equation 7.

[3] For example, to estimate $\mathbb{E}[Y^0|A = 1]$ we define $\mathcal{D}_a = \{x_i : A_i = 0\}$ and $\mathcal{D}_{pop} = \{x_i : A_i = 1\}$.

Algorithm 1 Adversarial balancing weights

Input: $\mathcal{D}_{pop}, \mathcal{D}_a$

Parameters: classification algorithm d, learning rate α, number of iterations T, loss function l

Output: balancing weight vector $\mathbf{w}$ for $\mathcal{D}_a$

$\mathbf{c} \leftarrow \big[\underbrace{0, 0, \ldots, 0}_{n\text{-times}}, \underbrace{1, 1, \ldots, 1}_{n_a\text{-times}}\big]$

$\mathbf{w} \leftarrow \big[\underbrace{1, 1, \ldots, 1}_{n\text{-times}}, \underbrace{1, 1, \ldots, 1}_{n_a\text{-times}}\big]$

$\mathcal{I}_a \leftarrow [n+1, ..., n+n_a]$ ▷ $\mathcal{I}_a$ contains the indexes where c=1

$w_i \leftarrow \frac{n}{n_a} w_i \ \ , \forall i \in \mathcal{I}_a$ ▷ Assign the same importance for both classes

for T number of iterations **do**

  $\widehat{\mathbf{c}} \leftarrow$ get_predictions$(d, \big[\mathcal{D}_{pop}, \mathcal{D}_a\big], \mathbf{c}, \mathbf{w})$

  $w_i \leftarrow w_i \exp\big(\alpha \cdot l(\widehat{c}_i, 1)\big) \ \ , \forall i \in \mathcal{I}_a$

  $w_i \leftarrow n \frac{w_i}{\sum_{j \in \mathcal{I}_a} w_j} \ \ , \forall i \in \mathcal{I}_a$

return $\frac{n_a}{n} \mathbf{w}[i \in \mathcal{I}_a]$

The choice of the classifier's family and its hyper parameters is important to enable us to approximate the minimal loss defined in Equation 7 with the empirical loss in Equation 8. On the one hand, we would like to reduce the estimation error of L_n due to over-fitting of the classifier. On the other hand, the family of classifiers should be rich enough to distinguish between "non-similar" (weighted) datasets. In Section 6, we describe our experiments with different classification algorithms, ranging from the low-capacity logistic regression to the large-capacity class of neural networks. We test the ability of our framework to tackle the challenge of bias-variance trade-off by applying a preliminary step of hyper-parameter selection using cross-validation, prior to running Algorithm 1.

5 Related work

In recent years, several methods addressed the sensitivity of IPW (e.g.,[11, 7, 14, 33, 1, 18, 32]). Each of these methods provides alternative solutions to the following elementary problems: (i) how to measure the bias between two distributions, and (ii) how to generate weights that minimize it. Some of the methods, (e.g., [7] and [14]), fit propensity score models with imbalance constraints, to guarantee that even if the propensity model is misspecified, the covariates are still balanced. The other algorithms, including the one presented here, are designed to minimize a selected imbalance measure without considering the related propensity scores.

A widely used criterion for assessing the imbalance between two treatment groups is the *standardized difference* in the mean of each covariate [28]. Therefore, it is not surprising that many of the algorithms, such as [11, 14, 1], focus on minimizing the difference between the first-order moments of the covariates or their transformations. The algorithm in [18] uses the *maximum mean discrepancy* (MMD) measure (see [8] for definition, which can account for an infinite number of higher order moments based on kernel methods. Very recently, an independent study [19] presented the idea of using GANs to generate a balancing weights task. However, the objective and optimization algorithm presented in the study differ from the ones we introduce in this paper.

