Deep Learning of Potential Outcomes

Bernard Koch[1]**,Tim Sainburg**[2]**, Pablo Geraldo Bastias**[1]
Song Jiang[3]**, Yizhou Sun**[3]**, Jacob Foster**[1]
1. UCLA Dept. of Sociology, 2. UCSD Dept. Psychology 3. UCLA Dept. of Computer Science

Abstract

In this paper, we review the rapidly emerging literature on *deep learning for causal inference*. Adoption of deep learning has been slower in areas of science that prioritize interpretable models and evidence of causality (e.g., medicine, epidemiology, and social science). New deep learning models can adjust for confounding in creative ways to estimate unbiased treatment effects and predict counterfactuals. Furthermore, they extend causal inference to new settings where data is high-dimensional and heterogeneous, confounding can be non-linear and time-varying, and/or confounders are encoded in complex non-tabular data like text, networks, or images. To make this survey relatively self-contained, we first provide a brief introduction to representation learning. We then discuss how fundamental deep learning concepts like representation learning, adversarial training, and generative modeling can be used to extend the scope of causal estimation. The review should be accessible to social scientists and data scientsists with an intuitive understanding of supervised machine learning and causal inference. We also supply annotated tutorials that show readers how to implement select models in Tensorflow.

Keywords: deep learning, causal inference, potential outcomes, machine learning.

Contents

Boxes

1. Introduction

In this paper, we systematize an emerging literature for estimating causal effects using deep neural networks. In recent years, both causal inference frameworks and neural machine learning models have seen rapid adoption across science, industry, and medicine. Causal inference has a long tradition in the social sciences (for a basic introduction, see Box 3), but deep learning (and machine learning more generally) is conspicuously underutilized. This review primarily aims to introduce social scientists to an exciting literature within the machine learning community exploring how deep learning might be used to estimate causal effects. Because this literature is growing rapidly, we organize proposed deep causal estimators into four basic categories that reflect the causal estimation strategies employed. We assume the reader has a basic familiarity with causal inference, but no prior knowledge on neural networks, so this paper also aims to provide a broad, conceptual introduction to deep learning and popular deep learning architectures. These concepts are presented in boxes throughout the paper.

The review is organized as follows. We first provide a brief, intuitive primer on deep learning and representation learning for social scientists inexperienced with neural networks. We then review assumptions needed for causal identification when applying deep learning models for causal estimation under a selection on observables strategy. To motivate the typology used to organize deep learning models, we discuss two distinct approaches to causal identification in the selection on observables setting: outcome modeling (including non-parametric matching and balancing approaches) and treatment modeling (i.e., propensity-score based approaches).

In the main body of the paper, we systematize deep learning algorithms into four basic categories. First, we discuss the use of neural networks as plug-in *outcome modelers* for conditional average treatment effects, a technique that is used ubiquitously in this literature and generally combined with other approaches. Second, we explain how representation learning can be described to *balance* covariate distributions. Third we discuss, the usage of neural networks to generate inverse propensity score weights for *treatment modeling*. The fourth section describes *adversarial training* regimes inspired by Generative Adversarial Networks (GANs) to build generative models of counterfactual outcome distributions or improve performance of

the above-described techniques (21). In the final section of the paper, we discuss models that extends estimation under selection on observables to different settings: data with time-varying treatments or scenarious where confounders, mediators, and treatments might be latently represented in graphs, text, or images. We conclude by presenting the pros and cons of these deep causal estimators compared to established approaches in the social sciences. To aide reachers in implementing approaches and learn how to develop deep neural networks themselves, extensive tutorials are provide for implementing these models on social science data in Tensorflow 2 available at https://github.com/kochbj/Deep-Learning-for-Causal-Inference.

1.1. Why use deep learning for causal inference?

Deep learning estimators present several advantages compared to existing traditional and machine learning estimators in social scientists' arsenal:

- **(Nearly) non-parametric modeling of relationships between covariates, treatments, and outcomes.** Using generalized linear models for causal estimation requires the analyst to make strong assumptions about the functional relationship between covariates (outcome predictors, confounders, mediators, colliders), treatment assignment, and outcomes. All machine learning estimators relax these parametric assumptions by exhaustively exploring non-linear interactions that may correlate covariates, treatment, and outcomes. However, neural networks have two distinct advantages compared to other machine learning approaches to causal inference (e.g. decision tree/forest-based approaches, LASSO regression, support vector machines). First neural networks naturally extract relevant information from covariates through representation learning (discussed extensively below), allowing the analyst to incorporate dozens or hundreds of observed covariates that predict/confound treatment assignment and outcome. Second at expense of usability, the flexibility of neural networks allow analysts to extend nearly non-parametric estimation to scenarios where not many viable non-parametric estimation strategies exist (e.g., observational data with time varying treatments, data with multiple treatments).

- **State of the art estimation of heterogeneous treatment effects.** A recent trend in causal social science has been an increasing focus on how treatment effects vary across categories, rather than on average in populations. The emergence of machine learning causal estimators has been a driving force behind this trend. Beacuse deep neural networks can theoretically approximate any continuous function, neural networks appear to substantially outperform other machine learning approaches to causal inference for the estimation of heterogeneous/conditional treatment effects with respect to bias, in both simulated and real data.

- **Moving from inference to prediction.** Deep causal estimators appear to perform well not just on in-sample inference, but also out-of-sample prediction. Predictive modeling would allow social scientists to train a model on observational data where treatment of some units is observed, and estimate effects in new datasets where no datasets are observed.

- **Causal inference in quantitative data, text, images, and graphs.** Through representation learning, deep neural network models can adjust for confounding not just in quantitative data, but also extract latent confounders encoded in text, networks, and graphs. To motivate the use of these models we discuss some example causal scenarios that can be addressed by using deep neural estimators:

  Traditional Data. The companion tutorials use a naturalistic simulation based on the Infant Health and Development Program (IHDP) example from Hill (27). One of the goals of the original IHDP study was to estimate the causal effect of specialized childcare interventions on cognitive outcomes for premature infants. The treatment (T) is attendance at a special child development center for premature infants. The outcome is some measure of cognitive development for infants after (Y). Measured covariates (X) such as socioeconomic status are predictive of both seeking treatment and cognitive development.

  Text. As a motivating example, Veitch et al. (64) consider the effect of the author's reported gender (T) on the number of upvotes a Reddit post receives (Y). However

gender may also "affect the text of the post, e.g., through tone, style, or topic choices, which also affects its score [(X)]." Controlling for a representation of the text would allow the analyst to more accurately estimate the causal effect of gender.

Images. Todorov et al. (62) showed that split second-judgments of a politician's competence (T) from pictures (X) of their face is predictive of their electability (Y). When attempting to replicate this study using machine learning classifiers rather than human classifiers, Joo et al. (36) suggest that the age of the face (Z) is a not-so-obvious confounder: while older individuals are more likely to appear competent, they are also more likely to be incumbents. Even if age is unknown, using neural networks to control for confounders implicitly encoded in the image (like age) could reduce bias.

Networks. Nagpal et al. (47) explore the question of which types of prescription opioids (e.g., natural, semi-synthetic, synthetic) (T) are most likely to cause long term addiction (Y). Because of predisposition to different injuries, type of employment (X) could be a common cause of both treatment and outcome. Suppose job type is unobserved, but we know that patients are likely to associate with coworkers through homophily. To capture some of the effects of this latent unobserved confounder, analysts might choose to control for a representation of the patient's position in their Twitter network when estimating the causal effect.

While the main body of this review focuses on algorithm for causal inference/prediction from social scientist's primary use case of traditional quantitative data, models for dealing with non-traditional data are discussed in Section 5.

2. Primer on Deep Learning

2.1. Artificial Neural Networks

Artificial neural networks (ANN) are statistical models inspired by the human brain (Brand et al.). In an ANN, each "neuron" in the network takes the weighted sum of its inputs (the outputs

of other neurons) and transforms them using a twice differentiable, non-linear function (e.g. sigmoid, rectified linear unit) that outputs a value between 0 and 1 if the transformed value is above some threshold. Neurons are arrayed in layers where an input layer takes and outputs the raw data, and each neuron in subsequent layers take the weighted sum of outputs in previous layers as input. An "output" layer contains a neuron for each of the predicted outcomes with transformation functions appropriate to those outcomes. For example, a regression network that predicts one outcome will have a single output neuron without a transformation function so that it produces a real number. A regression network without any hidden layers corresponds exactly to a generalized linear model (Fig. 1A). When additional "hidden" layers are added between the input and output layers, the architecture is called a **feed-forward network** or **multi-layer perceptron** (Fig. 1B). A neural network with multiple hidden layers is called a "deep" network, hence the name "deep learning" (42). A neural network with a single, large enough hidden layer can theoretically approximate any continuous function (14).

Box 1: Reading Machine Learning Papers: Computational Graphs and Loss Functions

Within the machine learning literature, novel algorithms are often presented in terms of their computational graph and loss function. A computational graph (not to be confused with a causal graph) uses arrows to depicts the flow of data from the inputs of a neural network, through parameters, to the outputs. Layers of neurons or specialized sub-architectures are often generically abstracted as shapes. In our diagrams, we use purple to represent observables, orange for representation layers of the network, white for produced outputs, and red and blue for outcome modeling layers. Operations that are computed *after* prediction (i.e., for which an error gradient is not calculated) are shown with dashed lines (e.g., plug-in estimation of causal estimands).

Along with the architecture, the loss function of a neural network is the primary means for the analyst to dictate what types of representations a neural network learns and what types of outputs it produces. In multi-task learning settings, we denote joint loss functions for an entire network as a weighted sum of the losses for substituent tasks and modules. These specific losses are weighted by hyperparameters. For example, we might weight the joint loss for a network that predicts outcomes and propensity scores as:

$$\arg\min_{h,\pi} \mathcal{L} = \mathcal{L}_h + \lambda \mathcal{L}_\pi = MSE(Y, \hat{Y}) + \lambda BCE(T, \hat{\pi}(X, T))$$

where $\hat{\pi}(X, T)$ is the predicted propensity score, λ is a hyperparameter and MSE and BCE stand for mean squared error and binary cross entropy (i.e., log loss), common losses for regression and binary classification respectively.

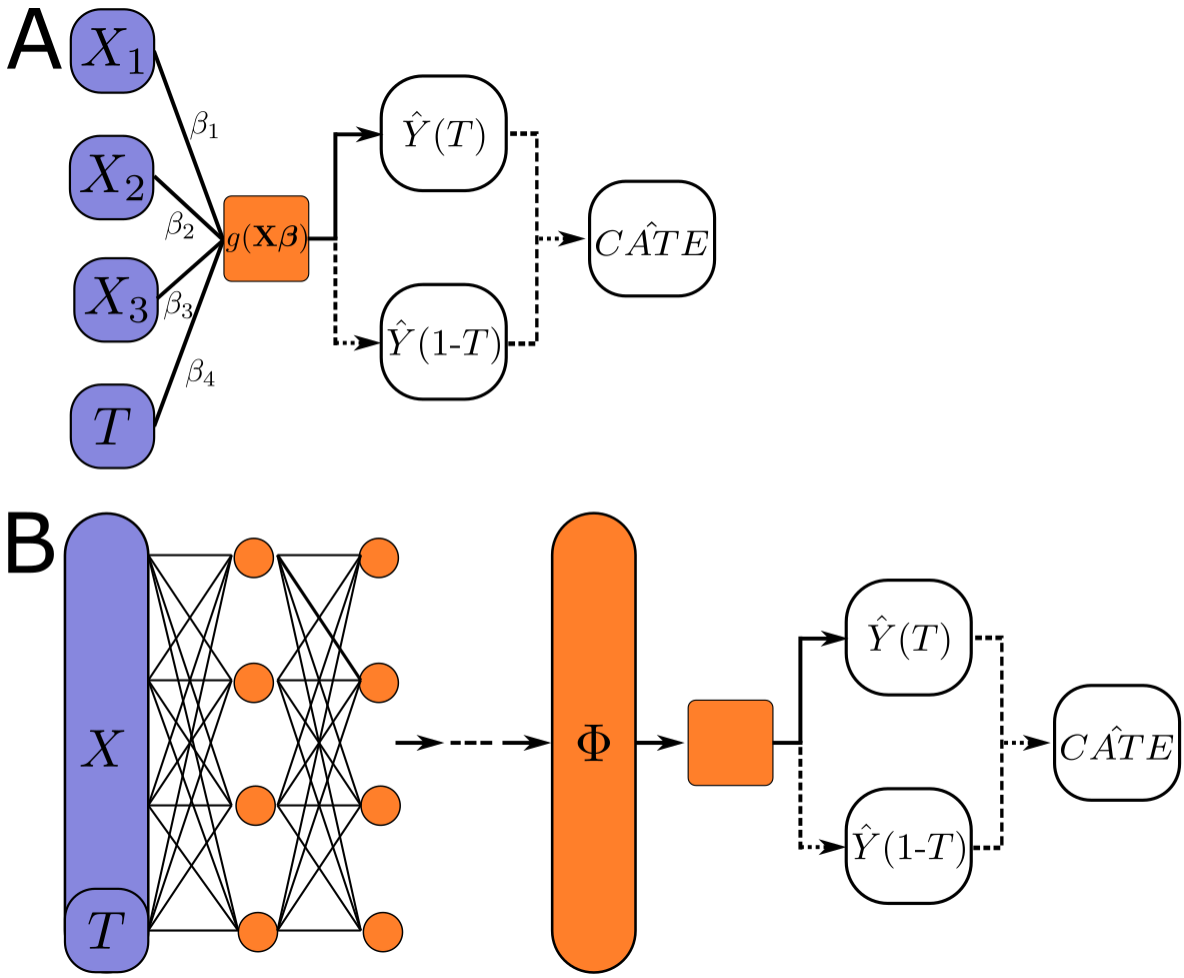


Figure 1: **A: Generalized linear model represented as a computational graph.** Observable covariates X_1, X_2, X_3 and treatment status T depicted in purple. Each of the lines between the purple inputs and the orange box represents a parameter (i.e., a β in a generalized linear model equation). The orange box is an "output neuron" that sums it's weighted inputs, performs a transformation g (the link function in GLM; in this case the identity function), and predicts the conditional outcome $\hat{Y}(T)$. Instead of theoretically interpreting these parameters from an inferential statistics perspective, machine learning approaches typically use the predicted observed and unobserved potential outcomes for plug-in estimation of causal estimands (e.g., the conditional average treatment effect $C\hat{A}TE$). After training, setting T to $1 - T$ for each observation can predict the unobserved potential outcome $\hat{Y}(1 - T)$. Because this operation occurs after prediction and does not feed a gradient back to the network to optimize the parameters, it is depicted here with a dotted line. Plug-in calculation of $C\hat{A}TE$ is similarly shown with a dotted line.