Our algorithm, and all the methods we reviewed above, balance covariates without using outcome data. The balanced data may be used for subsequent causal inference analysis of different outcomes, or integrated with outcome models. Other state-of-the-art methods for causal inference, such as BART [2] and Causal Forests [31] focus on training outcome models that allow causal inference for a specific outcome. Recent works analyze both outcomes and biases for deriving the upper bounds on effect estimation error, combining outcome prediction accuracy and a measure of the bias [16, 29]. These bounds allow learning a new representation of the data that optimizes the combined objective using deep neural networks. In [24], deep networks were also incorporated in a framework that attempts to estimate the joint probabilities that allow counterfactual reasoning using variational auto encoders. A different line of work applies GANs to another task of causal discovery: learning the causal structure that generates the data [22].

6 Experiments

We evaluated our adversarial weighting method with different classifier "plug-ins" on three previously published benchmarks of simulated data. We compared our method to IPW with the same classifier, and tested against more recent methods for balancing weights. We focused on methods that do not use information on the outcome for estimating the weights.

6.1 Experimental setting

The results reported in this section are based on the zero-one loss function. We considered the following "plug-in" classifiers as the discriminator: **LR**: Logistic regression (default parameters by `Scikit-learn`); **SVM**: a support vector machine with RBF kernel (default parameters by `Scikit-learn`); **MLP**: a multilayer perceptron with 1-3 internal layers. The number of nodes in each internal layer is $2p$ where p set to the number of variables in the input layer. The exact number of internal layers is selected as the one that minimizes the zero-one prediction error (generalization error) evaluated in a 5-fold cross-validation procedure; **LR/SVM/MLP**: a classifier that is selected from the previously described classifiers as the one minimizing the zero-one prediction error evaluated in a 5-fold cross-validation procedure.

Note that for the classifiers MLP and LR/SVM/MLP, the configuration is set once before running the weighting algorithms. We used a decaying learning rate α in Algorithm 1: $\alpha_{t+1} = \frac{1}{1+0.5 \cdot t}$ and limited the number of iterations T to 20. Finally, to speed running times we configured the function get_predictions in Step 5 of Algorithm 1 to return train predictions.

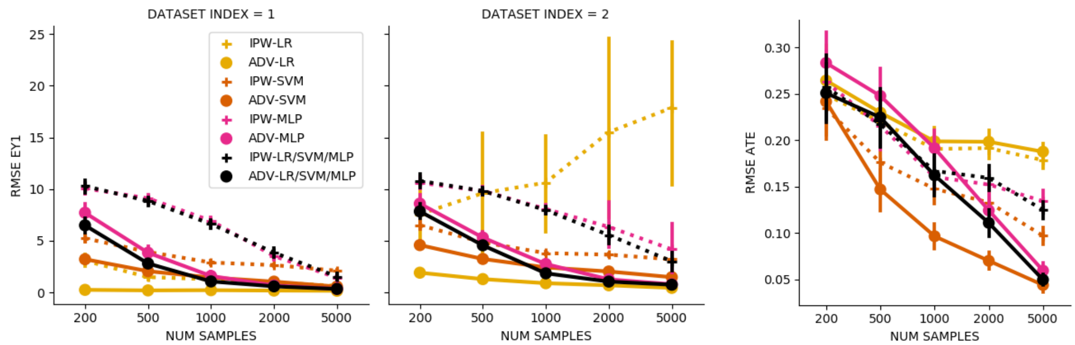
We compared the results of Algorithm 1 to the results obtained by the following weighting methods: **IPW**: The straightforward inverse propensity weighting, without weight trimming or other enhancements. We tested IPW with the same classifiers we used for the adversarial algorithm; **CBPS**: Covariate Balancing Propensity Score (CBPS) [14], using its R package [5]; **EBAL**: Entropy balancing [11], using its R package [12]; **MMD-V1**, **MMD-V2**: An algorithm for minimizing the maximum mean discrepancy (MMD) measure using an RBF kernel [17, 18]. In MMD-V1 the RBF scale parameter was set to 1. MMD-V2 includes a preliminary step for selecting the RBF scale and a regularization parameter [17]. We implemented MMD-V1 and MMD-V2 using the `quadprog` Python package.

6.2 Benchmarks

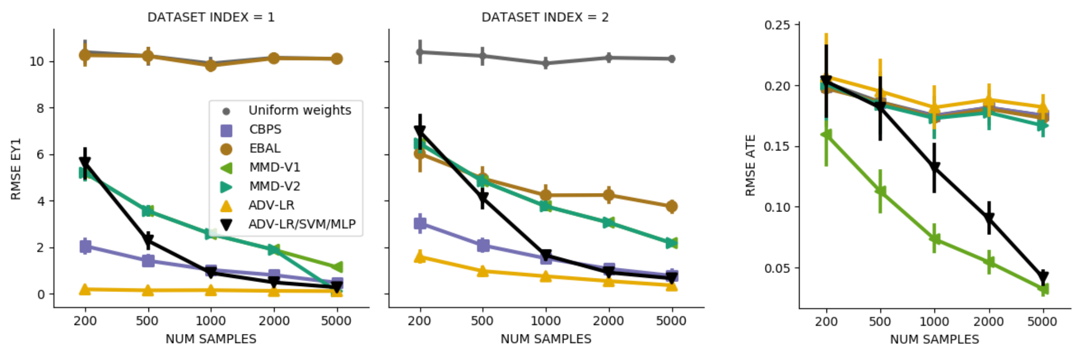
We evaluated and compared the different weighting methods on the following benchmarks:

Kang-Schafer benchmark [20]: The data includes four independently normally distributed covariates: $X_1, X_2, X_3, X_4 \sim N(0,1)$. The outcome covariate Y is generated as $Y = 210 + 27.4X_1 + 13.7X_2 + 13.7X_3 + 13.7X_4 + \epsilon$ where $\epsilon \sim N(0,1)$. The true propensity score is $p(A = 1|X_1, X_2, X_3, X_4) = \text{expit}(-X_1 + 0.5X_2 - 0.25X_3 - 0.1X_4)$. The outcome Y is observed only for $A = 1$. The simulation includes two scenarios. In the first, the covariates (X_1, X_2, X_3, X_4) are observed, while in the second the covariates actually seen, (X_1', X_2', X_3', X_4'), are generated as: $X_1' = \exp(X_1/2)$, $X_2' = X_2/(1 + \exp(X_1)) + 10$, $X_3' = (X_1 * X_3/25 + 0.6)^3$, and $X_4' = (X_2 + X_4 + 20)^2$. As Y is observed only for a biased selection of the data, the task in this benchmark is to estimate the expected potential outcome $E(Y_1)$ for the entire population. In this case we apply Algorithm 1 once to balance the subpopulation of $A = 1$ with the entire population. We generated 5 paired datasets, where each pair corresponds to the 2 scenarios, for data size $n = 200, 500, 1000, 2000, 5000$. Each of the datasets includes 100 random replications. Paired datasets are based on the same randomized covariates (X_1, X_2, X_3, X_4).

Circular benchmark: These simulations are based on the example given in [17] with a minor modification to accommodate estimation of ATE. The simulations follow a scenario with two covariates X_1 and X_2 independently drawn from a uniform distribution on $[-1, 1]$. The true propensity score is $p(A = 1|X_1, X_2) = 0.95/(1 + \frac{3}{\sqrt{2}}\|(X_1, X_2)\|_2)$. The potential outcomes Y^0 and Y^1 are independently normally distributed with means $\|X\|_2^2 - X_1/2 - X_2/2$ and $\|X\|_2^2$, respectively, and with a standard deviation of $\sqrt{3}$. We generated 5 datasets, each with 100 random replications, for this scenario with data size $n = [200, 500, 1000, 2000, 5000]$.