B: Feed-forward neural network. In a feed-forward neural network, additional fully connected (parameterized) layers of neurons are added between the inputs and output neuron. The size of the input covariates and hidden layers are generically abstracted as boxes. The final hidden layer before the output neuron is denoted Φ because the hidden layers collectively encode a representation function (see section 2.2). In causal inference settings,this architecture is sometimes called an S(ingle)-learner because one feed-forward network learns to predict both potential outcomes.

Neural networks are trained to predict their outcomes by optimizing a **loss function** (also called objective or cost function). During training, error in the loss function is distributed proportionally (i.e. **backpropagated**) to weight parameters in previous layers in the network. An optimizer, such as the stochastic gradient descent algorithm or the currently popular ADAM algorithm (41), then moves the parameters in the opposite direction of this error gradient. Neural networks first rose to popularity in the 1980s but fell out of favor compared to other machine learning model families (e.g., support vector machines) due to their expense of training. By the late 2000s, the improvements to backpropagation, advances in graphical processing units (i.e., graphic cards), and access to larger datasets, collectively enabled a deep learning revolution where ANNs began to significantly outperform other model families. Today, deep learning is the hegemonic machine learning approach in industries and fields other than social science. For further discussion on how deep learning models are trained and regularized in a supervised machine learning framework, see Box 2.

Box 2: Training and Regularizing Supervised Deep Learning Models

Supervised Training. In supervised learning, data is split into a training set and a validation set. Model parameters are optimized on the training set before out-of-sample performance is assessed on the validation set. Performance on the validation set is typically used to choose hyperparameter settings. Deep learning models differ from other machine learning approaches in that the loss function is typically non-convex and trained models may not converge on the same optima. Thus unlike other machine learning approaches which train first on the complete training set and are then evaluated subsequently on the complete validation set, neural networks are typically trained on small batches of training data at a time. Because a batch of data is only a sample of a sample (the training dataset), the optimizer only adjusts weight parameters by a fraction of the error gradient (the **learning rate**) to avoid overfitting. When a model has cycled through a set of batches that cover the complete training set, this is called a training **epoch**. After each training epoch, the network is typically tested over a validation epoch (i.e. a complete iteration of batches for the validation set) without updating the weights.

Regularization. Neural networks are highly susceptible to overfitting, and early stopping of training once the validation error begins to rise is a fundamental regularization technique. Other common regularization techniques include **weight decay** (i.e., ℓ^2 norm, ridge, or Tikhonov) penalties on the weight parameters, dropout of neurons during training, and batch normalization. **Dropout** is a regularization technique in deep learning where certain nodes are randomly "dropped out" from training during a given epoch (59). The general idea of dropout is to force two neurons in the same layer

to learn different aspects of the covariate/feature space and reduce overfitting. **Batch normalization** is another regularization technique applied to a layer of neurons (30). By standardizing (i.e. z-scoring) the inputs to a layer on a per-batch basis and then rescaling them using trainable parameters, batch normalization smooths the optimization of the loss function.

2.2. Representation Learning and Multitask Learning

One comparative advantage of deep learning over other machine learning approaches has been the ability of ANNs to encode and automatically compress informative features from complex data into flexible, relevant "**representations**" or "embeddings" that make downstream supervised learning tasks easier (20; 6). While other machine learning approaches may also encode representations, they often require extensive pre-processing to create useful features for the algorithm. Through the lens of representation learning, a geometric interpretation of the role of each layer in a supervised neural network is to transform it's inputs (either raw data or output of previous layers) into a typically lower (but possibly higher) dimensional vector space. As a means to share statistical power, encoded representations can also be jointly learned for two tasks at once in **multi-task learning**.

The simplest example of a representation might be the final layer in a feed-forward network, where the early layers of the network can be understood as non-linearly encoding the inputs into an array of latent linear features for the output neuron (20) (Fig. 1B). A famous example of representation learning is the use of neural networks for face detection. Examining the representations produced by each layer of these networks shows that subsequent layer seems to capture increasingly abstract features of a face (first edges, then noses and eyes, and finally whole faces) (42). A more familiar example of representation learning to social sciences might be word vector models like Word2Vec (45). Word2Vec is a two-layer neural network where words that are semantically similar are closer together in the representation produced by the hidden layer of the network.

The novel contribution of deep learning to causal estimation is the proposal that a neural network can learn a function Φ *that produces representations of the covariates deconfounded from*

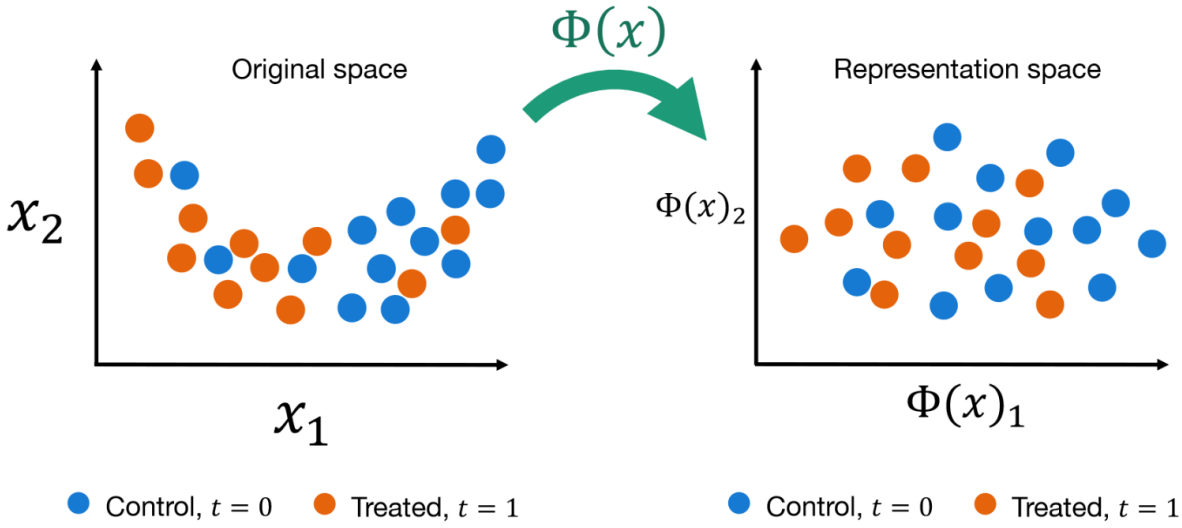


Figure 2: **Balancing through representation learning.** The promise of deep learning for causal inference is that a neural network encoding function Φ can transform the treated and control covariate distributions into a representation space such that they are indistinguishable. Used with permission from Johansson and Shen (32).

outcome and/or treatment. Fundamentally, the idea is that Φ can transform the treated and control covariate distributions into a representation space such that they are indistinguishable (Fig. 2). To ensure that these representations are also still predictive of the outcome (multi-task learning), multiple loss functions are generally applied simultaneously to balance these objectives. This approach is applied in a majority of the algorithms presented in the main body of this review (Section 4).

3. Causal Identification and Estimation Strategies

3.1. Identification of Causal Effects

The papers described in this review are primarily framed within the Potential Outcomes causal framework (Neyman-Rubins causal model) (54; 29). This framework is concerned with identifying the "potential outcomes" of each unit in the sample, had they both received treatment and not received treatment. However because units can only receive one treatment regime in reality, it is not possible to observe both potential outcomes for each individual

(often termed "the fundamental problem of causal inference"). While we cannot thus identify individual treatment effects, causal inference frameworks allow us to probabilistically estimate average treatment effects (ATE) and average treatment effects conditional on select covariates (CATE) across samples of treated and control units. Within this literature, the motivation of many papers is to present algorithms that can both infer CATEs from experimental and observational data, but also predict them for out-of-sample units where treatment status is unknown.

Box 3: Basic Introduction to Causal Inference

Correlation does not equal causation, and causal statistics is concerned with the identification of causal relationships between random variables. Many causal questions we would like to ask about social data can be framed as counterfactual questions with the general format: "What would have been the outcome Y for a unit with X characteristics, if T had happened or not happened?" Equivalently, this can be reworded to "What is the causal effect of T on Y for units with characteristics X.

Causal inference frameworks primarily take a random-controlled experiment, where each unit with covariate or features X is randomly assigned to the treatment or control groups and outcome Y is subsequently measured, as the ideal approach to answering this type of question. But in many scenarios it is prohibitively expensive (e.g., targeting a demographic group who is under-represented or has limited internet access) or morally prohibitive (e.g., randomly assigning students to attend college or not) to collect experimental data. In these cases, we can statistically adjust observational data (e.g., survey data on college attendance) to ask causal questions. The methods described in this paper are designed to answer counterfactual questions with primarily non-experimental observational data.

Three causal statistical frameworks have been independently introduced in social statistics(54; 29), epidemiology (51; 52; 26), and computer science (19; 49). The goal of these causal frameworks is to describe and correct for biases in data or study design that would prevent one from making a true causal claim. If these biases are correctable and the causal effect can be uniquely expressed in terms of the distribution of observed data, then we say that the causal effect is *identifiable* (40). If a causal effect is identifiable, we can use statistical tools that correct for identified biases to *estimate* the causal effect (e.g., generalized linear models, g-computation, deep learning).

The focus of the algorithms presented in this paper is on estimating causal effects while correcting for *confounding bias*. A confounding covariate/feature is one that is correlated with both the treatment and the outcome, misleadingly suggesting that the treatment has a causal effect on the outcome, or obscuring a true causal relationship between the treatment and outcome. Often times, the confounder is a cause of the treatment *and* outcome. As an example, estimating the causal effect of attending college (treatment) on adult income (outcome) requires controlling for the fact that parental income may be a common cause of both college attendance and adult income.

The ATE is defined as:

$$ATE = \mathbb{E}[Y(1) - Y(0)]$$

where $Y(1)$ and $Y(0)$ are the potential outcomes had the unit received or not received the treatment, respectively. The CATE is defined as,

$$CATE = \mathbb{E}[Y(1) - Y(0)|X = x]$$

where X is the set of selected, observable covariates, and $x \in X$.

Within the machine learning literature on causal inference surveyed here, the primary strategy for causal identification is **selection on observables**. A challenge to identifying causal effects is the presence of confounding relationships between covariates associated with both the treatment and the outcome. The key assumption allowing selection on observables to isolate causal effects in the presence of confounders is:

1. **Conditional Ignorability/Exchangability** The potential outcomes $Y(0)$, $Y(1)$ and the treatment T are conditionally independent given X,

$$Y(0), Y(1) \perp\!\!\!\perp T|X$$

Conditional Ignorability specifies that there are no unmeasured confounders that affect both treatment and outcome outside of those in the observed covariates/features X. Additionally X may contain predictors of the outcome, but should not contain instrumental variables or colliders within the conditioning set.

The other standard assumptions necessary for causal inference are:

2. **Consistency/Stable Unit Treatment Value Assumption (SUTVA)**. Consistency specifies that when a unit recieves treatment, we observe the potential outcome. Moreover, the response of any unit does not vary with the treatment assignment to other units (i.e., no network

effects), and the form/level of treatment is homogeneous and consistent across units,

$$T = t \rightarrow Y = Y(T)$$

3. **Overlap** In any context $x \in X$, any treatment $t \in \{0, 1\}$ has a non-zero probability of being observed in the data,

$$\forall x \in X, t \in \{0, 1\} : p(T = t|X = x) > 0$$

4. An additional assumption often invoked at the interface of identification and estimation using neural networks is:

Invertability

$$\Phi^{-1}(\Phi(X)) = X$$

In words, there must exist an inverse function of the representation function Φ encoded by a neural network that can reproduce X from representation space. This is required for the Conditional Ignorability assumption to hold when using representation learning.
For reference, we describe the full notation used within the review in Box 4.