(a) Kang-Schafer benchmark (b) Kallus benchmark



(c) Kang-Schafer benchmark (d) Kallus benchmark

Figure 1: Comparison of weighting algorithms. Horizontal lines represent 95% confidence intervals computing using bootstrapping. **Top:** Adversarial weighting algorithm versus IPW with different classifiers: logistic regression (LR), support vector machine with RBF kernel (SVM), and multi-layer perceptrons MLP. MLP corresponds to MLPS with 1/2/3 layers, respectively chosen by cross-validation. LR/SVM/MLP indicate a preceding step of model selection, prior to running IPW or ADV. **Bottom:** CBPS, EBAL, MMD, and the adversarial algorithm.

ACIC benchmark: The Atlantic Causal Inference Conference (ACIC) benchmark [4] includes 77 datasets, simulated with different treatment assignment mechanisms and outcome models. All the datasets use the same 58 covariates with 4802 observations derived from real-world data. These simulations accounted for various parameters, such as degrees of non-linearity, percentage of patients treated, and magnitude of the treatment effect. Each of the 77 datasets includes 100 random replications independently created by the same data generation process, yielding 7700 different realizations in total. For a complete description of this benchmark, see [4].

6.3 Results

Figure 1 shows the results of Algorithm 1, and the other weighting algorithms described above, on the Kang-Schafer and Circular benchmarks. The top row compares Algorithm 1 to IPW with the same classifier. For all considered classifiers, the adversarial algorithm outperformed its counterpart IPW in most of the tests, in particular on the large sample size. We see that each benchmark had a different classifier that obtained the best results, with LR excelling in the Kang-Schafer benchmark, SVM in the Circular benchmark, and MLP in ACIC. However, in all three benchmarks, LR/SVM/MLP was the second-best performing classifier. Note that in the Kang-Schafer and Circular benchmark, the performance of all classifiers improves with data size, and the performance of all is comparable in the final point (n=5000).

The bottom row adds the results of the other weighing algorithms CBPS, EBAL, MMD-V1 and MMD-V2. As a reference, we selected two classifiers for the adversarial algorithm: LR being the

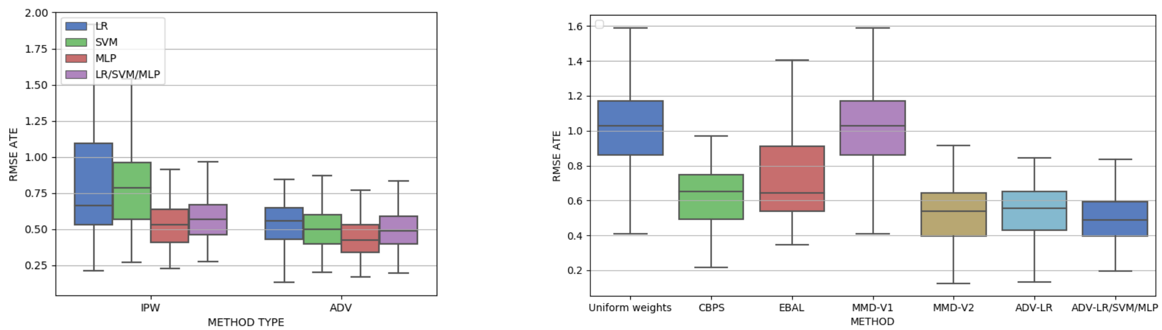


Figure 2: Comparison on ACIC benchmark

simplest classifier and LR/SVM/MLP for its ability to adapt to the data. As shown, in the Kang-Schafer and ACIC benchmarks, ADV-LR outperforms CBPS, EBAL, and both versions of MMD. In the Circular benchmark, MMD-V1 and ADV-SVM outperformed all compared methods, possibly because they both employ Gaussian kernels that can handle the circular contours of the propensity function.

Figure 2 presents the results on the ACIC benchmark. These results also support our previous observation that the adversarial framework is better at exploiting classifiers than the IPW method. This plot also provides some evidence for the robustness of the cross-validation procedure, as ADV-LR/SVM/MLP steadily remains one of the top-performing methods. Finally, even our weakest variant ADV-LR exhibited performance superior to CBPS and EBAL, and results comparable to MMD-V2.