Box 4: Notation for Causal Inference and Estimation

We use uppercase to denote general quantities and lowercase to denote specific quantities for individual units.

Causal identification

- Observed covariates/features: X
- Potential outcomes: $Y(0)$ and $Y(1)$
- Treatment: T
- Average Treatment Effect: $ATE = \mathbb{E}[Y(1) - Y(0)]$
- Conditional Average Treatment Effect: $CATE = \mathbb{E}[Y(1) - Y(0)|X = x]$

Deep learning estimation

- Predicted outcomes: $\hat{Y}(0)$ and $\hat{Y}(1)$
- Outcome modeling functions: $\hat{Y}(T) = h(X, T)$
- Representation functions: $\Phi(X)$ (producing representations ϕ)
- Propensity score function: $\pi(X, T) = P(T|X)$ (where $\pi(X, 0) = 1 - \pi(X, 1)$)

- Loss functions: $\mathcal{L}(true, predicted)$
- Loss hyperparameters: λ, α, β
- Estimated CATE: $\hat{cate} = (1-2t)(\hat{y}(t) - \hat{y}(1-t))$
- Estimated ATE: $\hat{ATE} = \frac{1}{n}\sum_{i=1}^{n} \hat{cate}_i$

Beyond the ATE and $CATE$ there is an additional metric commonly used in the machine learning literature, first introduced by Hill (27) called the *Precision in Estimated Heterogeneous Effects (PEHE)*. PEHE is the average error across the predicted $CATE$s.

- Precision in Estimated Heterogeneous Effects: $PEHE = \frac{1}{N}\sum_{i=1}^{N}(cate_i - \hat{cate}_i)^2$

Beyond being a metric, the $PEHE$ score has theoretical significance in the formulation of generalization bounds within this literature (56; 33; 34; 69).

3.2. Estimation of Causal Effects

Once a strategy for isolating causal effects from available data has been identified (arguably the harder and more important part of causal inference), statistical methods can be used to estimate causal effects. Innovative deep learning methods for estimation of causal effects are the focus of this review. Below we briefly review three traditional estimations strategies for removing confounding bias to motivate our systematization of deep learning models. First, we discuss outcome modeling approaches that adjust for the correlation between covariates and outcome using generalized linear models. Next, we consider balancing the covariate distributions through non-parametric matching. Finally, we discuss inverse propensity score weighting to remove the correlation between covariates and treatment.

Outcome Modeling

Assuming the treatment effect is constant across covariates/features or the probability of treatment is constant across all covariates/features (both improbable assumptions), the simplest consistent approach to estimating the ATE is to regress the outcome on the treatment indicator and covariates using a linear model.[1] The ATE is then the coefficient of the treatment indicator. Without loss of generality, we call outcome models of this nature, linear or

[1] Another outcome modeling approach that could be used to estimate the outcome, not discussed here, is g-computation (51; 26).

non-linear, h:

$$\hat{Y} = h(X, T)$$

A slightly more sophisticated semi-parametric approach to **outcome modeling**, used widely in the application of machine learning to causal inference, is to use $h(X, T)$ to impute $\hat{Y(1)}$ and $\hat{Y(0)}$, and calculate the CATE for each unit as a plug-in estimator:

$$\hat{cate} = \hat{y(1)} - \hat{y(0)}$$

and the ATE as:

$$\hat{ATE} = \frac{1}{n}\sum_{i=1}^{n} \hat{cate}_i$$

Because different models make different assumptions, it is not uncommon to combine outcome modeling and treatment modeling estimators to create **doubly-robust** estimators. For example, one of the most widely used doubly-robust estimators is the Augmented-IPW (AIPW) estimator.

(1)
$$\hat{ATE} = \frac{1}{n}\sum_{i=1}^{n}\{[\frac{T_i Y_i}{\hat{\pi}(X_i)} - \frac{(1-T_i)Y_i}{1-\hat{\pi}(X_i)}] - \frac{(T_i - \hat{\pi}(X_i))}{\hat{\pi}(X_i)(1-\hat{\pi}(X_i)} \cdot [(1-\hat{\pi}(X_i))\hat{\mathbb{E}}[Y(0)_i|X_i] + \hat{\pi}(X_i)\hat{\mathbb{E}}[Y(0)_i|X_i]]\}$$

The first term is the IPW estimator, the second term balances between the first term and the third term, and the third term is the propensity weighted semi-parametric outcome regression. This estimator is unbiased if the IPW and regression estimators are consistently estimated (18). However the model is attractive because it will be consistent if *either* the propensity score is correctly specified or the regression models are consistently specified. Many of the algorithms introduced below combine outcome regression with some type treatment modeling using multi-task learning for doubly robustness.

Non-Parametric Matching

Lastly, another common approach is balancing the treated and control covariate distributions though matching. Matching requires the analyst to select a distance measure that captures the difference in observed covariate distributons between two units Austin (5). Units with treatment status T can then be matched with one or more counterparts with treatment status $1-T$ using a variety of algorithms Stuart (61). In a one-to-one matching scenario where each treated unit has an otherwise identical untreated counterpart, matching induces correlations in the data exactly opposite of the confounding correlation between X and T Mansournia et al. (44).

Treatment Modeling

A common treatment-modeling strategy is **inverse propensity score weighting (IPW)**. In IPW, units are weighted on their inverse propensity to receive treatment. Without loss of generality, we call the propensity function π. The propensity score is calculated as the probability of receiving treatment conditional on covariates:

$$\pi(X,T) = P(T|X)(\text{where, } \pi(X,0) = 1 - \pi(X,1))$$

The simplest IPW estimator of the ATE is then:

$$\hat{ATE} = \frac{1}{n}\sum_{i=1}^{n}\{\frac{t_i y_i}{\hat{\pi}(x_i,t_i)} - \frac{(1-t_i)y_i}{1-\hat{\pi}(x_i,t_i)}\} \tag{2}$$

IPW weighting is attractive because the true propensity score π is an unbiased estimator of the ATE, and the IPW is consistent if π is estimated consistently (53; 18).

Double Robustness

Because different models make different assumptions, it is not uncommon to combine outcome modeling with treatment modeling or matching estimators to create **doubly-robust** estimators. For example, one of the most widely used doubly-robust estimators is the Augmented-IPW

(AIPW) estimator.

(3)

$$\hat{ATE} = \frac{1}{n}\sum_{i=1}^{n}\{[\frac{T_iY_i}{\hat{\pi}(X_i)} - \frac{(1-T_i)Y_i}{1-\hat{\pi}(X_i)}] - \frac{(T_i-\hat{\pi}(X_i))}{\hat{\pi}(X_i)(1-\hat{\pi}(X_i)}\cdot[(1-\hat{\pi}(X_i))\hat{\mathbb{E}}[Y(0)_i|X_i] + \hat{\pi}(X_i)\hat{\mathbb{E}}[Y(0)_i|X_i]]\}$$

The first term is the IPW estimator, the second term balances between the first term and the third term, and the third term is the propensity weighted semi-parametric outcome regression. This estimator is unbiased if the IPW and regression estimators are consistently estimated (18). However the model is attractive because it will be consistent if *either* the propensity score is correctly specified or the regression models are consistently specified. Many of the algorithms introduced below combine outcome regression with some type treatment modeling using multi-task learning for doubly robustness.

4. Four Different Approaches to Deep Causal Estimation

The architectures proposed in the deep learning literature draw inspiration from existing approaches to estimation under selection on observables: outcome modeling via deep regression, balancing via representation learning, and IPW adjustment after representation learning. Nearly every algorithm discussed below contains some form of outcome modeling, and most contain some form of representation learning. In addition to these three strategies we describe an approach unique to deep learning: generative modeling. Generative models estimate the joint distribution of covariates, treatment, and outcome and/or modeling of the treatment effect or counterfactual distribution. This section also describes other uses of GAN-like adversarial training to enhance performance of other deep causal estimators. Throughout the review, algorithms are presented via their loss functions and architectures (see Box 1).

4.1. Deep Outcome Modeling

Because one potential outcome is always unobserved, it is not possible to apply supervised learning models to directly learning treatment effects. Across econometrics, biostatistics, and machine learning, a common approach to this challenge has been to instead use machine learning to model the potential outcomes individually and use plug-in estimators for treatment

effects (11; 63; 67) . As with linear models, a single neural model can be trained to learn both potential outcomes ("S[ingle]-learner") (Fig. 1B), or two independent models can be trained to learn each potential outcome (a "T-learner") (34) (Fig. 3A). In both cases, the neural network estimators would be feed-forward networks tasked with minimizing the MSE in the prediction of observed outcomes. The joint loss function for a T-learner can be notated:

$$\mathcal{L}(Y, h(X,T)) = MSE(Y, h(X,0)) + MSE(Y, h(X,1)) \tag{4}$$

After training, we will get two predictions: $\hat{Y}(T)$ and $\hat{Y}(1-T)$. We can plug-in these predictions to estimate the $CATE$ for each unit,

$$\hat{cate} = (1-2t)(\hat{y}(t) - \hat{y}(1-t))$$

and the average treatment effect as,

$$\hat{ATE} = \frac{1}{n}\sum_{i=1}^{n} \hat{CATE}_i$$

Nearly all of the models described below combine this plug-in outcome modeling approach with other forms of treatment adjustment.

4.2. Balancing through Representation Learning

Balancing is a treatment adjustment strategy that aims to deconfound the treatment from outcome by forcing the treated and control covariate distributions closer together (35). The novel contribution of deep learning to the selection on observables literature is the proposal that a neural network can transform the covariates into a representation space Φ such that the treated and control covariate distributions are indistinguishable (Fig. 2). To encourage a neural network to learn balanced representations, Shalit et al. (56) (the seminal paper in this literature) propose a simple two-headed neural network called Treatment Agnostic Regression

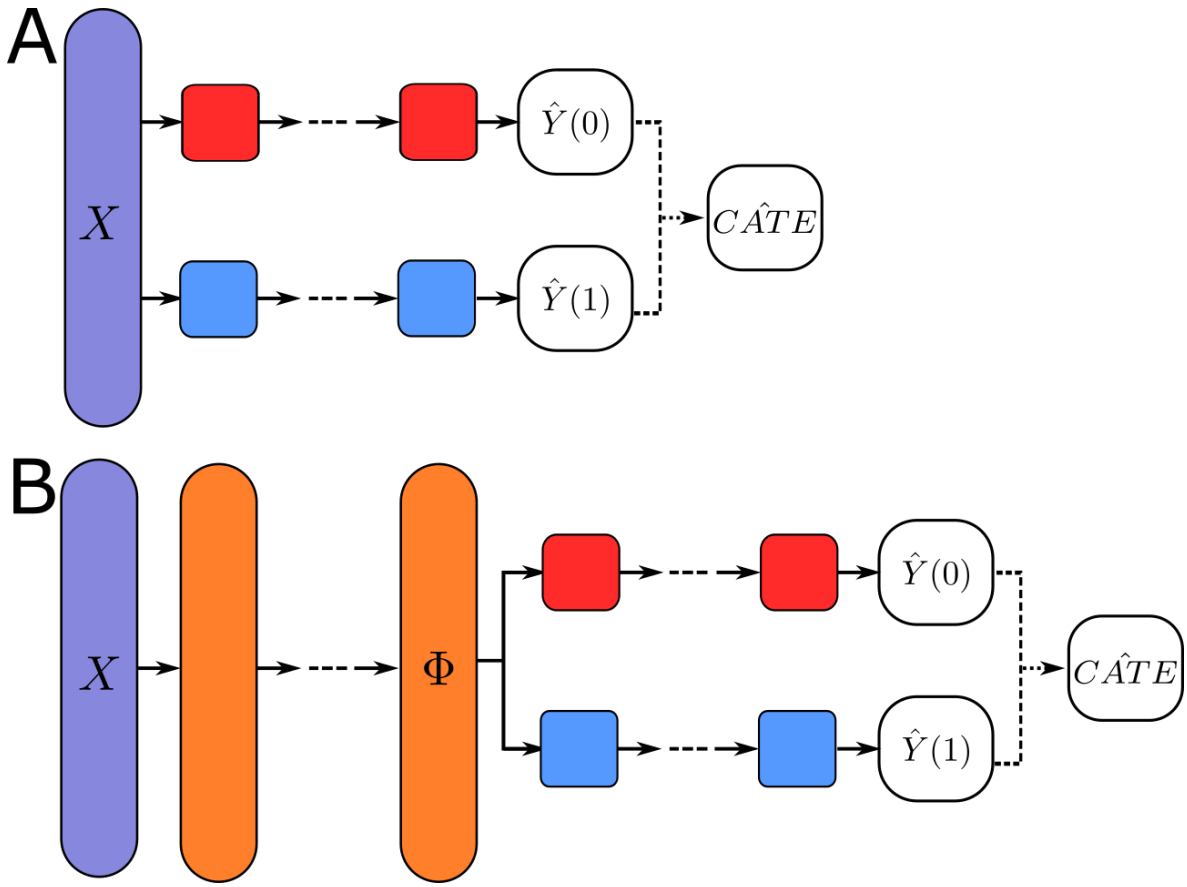


Figure 3: A. **T-learner.** In a T-learner, separate feed-forward networks are used to model each outcome. We denote the function encoded by these outcome modelers h. B. **TARNet.** TARNet extends the T-learner with shared representation layers. The motivation behind TARNet (and further elaborations of this model) is that the multi-task objective of accurately predicting both the treated and control potential outcomes forces the representation layers to learn a balancing function Φ such that the $\Phi(X|T=0)$ and $\Phi(X|T=1)$ are overlapping distributions in representation space.

Network (TARNet) that extends the outcome modeling T-learner with shared representation layers (Fig. 3B). Each head models a separate potential outcome. One head learns the function $\hat{Y}(1) = h(\Phi(X), 1)$, and the other head learns the function $\hat{Y}(0) = h(\Phi(X), 0)$. Both heads backpropagate their gradients to shared representation layers that learn $\Phi(X)$. The idea is that these representation layers must learn to balance the data because they are tasked with predicting both outcomes.

The complete objective for the network is to minimize the parameters of h and Φ for all n units in the training sample such that,

$$\arg\min_{h,\Phi} \frac{1}{n} \sum_{i=1}^{n} MSE(y_i(t_i), \underbrace{h(\Phi(x_i), t_i)}_{\hat{y}_i(t_i)}) + \lambda \underbrace{\mathcal{R}(h)}_{L_2} \tag{5}$$

where $\mathcal{R}(h)$ is a model complexity term (e.g., for L_2 regularization) and λ is a hyperparameter chosen by the user.

Extending Representation Balancing with IPMs

Beyond receiving outcome modeling gradients for both potential outcomes, the authors have subsequently extended TARNet with additional losses that explicitly encourage balancing by minimizing a statistical distance between the two covariate distributions in representation space called an integral probability metrics (46). [2] Johansson et al. (35); Shalit et al. (56); Johansson et al. (33) propose two possible IPMs, the Wasserstein distance and the maximum mean discrepancy distance (MMD) for use in these architectures.

The Wasserstein or "Earth Mover's" distance fits an interpretable "map" (i.e. a matrix) showing how to efficiently convert from one probability mass distribution to another. The Wasserstein distance is most easily understood as an optimal transport problem (i.e., a scenario where we want to transport one distribution to another at minimum cost). The nickname "Earth mover's distance" comes from the metaphor of shoveling dirt to terraform one landscape into

[2] Zhang et al. (69) criticize the usage of IPMs because they make no restrictions on the moments of the transformed distributions. Thus while the covariate distributions may have a high percentage of overlap in representation space, this overlap may be substantially biased in unknown ways.

another. In the idealized case in which one distribution can be perfectly transformed into another, the Wasserstein map corresponds exactly to a perfect one-to-one matching on covariates strategy (37).

The MMD is the normed distance between the means of two distributions, after a kernel function ϕ has transformed them into a high-dimensional space called a reproducing kernel Hibbert Space (RKHS) (22). The MMD with an L^2 norm in RKHS $\mathcal{H}$ can be specified as:

$$MMD(P,Q) = ||\mathbb{E}_{X\sim P}\phi(X) - \mathbb{E}_{X\sim Q}\phi(X)||^2_{\mathcal{H}} \tag{6}$$

The metric is built on the idea that there is no function that would have differing Expected Values for P and Q in this high-dimensional space if P and Q are the same distribution (28). The MMD is inexpensive to calculate using the "kernel trick" where the inner product between two points can be calculated in the RKHS without first transforming each point into the RKHS.[3]

When an IPM loss is applied to the representation layers in TARNet, the authors call the resulting network "CounterFactual Regression Network" (CFRNet) (Fig. 4) (56). The loss function for this network is

$$\min_{h,\Phi,IPM} \frac{1}{n}\sum_{i=1}^{n} \underbrace{MSE(y_i, h(\Phi(x_i), t_i))}_{\text{Outcome Loss}} + \lambda \underbrace{IPM(\Phi(X, |T=1), \Phi(X|T=0))}_{\text{Dist. b/w T \& C covar. distributions}} + \alpha \underbrace{\mathcal{R}(h)}_{L_2} \tag{7}$$

where $R(h)$ is a model complexity term and λ and α are hyperparameters.

These two papers also make important theoretical contributions by providing bounds on the generalization error for the PEHE (27). In Shalit et al. (56), they show that the PEHE is bounded by the sum of the factual loss, counterfactual loss, and the variance of the conditional outcome.

In Johansson et al. (34), the authors introduce estimated IPW weights $\pi(\Phi(X), T)$ to CFR-

[3]This kernel trick also makes support vector machines computationally tractable.

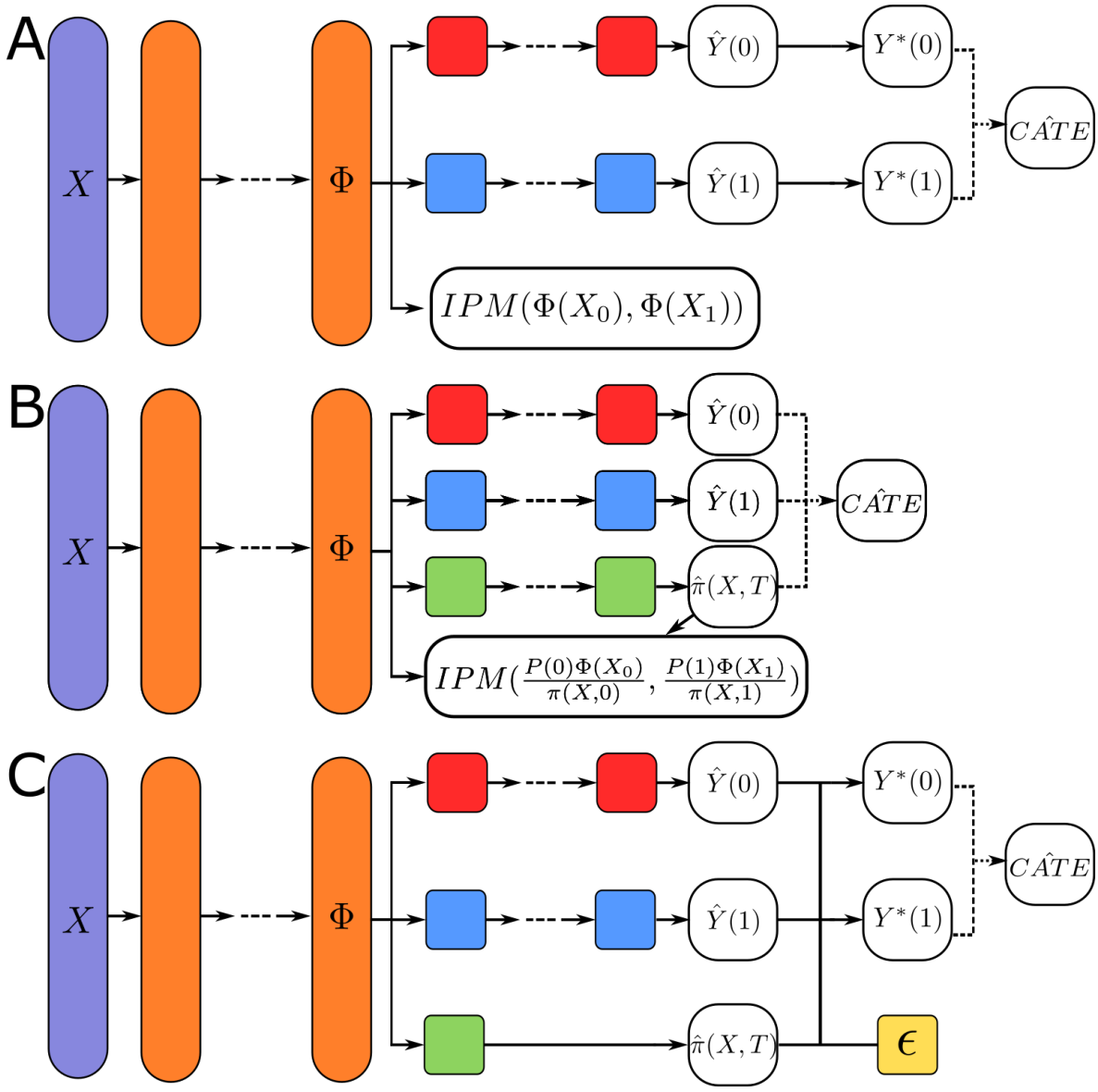


Figure 4: **A. CFRNet** architecture originally introduced in Shalit et al. (56). CFRNet adds an additional integral probability metric loss to explicitly force representations of the treated and control covariates closer in representation space.
B. Weighted CFRNet adds an propensity score head to CFRNet to predict IPW-weighted outcomes. During training, the propensity score is used to reweight both the outcomes $\hat{Y}(0)$ and $\hat{Y}(1)$, as well as the represented covariate distributions in calculation of the IPM loss. This allows the authors to provide consistency guarantees and relax the overlap assumption. Figures adapted from Johansson et al. (34).
C. Dragonnet Dragonnet also adds a propensity score head to TARNet and a free parameter ϵ. In an adaptation of Targeted Maximum Likelihood Estimation, the $\hat{\pi}$ and ϵ are used to re-weight the outcomes to provide lower biased estimates of the ATE.

Net to provide consistency guarantees (Fig. 4B). Theoretically, they also use these weights to relax the overlap assumption as long as the weights themselves obey the positivity assumption. From a practical standpoint, adding weights that are optimized smoothly acoss the whole dataset each epoch reduces noise create by calculating the IPM score in small batches. Weighted CFRNet minimizes the following loss function:

$$\underset{h,\Phi,IPM,\pi,\lambda_h,\lambda_w}{\arg\min} \frac{1}{n}\sum_{i=1}^{n} \underbrace{\frac{P(t_i)}{\pi(\Phi(x_i,t_i))}}_{IPW} \cdot \underbrace{MSE(y_i, h(\Phi(x_i),t_i))}_{\text{Outcome Loss}} + \lambda_h \underbrace{\mathcal{R}(h)}_{L_2 \text{ Outcome}} + \tag{8}$$
$$\underbrace{\alpha \cdot IPM(\underbrace{\frac{P(1)}{\pi(\Phi(X,1))}}_{IPW} \cdot \Phi(X,|T=1), \underbrace{\frac{P(0)}{\pi(\Phi(X,0))}}_{IPW} \cdot \Phi(X|T=0))}_{\text{Distance between } IPW \text{ weighted T \& C covar. distributions}} + \lambda_\pi \underbrace{\frac{||\pi||_2}{n}}_{L_2\mathsf{VAR}(\pi)}$$

where $R(h)$ is a model complexity term and λ_h, λ_π and α are hyperparameters. The final term is a regularization term on the weight parameters.

Extending Representation Balancing with Matching

Beyond IPMs, other approaches have directly embraced matching as a balancing strategy. Yao et al. (68) train their TARNet on six point mini-batches of propensity score-matched units with additional reconstruction losses designed to preserve the relative distances between these points when projecting them into representation space. Schwab et al. (55) takes an even simpler approach by feeding random batches of propensity-matched units to the TarNet outcome structure. While limited in their theoretical motivation, these approaches both show strong empirical performance in simulations.

4.3. Extensions with Treatment Modeling (IPW)

Rather than applying losses directly to the representation function, IPW methods estimate propensity scores from representations using the function $\pi(\Phi(X), T) = P(T|X)$. As in traditional IPW estimators, these methods exploit the sufficiency of correctly-specified propensity scores to reweight the plugged-in outcome predictions and provide unbiased estimates of the ATE (53). Because these models combine outcome modeling with IPW, they retain the

attractive statistical properties of doubly robust estimators discussed in section 3.2.2. Atan et al. (4) combines adversarial learning with IPW estimation, while Shi et al. (58)'s Dragonnet model adapts semi-parametric estimation theory for batch-wise neural network training in a procedure they call "Targeted Regularization" (TarReg) (40). We discuss Dragonnet and Targeted Regularization in more detail below, including a brief introduction to semi-parametric theory and targeted maximum likelihood (TMLE) for context.

Treatment Modeling with Dragonnet

Rather than adding an IPM loss, another trivial extension to TARNet is to add a third head to predict the propensity score. This third head could use multiple neural network layers or just a single neuron, as proposed in Dragonnet (Fig. **??**) (58).

The loss function for this network looks like this:

$$\underset{\Phi,\pi,h}{\arg\min}\ \underbrace{MSE(Y, h(\Phi(X), T))}_{\text{Outcome Loss}} + \alpha \underbrace{\text{BCE}(T, \pi(\Phi(X), T))}_{\pi \text{ Loss}} + \lambda \underbrace{\mathcal{R}(h)}_{L_2} \tag{9}$$

with α being a hyperparameter to balance the two objectives.

Semi-Parametric Theory

The application of semi-parametric theory to causal inference is focused on estimating a target parameter of a distribution P of treatment effects $T(P) := ATE$. While we do not know the true distribution of treatment effects because we lack counterfactuals, we do know some parameters of this distribution (e.g., the treatment assignment mechanism). We can encode these constraints in the form of a likelihood that parametrically defines a set of possible approximate distributions of P from our existing data called $\mathcal{P}$. Within this set there is a sample-inferred distribution $\tilde{P} \in \mathcal{P}$, that can be used to estimate $T(P)$ using $T(\tilde{P})$.