7 Discussion

We introduced a balancing framework using an adversarial approach that exploits the power of classifiers to evaluate biases. Under this framework, we propose a novel non-parametric weighting algorithm that uses exponentiated gradient ascent to search for weights that hamper the discriminator's ability to classify reweighted data. The exponentiated gradient ascent has a desired property that aims in each step to increase the objective, while minimizing the Kullback–Leibler divergence between the original and updated weights [21]. Since the highest entropy is obtained when the weights are uniform, the algorithm is expected to produce weights that remain as close as possible (in an entropy sense) to the uniform base weights. We expect this property to suppress the generation of extreme weights, which lead to high variance estimates. Indeed, we demonstrated that the same classifier, when used by the adversarial algorithm, leads to smaller variance estimates compared to its use for IPW. This setup allows us to plug-in a plethora of classification algorithms into our framework. Although such algorithms may rely on a different optimization criterion than the discriminator's defined loss, they may still be used in practice under the assumption that the two optimization criteria highly correlate. In particular, classifiers that incorporate regularization in their objective may provide a lower expected prediction error for the loss function. In our experiments, we used train predictions to speed up running times. We repeated the experiments after replacing train predictions with test-predictions in 5-fold cross validation (Step 5 of Algorithm 1), and obtained improved results when the discriminator was based on a multilayer perceptron (data not shown). This suggests that using train predictions may lead to less accurate estimations when the discriminator has a high capacity to memorize train data. Finally, we used cross-validation to select the classification algorithm with the lowest estimated generalization error. The experiments we conducted on different benchmarks demonstrated the effectiveness, robustness, and scalability of this approach.

This work sets initial steps for adversarial balancing. There are several frontiers in which our weighting algorithm may be improved. The first is related to optimization issues: should we also train the classifier in small steps, as is done in other GANs frameworks? Can we adapt more recent optimization knowhow to this framework and its objective? Another challenge is to improve the basic selection method presented here for the appropriate classifier family and its hyperparameters. Finally, there have been many advances in GANs since the seminal work in [6], with the introduction of new optimization objectives related to alternative discrepancy measures. It would be interesting to test whether these extensions can be adapted to the balancing weights generation problem in our work.

References

[1] K. C. G. Chan, S. C. P. Yam, and Z. Zhang. Globally efficient non-parametric inference of average treatment effects by empirical balancing calibration weighting. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 78(3):673–700, 2016.

[2] H. A. Chipman, E. I. George, R. E. McCulloch, et al. Bart: Bayesian additive regression trees. *The Annals of Applied Statistics*, 4(1):266–298, 2010.

[3] H. Daume III and D. Marcu. Domain adaptation for statistical classifiers. *Journal of Artificial Intelligence Research*, 26:101–126, 2006.

[4] V. Dorie, J. Hill, U. Shalit, M. Scott, and D. Cervone. Automated versus do-it-yourself methods for causal inference: Lessons learned from a data analysis competition. *ArXiv e-prints: 1707.02641*, 2017.

[5] C. Fong, M. Ratkovic, and K. Imai. Cbps: R package for covariate balancing propensity score. *Comprehensive R Archive Network (CRAN)*, 2014.

[6] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial nets. In *Advances in neural information processing systems*, pages 2672–2680, 2014.

[7] B. S. Graham, C. C. de Xavier Pinto, and D. Egel. Inverse probability tilting for moment condition models with missing data. *The Review of Economic Studies*, 79(3):1053–1079, 2012.

[8] A. Gretton, K. M. Borgwardt, M. Rasch, B. Schölkopf, and A. J. Smola. A kernel method for the two-sample-problem. In *Advances in neural information processing systems*, pages 513–520, 2007.

[9] A. Gretton, A. J. Smola, J. Huang, M. Schmittfull, K. M. Borgwardt, and B. Schölkopf. Covariate shift by kernel mean matching. 2009.