Regardless of $\tilde{P}$ chosen, $\tilde{P} \neq P \rightarrow T(\tilde{P}) \neq T(P)$. We do not know how to pick $\tilde{P}$ with finite data to get the best estimate $T(\tilde{P})$. We can maximize a likelihood function to pick $\tilde{P}$, but there may be "nuisance" parameters in the likelihood that are not the target and we do not care about estimating accurately. Maximum likelihood optimization may provide lower-biased

estimates of these nuissance terms at the cost of better estimates of $T(P)$.

To sharpen the likelihood's focus on $T(P)$, we define a "nudge" parameter ϵ that moves $\tilde{P}$ closer to P (thus moving $T(\tilde{P})$ closer to $T(P)$). An influence curve of $T(P)$ tells us how changes in ϵ will induce changes in $T(P + \epsilon(\tilde{P} - P))$. We'll use this influence curve to fit ϵ to get a better approximation of $T(P)$ within the likelihood framework. In particular, there is a specific **efficient influence curve (EIC)** that provides us with the lowest variance estimates of $T(P)$. In causal estimation, solving the EIC for the ATE yields estimates that are asymptotically unbiased, efficient, and have asymptotic confidence intervals.

The EIC for the ATE is,

$$(10)\quad EIC_{ATE} = \frac{1}{N}\sum_{i=1}^{N} \underbrace{[\underbrace{(\frac{T}{\pi(x,1)} - \frac{1-T}{\pi(x,0)})}_{\text{Treatment Modeling}} \times \underbrace{(Y - h(x,T))}_{\text{Residual Confounding}}]}_{\text{Adjustment}} + \underbrace{[h(x,1) - h(x,0)]}_{\text{Outcome Modeling}} - ATE$$

Minimizing EIC_{ATE} to 0,

$$(11)\qquad ATE = \frac{1}{N}\sum_{i=1}^{N} \underbrace{[\underbrace{(\frac{T}{\pi(x,1)} - \frac{1-T}{\pi(x,0)})}_{\text{Treatment Modeling}} \times \underbrace{(Y - h(x,T))}_{\text{Residual Confounding}}]}_{\text{Adjustment}} + \underbrace{[h(x,1) - h(x,0)]}_{\text{Outcome Modeling}}$$

The underbraces illustrate how EIC_{ATE} resembles a doubly robust estimator. When the EIC is minimized (set to 0) as in equation 11, the ATE is equal to the outcome modeling estimate plus a treatment modeling estimate proportional to the residual error.

From TMLE to Targeted Regularization

Targeted Regularization (TarReg) is closely modeled after "Targeted Maxmimum Likelihood Estimation" (TMLE) (63). TMLE is an iterative procedure where a nuissance parameter ϵ is used to nudge the outcome models towards sharper estimates of the ATE when minimizing the EIC as in 11.

1. Fit h by predicting outcomes (e.g., using TARNet) and minimizing $MSE(Y, h(X,T))$
2. Fit π by predicting treatment (e.g., using logistic regression) and $BCE(T, \pi(X,T))$
3. Plug-in h and π functions to fit ϵ and estimate $h^*(X,T)$ where,

$$\underbrace{h^*(X,T)}_{Y^*} = \underbrace{h(X,T)}_{\hat{Y}} + \underbrace{(\frac{T}{\pi(X,T)} - \frac{1-T}{\pi(X,1-T)})}_{\text{Treatment Modeling Adjustment}} \times \underbrace{\epsilon}_{\text{“nudge”}}$$

   by minimizing $MSE(Y, h^*(X,T))$. This is equivalent to minimizing the “Adjustment” part in equation 11.

4. Plug-in $h^*(X,T)$ to estimate $\hat{ATE}$:

$$\hat{ATE}_{TMLE} = \frac{1}{N}\sum_{i=1}^{N} \underbrace{h^*(x_i,1)}_{y_i^*(1)} - \underbrace{h^*(x_i,0)}_{y_i^*(0)}$$

Targeted Regularization takes TMLE and adapts it for a neural network loss function. The main difference is that steps 1 and 2 above are done concurrently by Dragonnet, and that the loss functions for the first three steps are combined into a single loss applied to the whole network at the end of each batch. It requires adding a single free parameter to the Dragonnet network for ϵ.

At a very intuitive level, Targeted Regularization is appealing because it introduces a loss function to TARNet that explicitly encourages the network to learn the mean of the treatment effect distribution, and not just the outcome distribution. The Targeted Regularization procedure proceeds as follows:

In each epoch:

1. (a) Use Dragonnet to predict $h(\Phi(X),T)$ and $\pi(\Phi(X),T)$.
   (b) Calculate the standard ML loss for the network using a hyperparameter α:

$$\underset{\Phi,\pi,h}{\arg\min}\ \underbrace{MSE(Y, h(\Phi(X),T))}_{\text{Outcome Loss}} + \alpha \underbrace{\text{BCE}(T, \pi(\Phi(X),T))}_{\pi\text{ Loss}} + \lambda \underbrace{\mathcal{R}(h)}_{L_2}$$

2. (a) Compute $h^*(\Phi(X),T)$ as above,

$$\underbrace{h^*(\Phi(X),T)}_{Y^*} = \underbrace{h(\Phi(X),T)}_{\hat{Y}} + \underbrace{(\frac{T}{\pi(\Phi(X),T)} - \frac{1-T}{\pi(\Phi(X),1-T)})}_{\text{Treatment Modeling Adjustment}} \times \underbrace{\epsilon}_{\text{“nudge”}}$$

(b) Calculate the targeted regularization loss: $MSE(Y, h^*(\Phi(X), T))$

3. Combine and minimize the losses from 1 and 2 using a hyperparameter β,

$$\underset{\Phi,h,\epsilon}{\arg\min} = [\underbrace{MSE(Y, h(\Phi(X), T))}_{\text{Outcome Loss}} + \alpha \cdot \underbrace{\text{BCE}(T, \pi(\Phi(X, T))]}_{\pi \text{ Loss}} + \lambda \underbrace{\mathcal{R}(h)}_{L_2} + \beta \cdot \underbrace{MSE(Y, h^*(\Phi(X), T))}_{\text{Targeted Regularization Loss}}$$

Step 3 of Targeted Regularization is exactly equivalent to minimizing the EIC up to a constant β.

At the end of training, we can thus estimate the targeted regularization estimate of the ATE $\hat{ATE}_{TR}$ as in TMLE:

$$\hat{ATE}_{TR} = \frac{1}{N} \sum_{i=1}^{N} \underbrace{h^*(\Phi(x_i), 1)}_{y_i^*(1)} - \underbrace{h^*(\Phi(x_i), 0)}_{y_i^*(0)}$$

Other approaches to estimating IPW weights using adversarial training are discussed in the next section (48; 38). We note that a number of other losses for the basic TarNet/Dragonnet architecture have been proposed with differing theoretical motivations. In the interest of brevity, these approaches are discussed briefly in Box 5.

Box 5: Other Flavors of TarNet

A number of additional losses have been proposed for the representation layers in the two-headed TARNet or three-headed Weighted CFRNet/Dragonnet architectures:

- **Reconstruction Loss.** A number of papers have proposed that reconstruction losses should be applied to representation layers to improve confidence in the invertability assumption (17; 69). These losses simply minimize an L_2 norm between inputs and outputs to force the representation function to be able to reconstruct it's inputs, along with it's other tasks: $\mathcal{L}(X, X') = ||X - X'||^2$
- **Adversarial Loss.** Rather than learn to predict the propensity score Du et al. (17), apply an adversarial gradient to force the representation layers to "unlearn" information about treatment assignment. This approach is also applied in Bica et al. (7).
- **Propensity Dropout.** While not a loss *per se*, Alaa et al. (2) propose probabilistically applying dropout to neurons based on the Shannon Entropy in propensity score predictions.This penalty forces the network to attend comparatively more to data where overlap is greatest and the propensity score is not close to either the 0 or 1 extremes.

Note that because they solve the EIC estimating equation for the ATE, both TMLE and Targeted Regularization are doubly robust estimators.

4.4. Adversarial Training of Generative Models, Representations, IPW

The Origins of Adversarial Training in GANs

Adversarial training approaches include a wide variety of architectures where two networks or loss functions compete against each other. Adversarial approaches are inspired by Generative Adversarial Networks (GANs) (Box 6) (21). In the machine learning literature on causal inference, adversarial training has been applied both to trade off outcome modeling and treatment modeling tasks during representation learning, as well as to trade off estimation and regularization of IPW weights. GANs have also been used directly as generative models for counterfactual and treatment effect distributions.

Box 6: Generative Adversarial Networks

In GANs, two networks, a discriminator network D and a generator network G, play a zero-sum game like cops and robbers. The generator network's job is to learn a distribution from which the training data X could have credibly been generated. In each training batch, the generator produces a new outcome (originally images, but could be IPW weights, counterfactuals or treatment effects) by drawing a random noise sample from a known distribution Z (e.g. Gaussian) and transforming it into outcomes with the function $G(Z) = \hat{X}$. The discriminator's job is to learn a function $D(X) = P(X\text{is real})$ that can distinguish whether the outcome is from the training data X, or whether it is a "fake" $\hat{X}$ created by the generator. The generator then receives a negative version of the discriminator's loss, a penalty that is proportional to how well it was able to "deceive" the discriminator. The discriminator's loss can be the log loss, Jensen-Shannon divergence (21), the Wasserstein distance (3; 23), or any number of divergences and IPMs. Formally, the generator attempts to minimize the following loss function,

$$\arg\min_G = \underbrace{\mathbb{E}_X}_{\text{EV data}}[\underbrace{\mathcal{L}(D(X)}_{P(X\text{is real})}] + \underbrace{\mathbb{E}_Z}_{\text{EV fakes}}[1 - \underbrace{\mathcal{L}(D(G(Z))}_{P(\hat{X}\text{is real})}]$$

where the first quantity is the discriminator's estimated probability data from X is indeed real, and the second quantity is the discriminator's estimate that a generated quantity from the distribution Z is real.

Because the discriminator is trying to catch the generator, its objective is to maxi-

mize the same function,

$$\arg\max_{D} = \underbrace{\mathbb{E}_X}_{\text{EV data}} [\underbrace{\mathcal{L}(D(X)}_{P(X\text{is real})}] + \underbrace{\mathbb{E}_Z}_{\text{EV fakes}} [1 - \underbrace{\mathcal{L}(D(G(Z))}_{P(\hat{X}\text{is real})}]$$

In practice, the discriminator and the generator are trained either alternatingly or simultaneously, with the discriminator increasing its ability to discern between real and fake outcomes over time, and the generator increasing its ability to deceive the discriminator. The idea is that the adaptive loss function created by the discriminator can coax the generator out of local minima to generate superior outcomes. Results by these models have been impressive, and many of the fake portraits and "deepfake" videos circulating online in recent years were generated by this architecture. The advantage of GANS is that they can impressively learn very complex generative distributions with limited modeling assumptions. The disadvantage of GANS is that they are extremely difficult and unreliable to train, often plateauing in local optima.

GANs as Generative Models of Treatment Effect Distributions

Although a generative model of the treatment effect distribution is generally unknown, a natural application of GANs is to try to machine learn this model from data. GANITE uses two GANs: GAN_1 to model the counterfactual distribution and GAN_2 to model the $CATE$ distribution (1) (Fig. 5). The training procedure for GAN_1 is as follows:

1. Taking X,T, and generative noise Z as input, G_1 generates both potential outcomes $< \tilde{Y}(T), \tilde{Y}(1-T)$. A factual loss $MSE(Y(T), \tilde{Y}(T))$ is applied.
2. Create a new vector $v\tilde{e}c = \{Y(T), \tilde{Y}(1-T)\}$ by combining the observed potential outcome and the counterfactual predicted by D_1.
3. Taking X and $v\tilde{e}c$ as inputs the discriminator rates each value in vec for the probability that it is real using the categorical cross entropy. :
   (12)
   $$\mathcal{L}(D) = CCE(\{\underbrace{P(vec_0 = Y(T))}_{\text{Prob idx 0 is real}}, \underbrace{P(v\tilde{e}c_1 = Y(T))}_{\text{Prob sec idx is real}}\}, \{\underbrace{v\tilde{e}c_0 == Y(T)}_{\text{1 if idx 0 is real}}, \underbrace{v\tilde{e}c_1 == Y(T)}_{\text{1 if idx 1 is real}}\})$$
4. This loss is then fed back to the generator G_1 such that the total loss for the is now
   (13)
   $$\arg\min_{G_1} = MSE(Y(T), \tilde{Y}(T)) + \lambda\mathcal{L}(D_1)$$

After generator G_1 is trained to completion, the authors use $v\tilde{e}c$ as a "complete dataset" containing both a factual outcome and a counterfactual outcome to train GAN_2 which generates treatment effects. GAN_2 is no different than any other conditional GAN.

1. Taking X and generative noise Z as input, G_2 generates a possible potential outcome vector $\hat{\vec{\cdot}} = \{\hat{Y}(T_{G2}), \hat{Y}(1 - T_{G2})\}$. G_2 receives an MSE loss to minimize the difference between it's predictions and the "complete dataset" $\tilde{vec}$: $MSE(\tilde{vec}, \hat{vec})$.

2. Discriminator D_2 takes X, $\tilde{\vec{\cdot}}$, and $\hat{\vec{\cdot}}$ as inputs and estimates a probability that each of them is the "complete" dataset $\tilde{\vec{\cdot}}$.
(14)
$$\mathcal{L}(D) = CCE(\{\underbrace{P(\tilde{vec} = \tilde{vec})}_{\text{Prob idx 0 is } \tilde{vec} \text{ (“CD”)}}, \underbrace{P(\hat{vec} = \tilde{vec})}_{\text{Prob } \hat{vec} \text{ is } \tilde{vec}}\}, \{\underbrace{\tilde{vec} == \tilde{vec}}_{\text{1 if idx 0 is ‘}\tilde{vec}}, \underbrace{\tilde{vec}_1 == Y(T)}_{\text{1 if idx 1 is } \tilde{vec}}\})$$

3. This loss is then fed back to the generator G_2 such that the total loss for the is now
(15)
$$\arg\min_{G_2} = MSE(\tilde{vec}, \hat{vec}) + \lambda\mathcal{L}(D_2)$$

GANTaking noise Z as input, the generator of the first GAN G_1 encodes a function that learns to predict both potential outcomes (as in TARNet) taking X, $Y(T)$, and T as inputs,

$$(\hat{Y(1)}, \hat{Y(0)}) = G_1(X, T, Y(T), Z)$$

This generator is trained on a joint loss function consisting of a supervised MSE loss for predicting the factual outcome correctly, and the negative gradient of it's discriminator D_1.

Although both potential outcomes are generated, D_1 attempts to discern which one of the two potential outcomes came from the factual distribution, driving G_1 to create more credible counterfactual outcomes. After training, the authors use G_1 to create a "complete" dataset $\mathcal{D}$ including both factual and counterfactual potential outcomes.

The second GAN, G_2, is trained to generate potential outcomes conditional only on X (not T) for out-of-sample prediction of $CATE$s when treatment assignment is unknown. D_2 then attempts to distinguish between these generated outcomes and outcomes from the complete dataset $\mathcal{D}$. The discrimination probabilities of D_2 give an estimate of confidence in $CATE$ predictions.

SCIGAN extends GANITE to settings with more than one treatment and continuous dosages (8). In this setting treatment is parameterized as a binary factor vector denoting which treatment $w \in W$ of possible treatments is used, and a substituent vector of randomly sampled dosages $\mathbb{D}_w$ for treatment w, such that $t = (w, \mathbb{D}_w)$. The generator G is identical to G_1 in

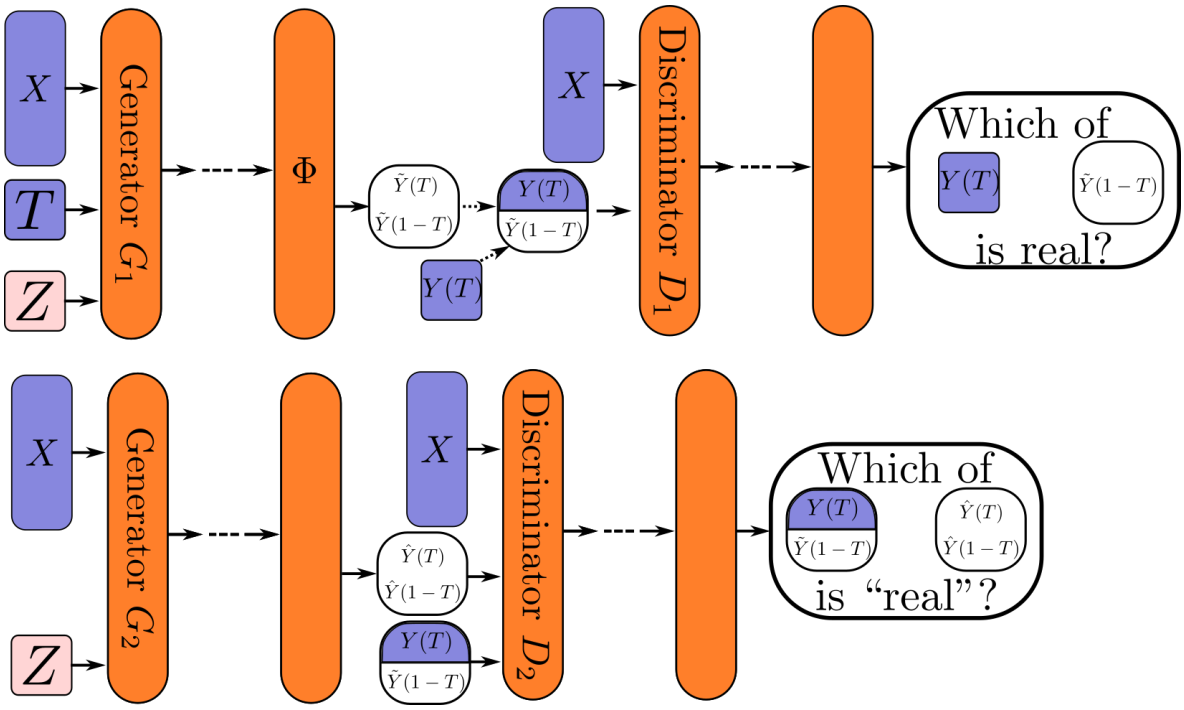


Figure 5: **GANITE**

GANITE except that G produces a separate potential outcome for each treatment-dosage combination $\hat{Y}(W, \mathbb{D}_W) = \{\hat{y}(w_1, \mathbb{D}_{w1}), \hat{y}(w_2, \mathbb{D}_{w2}), \ldots, \hat{y}(w_n, \mathbb{D}_{wn})\}$. G has two discriminators, D_w and $D_{\mathbb{D}}$. D_w selects which of the generated treatments $\hat{W}$ is the real one. $D_{\mathbb{D}}$ subsequently guesses which is the correct dosage for the treatment $\hat{w}$ selected by D_w. For out-of-sample prediction, SCIGAN replaces G_2 from GANITE with an MLP for out-of-sample prediction using the complete dataset produced by G.

Adversarial Representation Balancing

The use of the IPM loss in CFRNet (56) may be viewed as an adversarial approach in that the representation layers are forced to maximize performance on two competing tasks: predicting outcomes and minimizing an IPM. Rather than using an IPM loss, other authors have trained propensity score estimators that send positive (rather than negative) gradients back to the representation layers (4; 17). This strategy explicitly decorrelates the covariate representations from the treatment. Du et al. (17) adds a third head to CFRNet that learns the propensity score (similar to Dragonnet) but flips it's gradient when passing it back to $\Phi(X)$.

They also place an addition reconstruction loss on the representation layers to encourage invertibility (see Box 5).

Bica et al. (7) extend this approach to settings with treatment over time using a recurrent neural network. In their medical setting, decorrelating treatment from patient covariates and history allows them to estimate treatment effects at each individual snapshot. This algorithm is described in Section 5. In Atan et al. (4) representations are first trained using an autoencoder that includes both a reconstruction loss and this adversarial gradient. The representation produced by this autoencoder is then fed to TARNet.

Adversarial IPW Learning

In the most minimal adversarial IPW model, Ozery-flato et al. (48) estimate IPW weights for the ATE adversarially. A discriminator is presented with two sets of weights: one uniform and the other estimated, and tasked with distinguishing between the two distributions. The "generator" updates the estimated weights to minimize the ability of the discriminator to distinguish between them. The generator uses exponential gradient ascent during training to regularize the weights and minimize their variance.

The approach in Kallus (38) is similar. They first develop theory relating the generic "discriminative distance" of a discriminator's loss to formal IPMs. They then proposes GAN set up where a discriminator network attempts to minimize the discriminative distance, while the "generator" of the weights is itself a deep neural network.

Lastly, Johansson et al. (33) and Johansson et al. (34) rework the approach in Kallus (38) to combine adversarial training, IPW weighting, and IPM balancing as discussed above in 4.2.

5. Extending Causal Estimation to Non-tabular Data

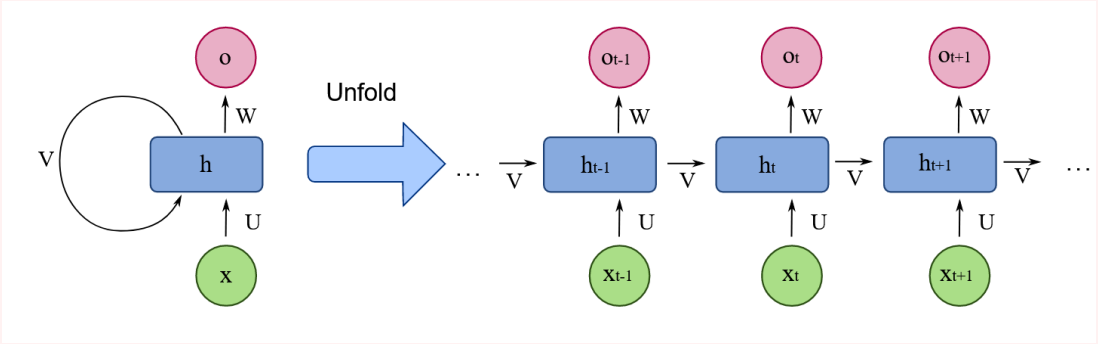
Below we describe extensions of causal inference that are not possible with other types of machine learning. First we address estimation of treatment effects with time varying treatments and confounding. We also see the great potential for neural networks in causal inference when latent confounding is encoded in non-tabular data as well (e.g., text, networks, and images).

5.1. Conditioning on Time-Varying Confounding

One natural extension to deep estimation on selection on observables is measuring treatment effects at discrete time points when treatment is administered over time.[4] A few papers have adapted the above representation learning approaches to temporal settings using recurrent neural networks (Box 7).

Box 7: Recurrent Neural Networks (RNN)

Recurrent neural networks are a specialized architecture created for learning outcomes from sequential data (e.g. time series, biological sequences, text). In a classic RNN, each "unit" in the network takes as input it's own input (or representation) and a representation produced by the previous unit, encoding cumulative information about earlier states in the sequence. These units are not hidden layers: there is a set of weights within each unit for it's raw inputs, the representation from the previous time step, and it's outputs. Different RNN variants have different operations for integrating past representations and raw inputs. Recurrent neural networks may be directed acyclic graphs or feedback on themselves. Commonly used variants include Gated Recurrent Unit networks (GRU) and Long-term Short-term memory networks (LSTM) (CITE).



figureRecurrent neural network. x denotes inputs, h denotes units and o denotes outputs.

A simple extension RNN-based extension to CFRNet proposed by Cheng et al. (10) is to replace each outcome modeling head with an RNN, one modeling the untreated group,

$$\{\hat{Y}_{t=0}(0) \dots \hat{Y}_t = t\} = \{h(X_t, 0, h(X_{t-1}, 0)) \dots h(X_T, 1, h(X_{T-1}, 0))\}.$$

[4] For a graphical description of this scenario and discussion of the challenges it entails, see CITE on marginal structural models

and one modeling the treated group,

$$\{\hat{Y}_{t=0}(1) \dots \hat{Y}_t = t\} = \{h(X_t, 1, h(X_{t-1}, 1)) \dots h(X_T, 1, h(X_{T-1}, 1))\}.$$

At each time step, they add an additional IPM loss across the RNN's to minimize the distance between $h(X_t, 0)$ and $h(X_t, 1)$.

To more explicitly address the residual confounding from previous treatment, Bica et al. (7) combine a similar RNN approach with adversarial training. This algorithm is described in more detail in 4.4.

5.2. Relaxing Strong Ignorability: Controlling for Latent Confounders

TOPICS NOT ADDRESSED:

Modeling Latent Confounder using Variational Autoencoders

[Note BK: I am ambivalent about this because I think approaches to capturing latent confounding are extremely controversial. Note I do not believe this literature and am wondering whether we should skip it. See https://arxiv.org/abs/2102.06648] Discuss CE-VAE (43), (31), Vowels? https://arxiv.org/abs/2102.06648 http://proceedings.mlr.press/v119/witty20a.html

5.3. CAUSAL REGULARIZATION

5.4. UNCERTAINTY

https://arxiv.org/abs/2002.10837

Latent Confounders Encoded in Networks

The causal effect of treating nodes on a network's structure is difficult to measure because the lack of treatment contagion/homophily is a fundamental assumption in all causal frameworks (SUTVA) (57). However, there is considerable interest in exploring whether the structural position of units in a network encodes information about unobserved/unmeasured confounders,

in scenarios where SUTVA holds (i.e., relax the strong ignorability assumption). For example, even if age or gender is unmeasured, these features might be inferrable from a person's homophilic friendship ties. To use the simulation proposed in Guo et al. (25) as another example, in a social network of bloggers, network structure may encode information about their subject matter (25). Each blogger may be interested in the effect viewing platform (e.g. desktop or mobile) (treatment) has on their viewership (outcome), but this is generically confounded by subject matter.