[10] M. U. Gutmann, R. Dutta, S. Kaski, J. Corander, et al. Likelihood-free inference via classification. *arXiv preprint arXiv:1407.4981*, 2014.

[11] J. Hainmueller. Entropy balancing for causal effects: A multivariate reweighting method to produce balanced samples in observational studies. *Political Analysis*, 20(1):25–46, 2012.

[12] J. Hainmueller and M. J. Hainmueller. Package 'ebal'. 2014.

[13] D. G. Horvitz and D. J. Thompson. A generalization of sampling without replacement from a finite universe. *Journal of the American statistical Association*, 47(260):663–685, 1952.

[14] K. Imai and M. Ratkovic. Covariate balancing propensity score. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 76(1):243–263, 2014.

[15] J. Jiang. A literature survey on domain adaptation of statistical classifiers. 2008.

[16] F. Johansson, U. Shalit, and D. Sontag. Learning representations for counterfactual inference. In *International Conference on Machine Learning*, pages 3020–3029, 2016.

[17] N. Kallus. Generalized optimal matching methods for causal inference. *arXiv preprint arXiv:1612.08321*, 2016.

[18] N. Kallus. Balanced policy evaluation and learning. *arXiv preprint arXiv:1705.07384*, 2017.

[19] N. Kallus. Deepmatch: Balancing deep covariate representations for causal inference using adversarial training. *arXiv preprint arXiv:1802.05664*, 2018.

[20] J. D. Kang and J. L. Schafer. Demystifying double robustness: A comparison of alternative strategies for estimating a population mean from incomplete data. *Statistical science*, pages 523–539, 2007.

[21] J. Kivinen and M. K. Warmuth. Exponentiated gradient versus gradient descent for linear predictors. *Information and Computation*, 132(1):1–63, 1997.

[22] M. Kocaoglu, C. Snyder, A. G. Dimakis, and S. Vishwanath. Causalgan: Learning causal implicit generative models with adversarial training. *arXiv preprint arXiv:1709.02023*, 2017.

[23] D. Lopez-Paz and M. Oquab. Revisiting classifier two-sample tests. *arXiv preprint arXiv:1610.06545*, 2016.

[24] C. Louizos, U. Shalit, J. M. Mooij, D. Sontag, R. Zemel, and M. Welling. Causal effect inference with deep latent-variable models. In *Advances in Neural Information Processing Systems*, pages 6449–6459, 2017.

[25] S. Mohamed and B. Lakshminarayanan. Learning in implicit generative models. *arXiv preprint arXiv:1610.03483*, 2016.

[26] P. R. Rosenbaum. Model-based direct adjustment. *Journal of the American Statistical Association*, 82(398):387–394, 1987.

[27] P. R. Rosenbaum and D. B. Rubin. The central role of the propensity score in observational studies for causal effects. *Biometrika*, 70(1):41–55, 1983.

[28] P. R. Rosenbaum and D. B. Rubin. Constructing a control group using multivariate matched sampling methods that incorporate the propensity score. *The American Statistician*, 39(1):33–38, 1985.

[29] U. Shalit, F. D. Johansson, and D. Sontag. Estimating individual treatment effect: generalization bounds and algorithms. In *International Conference on Machine Learning*, pages 3076–3085, 2017.

[30] M. Sugiyama, S. Nakajima, H. Kashima, P. V. Buenau, and M. Kawanabe. Direct importance estimation with model selection and its application to covariate shift adaptation. In *Advances in neural information processing systems*, pages 1433–1440, 2008.

[31] S. Wager and S. Athey. Estimation and inference of heterogeneous treatment effects using random forests. *Journal of the American Statistical Association*, (just-accepted), 2017.

[32] Q. Zhao. Covariate balancing propensity score by tailored loss functions. *arXiv preprint arXiv:1601.05890*, 2016.

[33] J. R. Zubizarreta. Stable weights that balance covariates for estimation with incomplete outcome data. *Journal of the American Statistical Association*, 110(511):910–922, 2015.