Nevertheless, causal inference is tricky in this scenario because we generally do not have compelling generative models for networks that describe how features/covariates might lead to network topologies. Instead of generative network models, early literature in this area is largely leveraging representations of nodes that are created by tasking a neural network to predict the probability of two nodes being linked. The architectures that generate these node embeddings can leverage either strictly local information, as in early random-walk based algorithms such as node2vec or DeepWalk (CITE), or information from the entire network, including node covariates/features, as in graph neural networks (see Box 8).

Box 8: Graph Neural Networks (GNN)

Graph neural networks (GNNs) are the current state-of-the-art approach for creating representations for nodes in graphs. Compared to previous approaches that relied on random walks from node to node, GNNs are attractive because their node representations are aggregated from the structural position and covariates of all nodes n degrees away from the target node, where n is the number of graph neural network layers.

The most intuitive understanding of how graph neural networks work is as a message passing system (CITE). We use the original GNN paper, the Graph Convolutional Network as an example. In this interpretation, each node has a message that it passes to it's neighbors through a graph convolution operation. In the first layer of a GNN this message would consist of the node's covariates/features. In consecutive layers of the network, these messages are actually representations of the node from the previous layer. During graph convolution, each node multiplies incoming messages from it's first degree neighbors, by it's weights and integrates them using an aggregation function (e.g., summation). By the n-th GNN layer, these messages will contain structure and covariate information from all nodes n degrees away. For interested readers, there is also a spectral interpretation of this process. The standard (unsupervised) loss functions for GNNs is the log loss of the probability that two nodes are linked in the network.

One variant of the GNN uses an "attention" mechanism to vary the extent that

nodes value messages from different neighbors (the graph attention network or GAT) (66). This network is closely related to the transformer architecture discussed below.

A first pass approach to this problem is proposed in Guo et al. (25). They train two independent networks for $h(\phi_h(X)$ (outcome prediction) and $\pi(\phi_t(X)$ (propensity prediction) where $\phi_h(X)$ and $\phi_t(X)$ take the network structure and node covariates as input and feed them to a GAT network. They then combine these independent outcome and propensity estimates using a doubly robust estimator. A shared loss between the networks force Φ_h and Φ_π to produce similar representations of the nodes/covariates.

[Note BK: Remove this. It's not interesting] Chu et al. (12) and Guo et al. (24) address the same scenario, but take on a significant technical problem: GNN's cannot actually learn distinct representations of treated and control units and balance their representations at the same time. GNNs learn from examples of both positive and negative links. If the network does encode unobserved confounders, the positive and negative example distribution are likely to be substantially different. Chu et al. (12) thus add an additional addaptive loss that forces the gradients supplied by negative examples to better reflect the overall structure of the graph. Guo et al. (24) instead pose learning high quality representations of treated and control nodes and balancing their distributions to address confounding as adversarial losses.

[Note BK: Also not sure how important this is given ridiculous assumption] Veitch et al. (65) take a formal semiparametric approach to identify a causal effect. They assume that a node representation need only encode necessary structural information about unobserved confounders X that allows one to make consistent estimates of the treatment and outcome, even if it does not encode all information about X completely. Under these conditions, along with an assumption that representations of the nodes are only weakly correlated (a strong assumption), they show that a variant of the AIPW estimator is asymptotically normal, unbiased, and $\frac{1}{\sqrt{n}}$ consistent ATE estimator. The proof is an adaptation of Chernozhukov et al. (11) that allows not having a complete validation set using the weak correlation assumption. The authors conduct a naturalistic simulation using 79,000 individuals in an online social network in Slovakia called Pokec.

5.5. Conditioning on Latent Confounding in Text Data

Compared to research on latent confounding encoded in networks, causal inference from text data is much larger area of development within both the computer science and social science communities. There are methods that treat text as treatments, mediators, proxies for latent confounding, or outcomes (see Keith et al. (39) for an exhaustive review). In this article we narrowly focus on articles that use deep representations, but further note that other methods relying on other quantitative representations (eg., topics, word counts) may address similar problems.

Box 9: Transformers

As of 2021, transformers are the hegemonic architecture used in natural language processing. After their introduction in 2017, these models improved performance on many high-profile NLP tasks across the board. Several enterprise-scale transformers have been featured in the media for their impressive performance in text generation and question answering (e.g. OpenAI's GPT-3). Smaller models in broad use are based on the BERT architecture.

The connection between GNNs, and specifically GATs, is the focus on attention mechanisms (Box 8). From this perspective, words in sentences are akin to nodes in networks, with their relative positions to each other being analogous to their structural positions in the graph. Transformers improved on previous sequential approaches to text analysis (i.e. RNNs) by having each word (or representation of a word) receive messages from not just adjacent words, but all sentences heterogeneously. Attention mechanisms throughout the architecture allow each layer of a transformer to attend to words or aggregated representation mechanisms heterogeneously. Architectures such as BERT that stack transformer layers to create models with hundreds of millions of parameters. These models are expensive to train, both computationally and with respect to data, so they are often pretrained on large datasets and then "fine-tuned" (lightly re-trained) with smaller datasets for specific tasks.

Veitch et al. (64) proposes some conditions under which a causal effect would be identifiable using text representations under a semi-parametric framework. The motivation for the paper is that causal inference directly from text (or even topic models) poses a curse of dimensionality problem, and thus it would be convenient to use pre-trained representations from transformer algorithms such as BERT (16) (see Box 9).

The main contribution of the paper is a theorem that states that if adjusting for confounding encoded in the text W is sufficient to identify a causal effect, then adjusting for confounding

Z encoded in a representation $\Phi(W)$ is also sufficient for causal identification. For this to be true, at least one of the following three must be measurable:

1. $h(Z, 1), h(Z, 1)$

2. $\pi(Z)$,

3. $h(1, \pi(Z)), h(0, \pi(Z))$,

If confounding by text is measurable by outcome modeling, propensity score modeling, or a doubly robust estimator in a representation of that text, the causal effect is identifiable. As an estimator, they fine-tune a BERT model to predict both h and π

In other words an architecture that can simultaneously learn a good text representation, predict the outcome from that representation, and predict the treatment. Assuming the conditions above, that the $h(Z)$ and $\pi(Z)$ are consistent estimators, and that the generalization error for some plugin estimator (e.g. IPW weighting of the CATE) for τ using h and g is bounded, then this plugin estimator will be asymptotically unbiased. They demonstrate this estimator on a Reddit scenario and another scenario to test the causal effect of equations on getting papers accepted to CS conferences.

Pryzant et al. (50) explores the scenario where both treatment (some linguistic property) and confounders are encoded in text. They note that an undeveloped idea in text identification is that the reader, not the writer must be able to identify the treatment and produce the outcome. Since the reader's perception of the treatment is also unobservable, we must use a proxy label (e.g. number of stars in a product review) to measure the treatment. They show both that the ATE is identifiable in this scenario, and that any bias induced by differences between the proxy label and the actual perception of treatment may attenuate the ATE but not change it's sign. In their proposed architecture a transformer representation is fed to TARNet along with the non-text covariates X. The TARNet loss is simultaneously used to train itself and as a fine-tuning objective for a Transformers.

5.6. Causal Inference on Images

[TBD]

6. Conclusion

In this review we summarize the emerging machine learning literature on deep learning estimators for potential outcomes. We give brief introductions to neural networks and causal inference, before describing traditional strategies for causal estimation under selection on observable identification assumptions. The main body of the review focuses on four estimation techniques common in this literature: deep outcome modeling, balancing through representation learning, IPW weighting, and adversarial training for representation balancing, IPW, or generating counterfactual distributions. Lastly, we highlight future directions for this literature towards greater interpretability and the usage of text, network, and image representations. The paper also provides a thorough introduction to some of the most common models in deep learning (e.g. autoencoders, recurrent neural networks, generative adversarial networks, graph neural networks, and transformers).

The marriage of machine learning and causal inference is an exciting trend across social science, computer science, and industry. Deep learning brings much to the table for causal inference. Because deep learning estimators can approximate any continuous function, they are the lowest bias heterogeneous treatment effect estimators in our arsenal. Deep learning estimators also provide opportunities for social science to extrapolate heterogeneous treatment effects estimated within sample to untreated populations. Moreover, these interpretable models allow social scientists to enjoy the benefits of deep learning for exploring text, network, and image data that have been developed across other sciences and industry.

However, this is an emerging literature and challenges remain for reliable causal estimation. While many early papers in this literature lacked theoretical guarantees, these are now standard for papers published in high-status venues. Nevertheless, many of these guarantees are framed as generalization bounds for prediction, rather than asymptotic properties like bias and consistency valued by social scientists and statisticians. However consistent, doubly

robust estimators do exist in this literature (e.g, (34; 58), and are available for use today.

Machine learning approaches present other challenges. For example, IPMs metrics are not consistent as data grows, and while they increase overlap between treated and control distributions, the distributional shape of this overlap is not guaranteed. Similarly, adversarial training may decorrelate treatment and outcome, but in highly parametric functions the form of this decorrelation is unknown. There has also been criticism that GANs are not able to truly model the generative distributions of the training data. Identification assumptions from text, networks, and images are all still being worked out. Nevertheless, these are all active and exciting areas of research.

We see several directions for this literature. As the machine learning and data mining communities becomes more familiar with causal inference, we expect to see more algorithms that are motivated by principled theoretical choices rather than improved performance in simulated datasets. In particular, the movement towards semi-parametric theory seems like a promising direction (58; 65; 65). Second, the emergence of non-parametric machine learning estimators has created a need for complementary tools that can identify meaningful heterogeneity in CATEs. Finally, we are excited by early work on causal inference in text and network data. We look forward to new models for causal inference in images in the near future. The included tutorials provide examples for social scientists to implement two consistent estimators from this literature, Dragonnet and Weighted CFRNet, so that they can begin using these models today (58; 34). The tutorials assume no previous experience with Tensorflow, and address practical considerations in training and regularizing deep causal estimators.

[Note BK: Table will be revised before Arxiving but it would be good to get feedback on how to do so.]

References

[1] (2018). GANITE :Estimation of Indvidualized Treatment using Generative Adversarial Networks. In *International Conference on Learning Representations*.

Table 1: Summary of Potential Outcome Approaches. Columns are selection on observable estimation strategies, rows are machine learning estimation techniques. Note that papers may appear multiple times.

	Balancing	IPW	Other
Integral Probability Metric	Johansson et al. (35); Shalit et al. (56) Johansson et al. (33); Du et al. (17)	Johansson et al. (33)	
Propensity-Matching	Yao et al. (68); Schwab et al. (55)		
Adversarial Training	Johansson et al. (33); Du et al. (17)	Kallus (38); Ozery-flato et al. (48) Johansson et al. (33)	Yoo (1)
Semiparametric Metaloss		Shi et al. (58)	
Selection Bias Dropout	Alaa et al. (2)		
Reconstruction Loss	Yao et al. (68); Du et al. (17)	Atan et al. (4)	

[2] Alaa, A. M., Weisz, M., and Schaar, M. V. D. (2017). Deep Counterfactual Networks with Propensity-Dropout. In *International Conference on Machine Learning*.

[3] Arjovsky, M., Chintala, S., and Bottou, L. (2017). Wasserstein gan. *arXiv preprint arXiv:1701.07875*.

[4] Atan, O., Jordon, J., and Schaar, M. V. D. (2018). Deep-Treat : Learning Optimal Personalized Treatments from Observational Data Using Neural Networks. In *Association for the Advancement of Artificial Intelligence*.

[5] Austin, P. C. (2011). An introduction to propensity score methods for reducing the effects of confounding in observational studies. *Multivariate behavioral research*, 46(3):399–424.

[6] Bengio, Y. (2013). Deep learning of representations: Looking forward. In *International Conference on Statistical Language and Speech Processing*, pages 1–37. Springer.

[7] Bica, I., Alaa, A. M., Jordon, J., and van der Schaar, M. (2020a). Estimating counterfactual treatment outcomes over time through adversarially balanced representations. *arXiv preprint arXiv:2002.04083*.

[8] Bica, I., Jordon, J., and van der Schaar, M. (2020b). Estimating the effects of continuous-valued interventions using generative adversarial networks. In Larochelle, H., Ranzato, M., Hadsell, R., Balcan, M. F., and Lin, H., editors, *Advances in Neural Information Processing Systems*, volume 33, pages 16434–16445. Curran Associates, Inc.

[Brand et al.] Brand, J. E., Koch, B., and Xu, J. Sage research methods.

[10] Cheng, L., Guo, R., and Liu, H. (2021). Long-term effect estimation with surrogate representation. In *Proceedings of the 14th ACM International Conference on Web Search and Data Mining*, WSDM '21, page 274–282, New York, NY, USA. Association for Computing Machinery.

[11] Chernozhukov, V., Chetverikov, D., Demirer, M., Duflo, E., Hansen, C., and Newey, W. K. (2016). Double machine learning for treatment and causal parameters.

[12] Chu, Z., Rathbun, S. L., and Li, S. (2021). Graph infomax adversarial learning for treatment effect estimation with networked observational data. *arXiv preprint arXiv:2106.02881*.

[13] Cuturi, M. (2013). Sinkhorn distances: Lightspeed computation of optimal transport. In *Advances in neural information processing systems*, pages 2292–2300.

[14] Cybenko, G. (1989). Approximation by superpositions of a sigmoidal function. *Mathematics of Control, Signals and Systems*, 5:455.

[15] Daza, D. (2019). Approximating wasserstein distances with pytorch. https://dfdazac.github.io/sinkhorn.html. Last accessed 2019-08-01.

[16] Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. (2018). Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

[17] Du, X., Duivesteijn, W., Sun, L., Nikolaev, A., and Pechizkiy, M. (2019). Adversarial Balancing-based Representation Learning for Causal Effect Inference with Observational Data.

[18] Glynn, A. N. and Quinn, K. M. (2010). An introduction to the augmented inverse propensity weighted estimator. *Political analysis*, 18(1):36–56.

[19] Goldszmidt, M. and Pearl, J. (1996). Qualitative probabilities for default reasoning, belief revision, and causal modeling. *Artificial Intelligence*, 84(1):57–112.

[20] Goodfellow, I., Bengio, Y., and Courville, A. (2016). *Deep Learning*. MIT Press. http://www.deeplearningbook.org.

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

[22] Gretton, A., Borgwardt, K. M., Rasch, M. J., Schölkopf, B., and Smola, A. (2012). A kernel two-sample test. *Journal of Machine Learning Research*, 13(Mar):723–773.

[23] Gulrajani, I., Ahmed, F., Arjovsky, M., Dumoulin, V., and Courville, A. (2017). Improved training of wasserstein gans.

[24] Guo, R., Li, J., Li, Y., Candan, K. S., Raglin, A., and Liu, H. (2020). Ignite: A minimax game toward learning individual treatment effects from networked observational data. In Bessiere, C., editor, *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI-20*, pages 4534–4540. International Joint Conferences on Artificial Intelligence Organization. Special Track on AI in FinTech.

[25] Guo, R., Li, J., and Liu, H. (2019). Counterfactual evaluation of treatment assignment functions with networked observational data. *arXiv preprint arXiv:1912.10536*.

[26] Hernán, M. and Robins, J. (2020). *Causal Inference: What If. 2020*. Chapman & Hall.

[27] Hill, J. L. (2011). Bayesian Nonparametric Modeling for Causal Inference. *Journal of Computational and Graphical Statistics*, 20(1):217–240.

[28] Huszar, F. (2015). Another favourite machine learning paper: Adversarial networks vs kernel scoring rules. https://www.inference.vc/another-favourite-machine-learning-paper-adversarial-networks-vs-kernel-scoring-rules/. Last accessed 2019-08-01.

[29] Imbens, G. W. and Rubin, D. B. (2015). *Causal inference in statistics, social, and biomedical sciences*. Cambridge University Press.

[30] Ioffe, S. and Szegedy, C. (2015). Batch normalization: Accelerating deep network training by reducing internal covariate shift. *ArXiv*, abs/1502.03167.

[31] Jesson, A., Mindermann, S., Shalit, U., and Gal, Y. (2020). Identifying causal effect inference failure with uncertainty-aware models. *CoRR*, abs/2007.00163.

[32] Johansson, F. and Shen, M. (2018). Causal inference deep learning. MIT IAP.

[33] Johansson, F. D., Kallus, N., Shalit, U., and Sontag, D. (2018). Learning Weighted Representations for Generalization Across Designs.

[34] Johansson, F. D., Shalit, U., Kallus, N., and Sontag, D. A. (2020). Generalization bounds and representation learning for estimation of potential outcomes and causal effects. *CoRR*, abs/2001.07426.

[35] Johansson, F. D., Shalit, U., and Sontag, D. (2016). Learning Representations for Counterfactual Inference. In *International Conference on Machine Learning*, volume 48.

[36] Joo, J., Steen, F. F., and Zhu, S.-C. (2015). Automated facial trait judgment and election outcome prediction: Social dimensions of face. In *Proceedings of the IEEE international conference on computer vision*, pages 3712–3720.

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

[38] Kallus, N. (2018). DeepMatch : Balancing Deep Covariate Representations for Causal Inference Using Adversarial Training.

[39] Keith, K., Jensen, D., and O'Connor, B. (2020). Text and causal inference: A review of using text to remove confounding from causal estimates. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5332–5344, Online. Association for Computational Linguistics.

[40] Kennedy, E. H. (2016). Semiparametric theory and empirical processes in causal inference. In *Statistical causal inferences and their applications in public health research*, pages 141–167. Springer.

[41] Kingma, D. P. and Ba, J. (2014). Adam: A method for stochastic optimization. *CoRR*, abs/1412.6980.

[42] LeCun, Y., Bengio, Y., and Hinton, G. (2015). Deep learning. *nature*, 521(7553):436.

[43] Louizos, C., Shalit, U., Mooij, J. M., Sontag, D., Zemel, R., and Welling, M. (2017). Causal effect inference with deep latent-variable models. In *Advances in Neural Information Processing Systems*, pages 6446–6456.

[44] Mansournia, M. A., Hernán, M. A., and Greenland, S. (2013). Matched designs and causal diagrams. *International journal of epidemiology*, 42(3):860–869.

[45] Mikolov, T., Sutskever, I., Chen, K., Corrado, G., and Dean, J. (2013). Distributed representations of words and phrases and their compositionality. In *Proceedings of the 26th International Conference on Neural Information Processing Systems - Volume 2*, NIPS'13, page 3111–3119, Red Hook, NY, USA. Curran Associates Inc.

[46] Müller, A. (1997). Integral probability metrics and their generating classes of functions. *Advances in Applied Probability*, 29(2):429–443.

[47] Nagpal, C., Wei, D., Vinzamuri, B., Shekhar, M., Berger, S. E., Das, S., and Varshney, K. R. (2020). Interpretable subgroup discovery in treatment effect estimation with application to opioid prescribing guidelines. CHIL '20, page 19–29, New York, NY, USA. Association for Computing Machinery.

[48] Ozery-flato, M., Thodoroff, P., and El-Hay, T. (2018). Adversarial balancing for causal inference.

[49] Pearl, J. (2009). *Causality*. Cambridge university press.

[50] Pryzant, R., Card, D., Jurafsky, D., Veitch, V., and Sridhar, D. (2020). Causal effects of linguistic properties. *arXiv preprint arXiv:2010.12919*.

[51] Robins, J. (1986). A new approach to causal inference in mortality studies with a sustained exposure period—application to control of the healthy worker survivor effect. *Mathematical modelling*, 7(9-12):1393–1512.

[52] Robins, J. (1987). A graphical approach to the identification and estimation of causal parameters in mortality studies with sustained exposure periods. *Journal of chronic diseases*, 40:139S–161S.

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

[54] Rubin, D. B. (1974). Estimating causal effects of treatments in randomized and nonrandomized studies. *Journal of educational Psychology*, 66(5):688.

[55] Schwab, P., Linhardt, L., and Karlen, W. (2018). Perfect match : A simple method for learning representations for counterfactual inference with neural networks.

[56] Shalit, U., Johansson, F. D., and Sontag, D. (2017). Estimating individual treatment effect : generalization bounds and algorithms. In *International Conference on Machine Learning*.

[57] Shalizi, C. R. and Thomas, A. C. (2011). Homophily and contagion are generically confounded in observational social network studies. *Sociological Methods & Research*, 40(2):211–239. PMID: 22523436.

[58] Shi, C., Blei, D. M., and Veitch, V. (2019). Adapting Neural Netwroks for the Estimation of Treatment Effects.

[59] Srivastava, N., Hinton, G. E., Krizhevsky, A., Sutskever, I., and Salakhutdinov, R. (2014). Dropout: a simple way to prevent neural networks from overfitting. *J. Mach. Learn. Res.*, 15:1929–1958.

[60] Stock, M. (2017). Notes on optimal transport. https://michielstock.github.io/OptimalTransport/. Last accessed 2019-08-01.

[61] Stuart, E. A. (2010). Matching methods for causal inference: A review and a look forward. *Statistical science: a review journal of the Institute of Mathematical Statistics*, 25(1):1.

[62] Todorov, A., Mandisodza, A. N., Goren, A., and Hall, C. C. (2005). Inferences of competence from faces predict election outcomes. *Science*, 308(5728):1623–1626.

[63] van der Laan, M. and Rose, S. (2011). *Targeted Learning: Causal Inference for Observational and Experimental Data*. Springer Series in Statistics. Springer New York.

[64] Veitch, V., Sridhar, D., and Blei, D. M. (2019a). Using text embeddings for causal inference.

[65] Veitch, V., Wang, Y., and Blei, D. M. (2019b). Using embeddings to correct for unobserved confounding in networks.

[66] Veličković, P., Cucurull, G., Casanova, A., Romero, A., Lio, P., and Bengio, Y. (2017). Graph attention networks. *arXiv preprint arXiv:1710.10903*.

[67] Wager, S. and Athey, S. (2018). Estimation and Inference of Heterogeneous Treatment Effects using Random Forests. *Journal of the American Statistical Association*, 113(523):1228–1242.

[68] Yao, L., Li, S., Li, Y., Huai, M., Gao, J., and Zhang, A. (2018). Representation Learning for Treatment Effect Estimation from Observational Data. In *Advances in neural information processing systems*.

[69] Zhang, Y., Bellot, A., and Schaar, M. (2020). Learning overlapping representations for the estimation of individualized treatment effects. In *International Conference on Artificial Intelligence and Statistics*, pages 1005–1014. PMLR.

A. Integral Probability Metrics

A.1. Wasserstein Distance

Following (60; 15), suppose we have two discrete distributions (treated and control) with marginal densities $p(x)$ and $q(x)$ captured as vectors t and c, with dimensions n and m respectively. To compute the Wasserstein distance, we must define a "mapping matrix" P

that defines the mapping of "earth" in $p(x)$ to corresponding piles in $q(x)$. Let $\mathbf{U}(t, c)$ be the set of positive, $n \times m$ mapping matrices where the sum of the rows is t and the sum of the columns is c.

$$\mathbf{U}(t, c) = P \subset \mathbb{R}_{>0}^{nxm} | P \cdot 1_m = \mathbf{t}, P^T \cdot 1_n = \mathbf{c} \tag{16}$$

In words, this matrix maps the probability mass from points in the support of $p(x)$ (i.e, the elements of t) to points in the support of $q(x)$ (the elements of c) (note that the mapping need not be one-to-one). We also have a "cost" matrix C that describes the cost of applying P (i.e. the cost of shoveling dirt according to the map described in P). The cost matrix can be computed using a norm ℓ (most commonly ℓ^2) between the points in t being mapped to c in the mapping matrix P. Finally, the ℓ-norm Wasserstein distance dW_ℓ can be defined as

$$dW_\ell = min_{P \subset (t,c)} \sum_{i,j} P_{ij} C_{ij} \tag{17}$$

In other words, the Wasserstein distance is the smallest Frobenius inner product of a mapping matrix P that fits the above constraints, and its associated cost matrix C. Although this problem can be solved via linear programming, the Wasserstein distance is often implemented in a different form that works with continuous distributions and can be optimized by gradient descent (3; 23). There is also a variant of the Wasserstein distance that imposes an entropy-based regularization on the coupling matrix to make it smoother or sparser called the Sinkhorn distance (13).

B. Semi-Parametric Theory

The application of semi-parametric theory to causal inference is focused on estimating a target parameter of a distribution P of treatment effects $T(P) := ATE$. While we do not know the true distribution of treatment effects because we lack counterfactuals, we do know some parameters of this distribution (e.g., the treatment assignment mechanism). We can encode these constraints in the form of a likelihood that parametrically defines a set of possible

approximate distributions of P from our existing data called $\mathcal{P}$. Within this set there is a sample-inferred distribution $\tilde{P} \in \mathcal{P}$, that can be used to estimate $T(P)$ using $T(\tilde{P})$.

Regardless of $\tilde{P}$ chosen, $\tilde{P} \neq PT(\tilde{P}) \neq T(P)$. We do not know how to pick $\tilde{P}$ with finite data to get the best estimate $T(\tilde{P})$. We can maximize the likelihood function to pick $\tilde{P}$, but there are a lot of "nuisance" parameters in the likelihood that are not the target and we do not care about estimating accurately, so this will not necessarily give us the best estimate of $T(P)$.

To sharpen the likelihood's focus on $T(P)$, we define a "nudge" parameter ϵ that moves $\tilde{P}$ closer to P (thus moving $T(\tilde{P})$ closer to $T(P)$). An influence curve of $T(P)$ tells us how changes in ϵ will induce changes in $T(P + \epsilon(\tilde{P} - P))$. We'll use this influence curve to fit ϵ to get a better approximation of $T(P)$ within the likelihood framework. In particular, there is a specific **efficient influence curve (EIC)** that provides us with the lowest variance estimates of $T(P)$.

Affiliation:

Bernard Koch

UCLA Department of Sociology

E-mail: bernardkoch@ucla.edu

URL: https://soc.ucla.edu/grads/bernard-koch

SocArXiv Website https://socopen.org/

SocArXiv Preprints https://osf.io/preprints/socarxiv

Preprint *Submitted:* July 21, 2021

Accepted: July 21, 2021